

hw期间，在公司事情比较少，就把之前没有系统研究过的二进制格式、linux c程序的从源码到二进制文件的转换以及二进制文件的装载和执行，再从头部到尾捋一遍，把底层的原理弄清楚，让以前隐藏在迷雾中的摸棱两可的知识，一点一点呈现出来，也能让自己的基础更扎实些。

1. c源码设计

经过对各种设计类、架构类书籍的洗礼，应该可以“设计”出比较牛逼的软件架构了，然后就拿着各种编辑器啊、ide啊开始一顿写代码。

设计这块儿的内容，主要集中c的基础、c的高级技术、linux系统编程、linux网络编程、并行编程、IPC技术、内核编程等等，基础的编程技术的学习。还有操作系统、编译原理、网络原理、计算机体系结构、算法、架构设计、重构、系统分析、设计模式等等技术的学习。然后就是夜以继日无休止的撸代码，打副本升级。

2. c源码编写

然后在经历无数昼夜的百度、狗狗之后，终于把贼牛逼的架构实现了，虽然对写的什么东西一脸懵逼，但不耽误完成领导布置的任务，妹汁儿汁儿。

吭哧吭哧，终于把代码写完了，然后就是编译、执行。好像很自然的操作，但是这两部操作到底干了啥？我完全不知道，完全是傻子一样，等着计算机帮我处理好。所以，后面进入到linux c程序的编译阶段。

3. gcc预处理：cpp

4. gcc编译：cc1

5. gcc汇编：as

elf文件格式

汇编之后会产生relocatable file, relocatable file是c程序生命周期中第一个以elf格式存在的文件，后面还有executable file和shared object file都是以elf格式存在，并且在elf定义中，都属于object file，因此在这里记录elf文件格式。

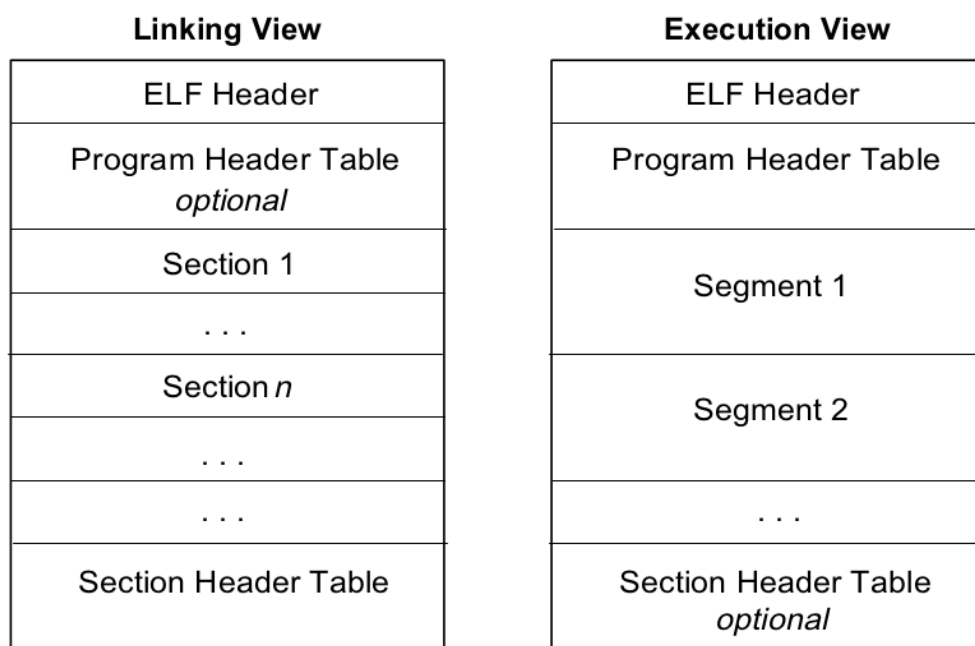
elf截至当前为止，分为两部分。一个是32位标准定义，一个是64位补充定义。在标准中没有定义的部分，属于实现相关的部分，可以参考linux系统下的/usr/include/elf.h中的定义。

[elf 32位标准定义](#)

[elf 64位补充定义](#)

鉴于目前64位已比较普遍，所以在记录时，直接合并32位和64位定义中的相关数据结构定义。

object file会参与到程序的链接和执行过程中，因此elf文件划分出链接视图和执行视图，两种视图来体现链接和执行过程中的不同要素。



OSD1980

ELF header: 描述了整个elf的结构和组织。

Sections: 包含所有“链接视图”所需的信息。

Segments: 包含所有“执行视图”所需的信息。

program header table: 定义如何创建process image。

section header table: 包含所有section的全部信息。

ELF Header

Table 1. ELF-64 Data Types

<i>Name</i>	<i>Size</i>	<i>Alignment</i>	<i>Purpose</i>
Elf64_Addr	8	8	Unsigned program address
Elf64_Off	8	8	Unsigned file offset
Elf64_Half	2	2	Unsigned medium integer
Elf64_Word	4	4	Unsigned integer
Elf64_Sword	4	4	Signed integer
Elf64_Xword	8	8	Unsigned long integer
Elf64_Sxword	8	8	Signed long integer
unsigned char	1	1	Unsigned small integer

```
typedef struct
```

```
{
```

```
    unsigned char    e_ident[16];    /* ELF identification */
    Elf64_Half       e_type;         /* Object file type */
    Elf64_Half       e_machine;      /* Machine type */
    Elf64_Word       e_version;      /* Object file version */
    Elf64_Addr       e_entry;        /* Entry point address */
    Elf64_Off        e_phoff;        /* Program header offset */
    Elf64_Off        e_shoff;        /* Section header offset */
    Elf64_Word       e_flags;        /* Processor-specific flags */
    Elf64_Half       e_ehsize;       /* ELF header size */
    Elf64_Half       e_phentsize;    /* Size of program header entry */
    Elf64_Half       e_phnum;        /* Number of program header entries */
    Elf64_Half       e_shentsize;    /* Size of section header entry */
    Elf64_Half       e_shnum;        /* Number of section header entries */
    Elf64_Half       e_shstrndx;     /* Section name string table index */
```

```
} Elf64_Ehdr;
```

Figure 2. ELF-64 Header**Table 2. ELF Identification, e_ident**

<i>Name</i>	<i>Value</i>	<i>Purpose</i>
EI_MAG0	0	File identification
EI_MAG1	1	
EI_MAG2	2	
EI_MAG3	3	
EI_CLASS	4	File class
EI_DATA	5	Data encoding
EI_VERSION	6	File version
EI_OSABI	7	OS/ABI identification
EI_ABIVERSION	8	ABI version
EI_PAD	9	Start of padding bytes
EI_NIDENT	16	Size of e_ident[]

Table 3. Object File Classes, e_ident[EI_CLASS]

<i>Name</i>	<i>Value</i>	<i>Meaning</i>
ELFCLASS32	1	32-bit objects
ELFCLASS64	2	64-bit objects

Table 4. Data Encodings, e_ident[EI_DATA]

<i>Name</i>	<i>Value</i>	<i>Meaning</i>
ELFDATA2LSB	1	Object file data structures are little-endian
ELFDATA2MSB	2	Object file data structures are big-endian

Table 5. Operating System and ABI Identifiers, e_ident[EI_OSABI]

<i>Name</i>	<i>Value</i>	<i>Meaning</i>
ELFOSABI_SYSV	0	System V ABI
ELFOSABI_HPUX	1	HP-UX operating system
ELFOSABI_STANDALONE	255	Standalone (embedded) application

Table 6. Object File Types, e_type

<i>Name</i>	<i>Value</i>	<i>Meaning</i>
ET_NONE	0	No file type
ET_REL	1	Relocatable object file
ET_EXEC	2	Executable file
ET_DYN	3	Shared object file
ET_CORE	4	Core file
ET_LOOS	0xFE00	Environment-specific use
ET_HIOS	0xFEFF	
ET_LOPROC	0xFF00	Processor-specific use
ET_HIPROC	0xFFFF	

Name	Value	Meaning
ET_NONE	0	No machine
EM_M32	1	AT&T WE 32100
EM_SPARC	2	SPARC
EM_386	3	Intel Architecture
EM_68K	4	Motorola 68000
EM_88K	5	Motorola 88000
EM_860	7	Intel 80860
EM_MIPS	8	MIPS RS3000 Big-Endian
EM_MIPS_RS4_BE	10	MIPS RS4000 Big-Endian
RESERVED	11-16	Reserved for future use

实例:

```
[root@localhost tmp]# readelf -h /bin/ls
ELF Header:
  Magic:   7f 45 4c 46 02 01 01 00 00 00 00 00 00 00 00 00
  Class:                               ELF64
  Data:                                   2's complement, little endian
  Version:                               1 (current)
  OS/ABI:                                UNIX - System V
  ABI Version:                           0
  Type:                                  EXEC (Executable file)
  Machine:                               Advanced Micro Devices X86-64
  Version:                               0x1
  Entry point address:                   0x404324
  Start of program headers:              64 (bytes into file)
  Start of section headers:              115688 (bytes into file)
  Flags:                                 0x0
  Size of this header:                   64 (bytes)
  Size of program headers:               56 (bytes)
  Number of program headers:              9
  Size of section headers:               64 (bytes)
  Number of section headers:              30
  Section header string table index: 29
```

```
[root@localhost tmp]# xxd -s 0x0 -l 0x40 /bin/ls
00000000: 7f45 4c46 0201 0100 0000 0000 0000 0000  .ELF.....
00000010: 0200 3e00 0100 0000 2443 4000 0000 0000  ..>.....$c@....
00000020: 4000 0000 0000 0000 e8c3 0100 0000 0000  @.....
00000030: 0000 0000 4000 3800 0900 4000 1e00 1d00  ....@.8...@.....
```

Section

```

typedef struct
{
    Elf64_Word    sh_name;        /* Section name */
    Elf64_Word    sh_type;        /* Section type */
    Elf64_Xword   sh_flags;       /* Section attributes */
    Elf64_Addr    sh_addr;        /* Virtual address in memory */
    Elf64_Off     sh_offset;      /* Offset in file */
    Elf64_Xword   sh_size;        /* Size of section */
    Elf64_Word    sh_link;        /* Link to other section */
    Elf64_Word    sh_info;        /* Miscellaneous information */
    Elf64_Xword   sh_addralign;   /* Address alignment boundary */
    Elf64_Xword   sh_entsize;     /* Size of entries, if section has table */
} Elf64_Shdr;

```

Figure 3. ELF-64 Section Header

Table 8. Section Types, sh_type

<i>Name</i>	<i>Value</i>	<i>Meaning</i>
SHT_NULL	0	Marks an unused section header
SHT_PROGBITS	1	Contains information defined by the program
SHT_SYMTAB	2	Contains a linker symbol table
SHT_STRTAB	3	Contains a string table
SHT_RELA	4	Contains “Rela” type relocation entries
SHT_HASH	5	Contains a symbol hash table
SHT_DYNAMIC	6	Contains dynamic linking tables
SHT_NOTE	7	Contains note information
SHT_NOBITS	8	Contains uninitialized space; does not occupy any space in the file
SHT_REL	9	Contains “Rel” type relocation entries
SHT_SHLIB	10	Reserved
SHT_DYNSYM	11	Contains a dynamic loader symbol table
SHT_LOOS	0x6000 0000	Environment-specific use
SHT_HIOS	0x6FFFFFFF	
SHT_LOPROC	0x7000 0000	Processor-specific use
SHT_HIPROC	0x7FFFFFFF	

Table 9. Section Attributes, sh_flags

<i>Name</i>	<i>Value</i>	<i>Meaning</i>
SHF_WRITE	0x1	Section contains writable data
SHF_ALLOC	0x2	Section is allocated in memory image of program
SHF_EXECINSTR	0x4	Section contains executable instructions
SHF_MASKOS	0x0F000000	Environment-specific use
SHF_MASKPROC	0xF0000000	Processor-specific use

Table 10. Use of the sh_link Field

<i>Section Type</i>	<i>Associated Section</i>
SHT_DYNAMIC	String table used by entries in this section
SHT_HASH	Symbol table to which the hash table applies
SHT_REL	Symbol table referenced by relocations
SHT_RELA	
SHT_SYMTAB	String table used by entries in this section
SHT_DYNSYM	
Other	SHN_UNDEF

Table 11. Use of the sh_info Field

<i>Section Type</i>	<i>sh_info</i>
SHT_REL	Section index of section to which the relocations apply
SHT_RELA	
SHT_SYMTAB	Index of first non-local symbol (i.e., number of local symbols)
SHT_DYNSYM	
Other	0

Table 12. Standard Sections for Code and Data

<i>Section Name</i>	<i>Section Type</i>	<i>Flags</i>	<i>Use</i>
.bss	SHT_NOBITS	A, W	Uninitialized data
.data	SHT_PROGBITS	A, W	Initialized data
.interp	SHT_PROGBITS	[A]	Program interpreter path name
.rodata	SHT_PROGBITS	A	Read-only data (constants and literals)
.text	SHT_PROGBITS	A, X	Executable code

Table 13. Other Standard Sections

Section Name	Section Type	Flags	Use
.comment	SHT_PROGBITS	none	Version control information
.dynamic	SHT_DYNAMIC	A[, W]	Dynamic linking tables
.dynstr	SHT_STRTAB	A	String table for .dynamic section
.dynsym	SHT_DYNSYM	A	Symbol table for dynamic linking
.got	SHT_PROGBITS	mach. dep.	Global offset table
.hash	SHT_HASH	A	Symbol hash table
.note	SHT_NOTE	none	Note section
.plt	SHT_PROGBITS	mach. dep.	Procedure linkage table
.relname	SHT_REL	[A]	Relocations for section <i>name</i>
.relaname	SHT_RELA		
.shstrtab	SHT_STRTAB	none	Section name string table
.strtab	SHT_STRTAB	none	String table
.symtab	SHT_SYMTAB	[A]	Linker symbol table

```
[root@localhost 3]# readelf -s /bin/ls
```

There are 30 section headers, starting at offset 0x1c3e8:

Section Headers:

[Nr]	Name	Type	Address	Offset
	Size	EntSize	Flags Link Info Align	
[0]		NULL	0000000000000000	00000000
	0000000000000000	0000000000000000	0 0 0	
[1]	.interp	PROGBITS	0000000000400238	00000238
	000000000000001c	0000000000000000	A 0 0 1	
[2]	.note.ABI-tag	NOTE	0000000000400254	00000254
	0000000000000020	0000000000000000	A 0 0 4	
[3]	.note.gnu.build-id	NOTE	0000000000400274	00000274
	0000000000000024	0000000000000000	A 0 0 4	
[4]	.gnu.hash	GNU_HASH	0000000000400298	00000298
	0000000000000038	0000000000000000	A 5 0 8	
[5]	.dynsym	DYNSYM	00000000004002d0	000002d0
	00000000000000c18	0000000000000018	A 6 1 8	
[6]	.dynstr	STRTAB	0000000000400ee8	00000ee8
	0000000000000572	0000000000000000	A 0 0 1	
[7]	.gnu.version	VERSYM	000000000040145a	0000145a
	0000000000000102	0000000000000002	A 5 0 2	
[8]	.gnu.version_r	VERNEED	0000000000401560	00001560
	0000000000000090	0000000000000000	A 6 2 8	
[9]	.rela.dyn	RELA	00000000004015f0	000015f0
	00000000000000d8	0000000000000018	A 5 0 8	
[10]	.rela.plt	RELA	00000000004016c8	000016c8
	00000000000000ac8	0000000000000018	AI 5 24 8	
[11]	.init	PROGBITS	0000000000402190	00002190
	000000000000001a	0000000000000000	AX 0 0 4	
[12]	.plt	PROGBITS	00000000004021b0	000021b0
	0000000000000740	0000000000000010	AX 0 0 16	
[13]	.text	PROGBITS	00000000004028f0	000028f0
	000000000001014a	0000000000000000	AX 0 0 16	
[14]	.fini	PROGBITS	0000000000412a3c	00012a3c
	0000000000000009	0000000000000000	AX 0 0 4	
[15]	.rodata	PROGBITS	0000000000412a60	00012a60
	00000000000003cce	0000000000000000	A 0 0 32	
[16]	.eh_frame_hdr	PROGBITS	0000000000416730	00016730
	0000000000000754	0000000000000000	A 0 0 4	

[17]	.eh_frame	PROGBITS	0000000000416e88	00016e88
	0000000000002704	0000000000000000	A 0 0 8	
[18]	.init_array	INIT_ARRAY	0000000000061a328	0001a328
	0000000000000008	0000000000000008	WA 0 0 8	
[19]	.fini_array	FINI_ARRAY	0000000000061a330	0001a330
	0000000000000008	0000000000000008	WA 0 0 8	
[20]	.jcr	PROGBITS	0000000000061a338	0001a338
	0000000000000008	0000000000000000	WA 0 0 8	
[21]	.data.rel.ro	PROGBITS	0000000000061a340	0001a340
	00000000000000a68	0000000000000000	WA 0 0 32	
[22]	.dynamic	DYNAMIC	0000000000061ada8	0001ada8
	0000000000000200	0000000000000010	WA 6 0 8	
[23]	.got	PROGBITS	0000000000061afa8	0001afa8
	0000000000000048	0000000000000008	WA 0 0 8	
[24]	.got.plt	PROGBITS	0000000000061b000	0001b000
	000000000000003b0	0000000000000008	WA 0 0 8	
[25]	.data	PROGBITS	0000000000061b3c0	0001b3c0
	0000000000000240	0000000000000000	WA 0 0 32	
[26]	.bss	NOBITS	0000000000061b600	0001b600
	00000000000000d20	0000000000000000	WA 0 0 32	
[27]	.gnu_debuglink	PROGBITS	00000000000000000	0001b600
	0000000000000010	0000000000000000	0 0 4	
[28]	.gnu_debugdata	PROGBITS	00000000000000000	0001b610
	00000000000000cb8	0000000000000000	0 0 1	
[29]	.shstrtab	STRTAB	00000000000000000	0001c2c8
	000000000000011a	0000000000000000	0 0 1	

Key to Flags:

W (write), A (alloc), X (execute), M (merge), S (strings), I (info),
L (link order), O (extra OS processing required), G (group), T (TLS),
C (compressed), x (unknown), o (OS specific), E (exclude),
l (large), p (processor specific)

```
[root@localhost 3]# xxd -s 0x1c3e8 -l 0x780 /bin/ls
001c3e8: 0000 0000 0000 0000 0000 0000 0000 0000 .....
001c3f8: 0000 0000 0000 0000 0000 0000 0000 0000 .....
001c408: 0000 0000 0000 0000 0000 0000 0000 0000 .....
001c418: 0000 0000 0000 0000 0000 0000 0000 0000 .....
001c428: 0b00 0000 0100 0000 0200 0000 0000 0000 .....
001c438: 3802 4000 0000 0000 3802 0000 0000 0000 8.@.....8.....
001c448: 1c00 0000 0000 0000 0000 0000 0000 0000 .....
001c458: 0100 0000 0000 0000 0000 0000 0000 0000 .....
001c468: 1300 0000 0700 0000 0200 0000 0000 0000 .....
001c478: 5402 4000 0000 0000 5402 0000 0000 0000 T.@.....T.....
001c488: 2000 0000 0000 0000 0000 0000 0000 0000 .....
001c498: 0400 0000 0000 0000 0000 0000 0000 0000 .....
001c4a8: 2100 0000 0700 0000 0200 0000 0000 0000 !.....
001c4b8: 7402 4000 0000 0000 7402 0000 0000 0000 t.@.....t.....
001c4c8: 2400 0000 0000 0000 0000 0000 0000 0000 $......
001c4d8: 0400 0000 0000 0000 0000 0000 0000 0000 .....
001c4e8: 3400 0000 f6ff ff6f 0200 0000 0000 0000 4.....o.....
001c4f8: 9802 4000 0000 0000 9802 0000 0000 0000 ..@.....
001c508: 3800 0000 0000 0000 0500 0000 0000 0000 8.....
001c518: 0800 0000 0000 0000 0000 0000 0000 0000 .....
001c528: 3e00 0000 0b00 0000 0200 0000 0000 0000 >.....
001c538: d002 4000 0000 0000 d002 0000 0000 0000 ..@.....
001c548: 180c 0000 0000 0000 0600 0000 0100 0000 .....
001c558: 0800 0000 0000 0000 1800 0000 0000 0000 .....
```

001c568:	4600	0000	0300	0000	0200	0000	0000	0000	F.....
001c578:	e80e	4000	0000	0000	e80e	0000	0000	0000	..@.....
001c588:	7205	0000	0000	0000	0000	0000	0000	0000	r.....
001c598:	0100	0000	0000	0000	0000	0000	0000	0000
001c5a8:	4e00	0000	ffff	ff6f	0200	0000	0000	0000	N.....o.....
001c5b8:	5a14	4000	0000	0000	5a14	0000	0000	0000	Z.@.....Z.....
001c5c8:	0201	0000	0000	0000	0500	0000	0000	0000
001c5d8:	0200	0000	0000	0000	0200	0000	0000	0000
001c5e8:	5b00	0000	feff	ff6f	0200	0000	0000	0000	[.....o.....
001c5f8:	6015	4000	0000	0000	6015	0000	0000	0000	`.@.....`.....
001c608:	9000	0000	0000	0000	0600	0000	0200	0000
001c618:	0800	0000	0000	0000	0000	0000	0000	0000
001c628:	6a00	0000	0400	0000	0200	0000	0000	0000	j.....
001c638:	f015	4000	0000	0000	f015	0000	0000	0000	..@.....
001c648:	d800	0000	0000	0000	0500	0000	0000	0000
001c658:	0800	0000	0000	0000	1800	0000	0000	0000
001c668:	7400	0000	0400	0000	4200	0000	0000	0000	t.....B.....
001c678:	c816	4000	0000	0000	c816	0000	0000	0000	..@.....
001c688:	c80a	0000	0000	0000	0500	0000	1800	0000
001c698:	0800	0000	0000	0000	1800	0000	0000	0000
001c6a8:	7e00	0000	0100	0000	0600	0000	0000	0000	~.....
001c6b8:	9021	4000	0000	0000	9021	0000	0000	0000	.!@.....!.....
001c6c8:	1a00	0000	0000	0000	0000	0000	0000	0000
001c6d8:	0400	0000	0000	0000	0000	0000	0000	0000
001c6e8:	7900	0000	0100	0000	0600	0000	0000	0000	y.....
001c6f8:	b021	4000	0000	0000	b021	0000	0000	0000	.!@.....!.....
001c708:	4007	0000	0000	0000	0000	0000	0000	0000	@.....
001c718:	1000	0000	0000	0000	1000	0000	0000	0000
001c728:	8400	0000	0100	0000	0600	0000	0000	0000
001c738:	f028	4000	0000	0000	f028	0000	0000	0000	.(@.....(.....
001c748:	4a01	0100	0000	0000	0000	0000	0000	0000	J.....
001c758:	1000	0000	0000	0000	0000	0000	0000	0000
001c768:	8a00	0000	0100	0000	0600	0000	0000	0000
001c778:	3c2a	4100	0000	0000	3c2a	0100	0000	0000	<*A.....<*
001c788:	0900	0000	0000	0000	0000	0000	0000	0000
001c798:	0400	0000	0000	0000	0000	0000	0000	0000
001c7a8:	9000	0000	0100	0000	0200	0000	0000	0000
001c7b8:	602a	4100	0000	0000	602a	0100	0000	0000	`*A.....`*.....
001c7c8:	ce3c	0000	0000	0000	0000	0000	0000	0000	.<.....
001c7d8:	2000	0000	0000	0000	0000	0000	0000	0000
001c7e8:	9800	0000	0100	0000	0200	0000	0000	0000
001c7f8:	3067	4100	0000	0000	3067	0100	0000	0000	0gA.....0g.....
001c808:	5407	0000	0000	0000	0000	0000	0000	0000	T.....
001c818:	0400	0000	0000	0000	0000	0000	0000	0000
001c828:	a600	0000	0100	0000	0200	0000	0000	0000
001c838:	886e	4100	0000	0000	886e	0100	0000	0000	.nA.....n.....
001c848:	0427	0000	0000	0000	0000	0000	0000	0000	.'......
001c858:	0800	0000	0000	0000	0000	0000	0000	0000
001c868:	b000	0000	0e00	0000	0300	0000	0000	0000
001c878:	28a3	6100	0000	0000	28a3	0100	0000	0000	(.a.....(.....
001c888:	0800	0000	0000	0000	0000	0000	0000	0000
001c898:	0800	0000	0000	0000	0800	0000	0000	0000
001c8a8:	bc00	0000	0f00	0000	0300	0000	0000	0000
001c8b8:	30a3	6100	0000	0000	30a3	0100	0000	0000	0.a.....0.....
001c8c8:	0800	0000	0000	0000	0000	0000	0000	0000
001c8d8:	0800	0000	0000	0000	0800	0000	0000	0000
001c8e8:	c800	0000	0100	0000	0300	0000	0000	0000
001c8f8:	38a3	6100	0000	0000	38a3	0100	0000	0000	8.a.....8.....

```

001c908: 0800 0000 0000 0000 0000 0000 0000 0000 .....
001c918: 0800 0000 0000 0000 0000 0000 0000 0000 .....
001c928: cd00 0000 0100 0000 0300 0000 0000 0000 .....
001c938: 40a3 6100 0000 0000 40a3 0100 0000 0000 @.a.....@.....
001c948: 680a 0000 0000 0000 0000 0000 0000 0000 h.....
001c958: 2000 0000 0000 0000 0000 0000 0000 0000 .....
001c968: da00 0000 0600 0000 0300 0000 0000 0000 .....
001c978: a8ad 6100 0000 0000 a8ad 0100 0000 0000 ..a.....
001c988: 0002 0000 0000 0000 0600 0000 0000 0000 .....
001c998: 0800 0000 0000 0000 1000 0000 0000 0000 .....
001c9a8: e300 0000 0100 0000 0300 0000 0000 0000 .....
001c9b8: a8af 6100 0000 0000 a8af 0100 0000 0000 ..a.....
001c9c8: 4800 0000 0000 0000 0000 0000 0000 0000 H.....
001c9d8: 0800 0000 0000 0000 0800 0000 0000 0000 .....
001c9e8: e800 0000 0100 0000 0300 0000 0000 0000 .....
001c9f8: 00b0 6100 0000 0000 00b0 0100 0000 0000 ..a.....
001ca08: b003 0000 0000 0000 0000 0000 0000 0000 .....
001ca18: 0800 0000 0000 0000 0800 0000 0000 0000 .....
001ca28: f100 0000 0100 0000 0300 0000 0000 0000 .....
001ca38: c0b3 6100 0000 0000 c0b3 0100 0000 0000 ..a.....
001ca48: 4002 0000 0000 0000 0000 0000 0000 0000 @.....
001ca58: 2000 0000 0000 0000 0000 0000 0000 0000 .....
001ca68: f700 0000 0800 0000 0300 0000 0000 0000 .....
001ca78: 00b6 6100 0000 0000 00b6 0100 0000 0000 ..a.....
001ca88: 200d 0000 0000 0000 0000 0000 0000 0000 .....
001ca98: 2000 0000 0000 0000 0000 0000 0000 0000 .....
001caa8: fc00 0000 0100 0000 0000 0000 0000 0000 .....
001cab8: 0000 0000 0000 0000 00b6 0100 0000 0000 .....
001cac8: 1000 0000 0000 0000 0000 0000 0000 0000 .....
001cad8: 0400 0000 0000 0000 0000 0000 0000 0000 .....
001cae8: 0b01 0000 0100 0000 0000 0000 0000 0000 .....
001caf8: 0000 0000 0000 0000 10b6 0100 0000 0000 .....
001cb08: b80c 0000 0000 0000 0000 0000 0000 0000 .....
001cb18: 0100 0000 0000 0000 0000 0000 0000 0000 .....
001cb28: 0100 0000 0300 0000 0000 0000 0000 0000 .....
001cb38: 0000 0000 0000 0000 c8c2 0100 0000 0000 .....
001cb48: 1a01 0000 0000 0000 0000 0000 0000 0000 .....
001cb58: 0100 0000 0000 0000 0000 0000 0000 0000 .....

```

Symbol Table

符号表包含了所有的在文件中需要定位和重定位的符号定义和引用的信息。

```

typedef struct
{
    Elf64_Word      st_name;      /* Symbol name */
    unsigned char    st_info;     /* Type and Binding attributes */
    unsigned char    st_other;    /* Reserved */
    Elf64_Half      st_shndx;     /* Section table index */
    Elf64_Addr      st_value;     /* Symbol value */
    Elf64_Xword     st_size;     /* Size of object (e.g., common) */
} Elf64_Sym;

```

Figure 4. ELF-64 Symbol Table Entry

Table 14. Symbol Bindings

<i>Name</i>	<i>Value</i>	<i>Meaning</i>
STB_LOCAL	0	Not visible outside the object file
STB_GLOBAL	1	Global symbol, visible to all object files
STB_WEAK	2	Global scope, but with lower precedence than global symbols
STB_LOOS	10	Environment-specific use
STB_HIOS	12	
STB_LOPROC	13	Processor-specific use
STB_HIPROC	15	

Table 15. Symbol Types

<i>Name</i>	<i>Value</i>	<i>Meaning</i>
STT_NOTYPE	0	No type specified (e.g., an absolute symbol)
STT_OBJECT	1	Data object
STT_FUNC	2	Function entry point
STT_SECTION	3	Symbol is associated with a section

Table 15. Symbol Types (Continued)

<i>Name</i>	<i>Value</i>	<i>Meaning</i>
STT_FILE	4	Source file associated with the object file
STT_LOOS	10	Environment-specific use
STT_HIOS	12	
STT_LOPROC	13	Processor-specific use
STT_HIPROC	15	

```
[root@localhost tmp]# cat HelloWorld.c
```

```
#include <stdio.h>

int main(void) {
    printf("Hello world!\n");

    return 0;
}
```

```
[root@localhost tmp]# gcc HelloWorld.c -o HelloWorld
```

```
[root@localhost tmp]# readelf -S HelloWorld
[26] .symtab          SYMTAB          0000000000000000  00001080
      00000000000005a0 0000000000000018          27   43   8
```

```
[root@localhost tmp]# readelf -s HelloWorld
```

Symbol table '.dynsym' contains 4 entries:

Num:	Value	Size	Type	Bind	Vis	Ndx	Name
0:	0000000000000000	0	NOTYPE	LOCAL	DEFAULT	UND	
1:	0000000000000000	0	FUNC	GLOBAL	DEFAULT	UND	puts@GLIBC_2.2.5 (2)
2:	0000000000000000	0	FUNC	GLOBAL	DEFAULT	UND	
__libc_start_main@GLIBC_2.2.5 (2)							
3:	0000000000000000	0	NOTYPE	WEAK	DEFAULT	UND	__gmon_start__

Symbol table '.symtab' contains 60 entries:

Num:	Value	Size	Type	Bind	Vis	Ndx	Name
0:	0000000000000000	0	NOTYPE	LOCAL	DEFAULT	UND	
1:	0000000000400238	0	SECTION	LOCAL	DEFAULT	1	
2:	0000000000400254	0	SECTION	LOCAL	DEFAULT	2	
3:	0000000000400274	0	SECTION	LOCAL	DEFAULT	3	
4:	0000000000400298	0	SECTION	LOCAL	DEFAULT	4	
5:	00000000004002b8	0	SECTION	LOCAL	DEFAULT	5	
6:	0000000000400318	0	SECTION	LOCAL	DEFAULT	6	
7:	0000000000400356	0	SECTION	LOCAL	DEFAULT	7	
8:	0000000000400360	0	SECTION	LOCAL	DEFAULT	8	
9:	0000000000400380	0	SECTION	LOCAL	DEFAULT	9	
10:	0000000000400398	0	SECTION	LOCAL	DEFAULT	10	
11:	00000000004003e0	0	SECTION	LOCAL	DEFAULT	11	
12:	0000000000400400	0	SECTION	LOCAL	DEFAULT	12	
13:	0000000000400440	0	SECTION	LOCAL	DEFAULT	13	
14:	00000000004005a4	0	SECTION	LOCAL	DEFAULT	14	
15:	00000000004005b0	0	SECTION	LOCAL	DEFAULT	15	
16:	00000000004005c4	0	SECTION	LOCAL	DEFAULT	16	
17:	00000000004005f8	0	SECTION	LOCAL	DEFAULT	17	
18:	0000000000600e18	0	SECTION	LOCAL	DEFAULT	18	
19:	0000000000600e20	0	SECTION	LOCAL	DEFAULT	19	
20:	0000000000600e28	0	SECTION	LOCAL	DEFAULT	20	
21:	0000000000600ff8	0	SECTION	LOCAL	DEFAULT	21	
22:	0000000000601000	0	SECTION	LOCAL	DEFAULT	22	
23:	0000000000601030	0	SECTION	LOCAL	DEFAULT	23	
24:	0000000000601040	0	SECTION	LOCAL	DEFAULT	24	
25:	0000000000000000	0	SECTION	LOCAL	DEFAULT	25	
26:	0000000000000000	0	FILE	LOCAL	DEFAULT	ABS	crtstuff.c
27:	0000000000400470	0	FUNC	LOCAL	DEFAULT	13	deregister_tm_clones
28:	00000000004004a0	0	FUNC	LOCAL	DEFAULT	13	register_tm_clones
29:	00000000004004e0	0	FUNC	LOCAL	DEFAULT	13	__do_global_dtors_aux
30:	0000000000601040	1	OBJECT	LOCAL	DEFAULT	24	completed.7311
31:	0000000000600e20	0	OBJECT	LOCAL	DEFAULT	19	
__do_global_dtors_aux_fin							
32:	0000000000400510	0	FUNC	LOCAL	DEFAULT	13	frame_dummy
33:	0000000000600e18	0	OBJECT	LOCAL	DEFAULT	18	
__frame_dummy_init_array_							
34:	0000000000000000	0	FILE	LOCAL	DEFAULT	ABS	Helloworld.c
35:	0000000000000000	0	FILE	LOCAL	DEFAULT	ABS	crtstuff.c
36:	00000000004006e8	0	OBJECT	LOCAL	DEFAULT	17	__FRAME_END__
37:	0000000000000000	0	FILE	LOCAL	DEFAULT	ABS	
38:	0000000000600e20	0	NOTYPE	LOCAL	DEFAULT	18	__init_array_end
39:	0000000000600e28	0	OBJECT	LOCAL	DEFAULT	20	__DYNAMIC
40:	0000000000600e18	0	NOTYPE	LOCAL	DEFAULT	18	__init_array_start
41:	00000000004005c4	0	NOTYPE	LOCAL	DEFAULT	16	__GNU_EH_FRAME_HDR
42:	0000000000601000	0	OBJECT	LOCAL	DEFAULT	22	__GLOBAL_OFFSET_TABLE__
43:	00000000004005a0	2	FUNC	GLOBAL	DEFAULT	13	__libc_csu_fini

44:	0000000000601030	0	NOTYPE	WEAK	DEFAULT	23	data_start
45:	0000000000000000	0	FUNC	GLOBAL	DEFAULT	UND	puts@@GLIBC_2.2.5
46:	0000000000601040	0	NOTYPE	GLOBAL	DEFAULT	23	_edata
47:	00000000004005a4	0	FUNC	GLOBAL	DEFAULT	14	_fini
48:	0000000000000000	0	FUNC	GLOBAL	DEFAULT	UND	
__libc_start_main@@GLIBC_							
49:	0000000000601030	0	NOTYPE	GLOBAL	DEFAULT	23	__data_start
50:	0000000000000000	0	NOTYPE	WEAK	DEFAULT	UND	__gmon_start__
51:	0000000000601038	0	OBJECT	GLOBAL	HIDDEN	23	__dso_handle
52:	00000000004005b0	4	OBJECT	GLOBAL	DEFAULT	15	_IO_stdin_used
53:	0000000000400530	101	FUNC	GLOBAL	DEFAULT	13	__libc_csu_init
54:	0000000000601048	0	NOTYPE	GLOBAL	DEFAULT	24	_end
55:	0000000000400440	0	FUNC	GLOBAL	DEFAULT	13	_start
56:	0000000000601040	0	NOTYPE	GLOBAL	DEFAULT	24	__bss_start
57:	0000000000400512	21	FUNC	GLOBAL	DEFAULT	13	main
58:	0000000000601040	0	OBJECT	GLOBAL	HIDDEN	23	__TMC_END__
59:	00000000004003e0	0	FUNC	GLOBAL	DEFAULT	11	_init

```
[root@localhost tmp]# xxd -s 0x1080 -l 0x5a0 HelloWorld
```

```
0001080: 0000 0000 0000 0000 0000 0000 0000 0000  .....
0001090: 0000 0000 0000 0000 0000 0000 0300 0100  .....
00010a0: 3802 4000 0000 0000 0000 0000 0000 0000  8.@.....
00010b0: 0000 0000 0300 0200 5402 4000 0000 0000  .....T.@....
00010c0: 0000 0000 0000 0000 0000 0000 0300 0300  .....
00010d0: 7402 4000 0000 0000 0000 0000 0000 0000  t.@.....
00010e0: 0000 0000 0300 0400 9802 4000 0000 0000  .....@.....
00010f0: 0000 0000 0000 0000 0000 0000 0300 0500  .....
0001100: b802 4000 0000 0000 0000 0000 0000 0000  ..@.....
0001110: 0000 0000 0300 0600 1803 4000 0000 0000  .....@.....
0001120: 0000 0000 0000 0000 0000 0000 0300 0700  .....
0001130: 5603 4000 0000 0000 0000 0000 0000 0000  V.@.....
0001140: 0000 0000 0300 0800 6003 4000 0000 0000  .....`.@.....
0001150: 0000 0000 0000 0000 0000 0000 0300 0900  .....
0001160: 8003 4000 0000 0000 0000 0000 0000 0000  ..@.....
0001170: 0000 0000 0300 0a00 9803 4000 0000 0000  .....@.....
0001180: 0000 0000 0000 0000 0000 0000 0300 0b00  .....
0001190: e003 4000 0000 0000 0000 0000 0000 0000  ..@.....
00011a0: 0000 0000 0300 0c00 0004 4000 0000 0000  .....@.....
00011b0: 0000 0000 0000 0000 0000 0000 0300 0d00  .....
00011c0: 4004 4000 0000 0000 0000 0000 0000 0000  @.@.....
00011d0: 0000 0000 0300 0e00 a405 4000 0000 0000  .....@.....
00011e0: 0000 0000 0000 0000 0000 0000 0300 0f00  .....
00011f0: b005 4000 0000 0000 0000 0000 0000 0000  ..@.....
0001200: 0000 0000 0300 1000 c405 4000 0000 0000  .....@.....
0001210: 0000 0000 0000 0000 0000 0000 0300 1100  .....
0001220: f805 4000 0000 0000 0000 0000 0000 0000  ..@.....
0001230: 0000 0000 0300 1200 180e 6000 0000 0000  .....`.....
0001240: 0000 0000 0000 0000 0000 0000 0300 1300  .....
0001250: 200e 6000 0000 0000 0000 0000 0000 0000  .`.....
0001260: 0000 0000 0300 1400 280e 6000 0000 0000  .....(.`.....
0001270: 0000 0000 0000 0000 0000 0000 0300 1500  .....
0001280: f80f 6000 0000 0000 0000 0000 0000 0000  ..`.....
0001290: 0000 0000 0300 1600 0010 6000 0000 0000  .....`.....
00012a0: 0000 0000 0000 0000 0000 0000 0300 1700  .....
00012b0: 3010 6000 0000 0000 0000 0000 0000 0000  0.`.....
00012c0: 0000 0000 0300 1800 4010 6000 0000 0000  .....@.`.....
00012d0: 0000 0000 0000 0000 0000 0000 0300 1900  .....
```

```
00012e0: 0000 0000 0000 0000 0000 0000 0000 0000 .....
00012f0: 0100 0000 0400 f1ff 0000 0000 0000 0000 .....
0001300: 0000 0000 0000 0000 0c00 0000 0200 0d00 .....
0001310: 7004 4000 0000 0000 0000 0000 0000 0000 p.@.....
0001320: 0e00 0000 0200 0d00 a004 4000 0000 0000 .....@.....
0001330: 0000 0000 0000 0000 2100 0000 0200 0d00 .....!.....
0001340: e004 4000 0000 0000 0000 0000 0000 0000 ..@.....
0001350: 3700 0000 0100 1800 4010 6000 0000 0000 7.....@.`.....
0001360: 0100 0000 0000 0000 4600 0000 0100 1300 .....F.....
0001370: 200e 6000 0000 0000 0000 0000 0000 0000 .`.....
0001380: 6d00 0000 0200 0d00 1005 4000 0000 0000 m.....@.....
0001390: 0000 0000 0000 0000 7900 0000 0100 1200 .....y.....
00013a0: 180e 6000 0000 0000 0000 0000 0000 0000 ..`.....
00013b0: 9800 0000 0400 f1ff 0000 0000 0000 0000 .....
00013c0: 0000 0000 0000 0000 0100 0000 0400 f1ff .....
00013d0: 0000 0000 0000 0000 0000 0000 0000 0000 .....
00013e0: a500 0000 0100 1100 e806 4000 0000 0000 .....@.....
00013f0: 0000 0000 0000 0000 0000 0000 0400 f1ff .....
0001400: 0000 0000 0000 0000 0000 0000 0000 0000 .....
0001410: b300 0000 0000 1200 200e 6000 0000 0000 .....`.....
0001420: 0000 0000 0000 0000 c400 0000 0100 1400 .....
0001430: 280e 6000 0000 0000 0000 0000 0000 0000 (.`.....
0001440: cd00 0000 0000 1200 180e 6000 0000 0000 .....`.....
0001450: 0000 0000 0000 0000 e000 0000 0000 1000 .....
0001460: c405 4000 0000 0000 0000 0000 0000 0000 ..@.....
0001470: f300 0000 0100 1600 0010 6000 0000 0000 .....`.....
0001480: 0000 0000 0000 0000 0901 0000 1200 0d00 .....
0001490: a005 4000 0000 0000 0200 0000 0000 0000 ..@.....
00014a0: 5301 0000 2000 1700 3010 6000 0000 0000 S... ..0.`.....
00014b0: 0000 0000 0000 0000 1901 0000 1200 0000 .....
00014c0: 0000 0000 0000 0000 0000 0000 0000 0000 .....
00014d0: 2b01 0000 1000 1700 4010 6000 0000 0000 +.....@.`.....
00014e0: 0000 0000 0000 0000 1301 0000 1200 0e00 .....
00014f0: a405 4000 0000 0000 0000 0000 0000 0000 ..@.....
0001500: 3201 0000 1200 0000 0000 0000 0000 0000 2.....
0001510: 0000 0000 0000 0000 5101 0000 1000 1700 .....Q.....
0001520: 3010 6000 0000 0000 0000 0000 0000 0000 0.`.....
0001530: 5e01 0000 2000 0000 0000 0000 0000 0000 ^.....
0001540: 0000 0000 0000 0000 6d01 0000 1102 1700 .....m.....
0001550: 3810 6000 0000 0000 0000 0000 0000 0000 8.`.....
0001560: 7a01 0000 1100 0f00 b005 4000 0000 0000 z.....@.....
0001570: 0400 0000 0000 0000 8901 0000 1200 0d00 .....
0001580: 3005 4000 0000 0000 6500 0000 0000 0000 0.@....e.....
0001590: bf00 0000 1000 1800 4810 6000 0000 0000 .....H.`.....
00015a0: 0000 0000 0000 0000 5701 0000 1200 0d00 .....W.....
00015b0: 4004 4000 0000 0000 0000 0000 0000 0000 @.@.....
00015c0: 9901 0000 1000 1800 4010 6000 0000 0000 .....@.`.....
00015d0: 0000 0000 0000 0000 a501 0000 1200 0d00 .....
00015e0: 1205 4000 0000 0000 1500 0000 0000 0000 ..@.....
00015f0: aa01 0000 1102 1700 4010 6000 0000 0000 .....@.`.....
0001600: 0000 0000 0000 0000 9301 0000 1200 0b00 .....
0001610: e003 4000 0000 0000 0000 0000 0000 0000 ..@.....
```

Relocations

```
typedef struct
{
    Elf64_Addr    r_offset;    /* Address of reference */
    Elf64_Xword   r_info;     /* Symbol index and type of relocation */
} Elf64_Rel;

typedef struct
{
    Elf64_Addr    r_offset;    /* Address of reference */
    Elf64_Xword   r_info;     /* Symbol index and type of relocation */
    Elf64_Sxword  r_addend;    /* Constant part of expression */
} Elf64_Rela;
```

Figure 5. ELF-64 Relocation Entries

```
#define ELF64_R_SYM(i)((i) >> 32)
#define ELF64_R_TYPE(i)((i) & 0xffffffffL)
#define ELF64_R_INFO(s, t)(((s) << 32) + ((t) & 0xffffffffL))
```

```
[root@localhost tmp]# gcc -c HelloWorld.c -o HelloWorld.o
[root@localhost tmp]# readelf -S HelloWorld.o
```

```
[ 2] .rela.text          RELA             0000000000000000 000001d0
      0000000000000030 0000000000000018 I      10      1      8
[ 9] .rela.eh_frame      RELA             0000000000000000 00000200
      0000000000000018 0000000000000018 I      10      8      8
```

```
[root@localhost tmp]# readelf -r HelloWorld.o
```

Relocation section **'*.rela.text*'** at offset 0x1d0 contains 2 entries:

Offset	Info	Type	Sym. Value	Sym. Name + Addend
0000000000000005	000500000000a	R_X86_64_32	0000000000000000	.rodata + 0
000000000000000a	000a000000002	R_X86_64_PC32	0000000000000000	puts - 4

Relocation section **'*.rela.eh_frame*'** at offset 0x200 contains 1 entries:

Offset	Info	Type	Sym. Value	Sym. Name + Addend
0000000000000020	0002000000002	R_X86_64_PC32	0000000000000000	.text + 0

```
[root@localhost tmp]# xxd -s 0x1d0 -l 0x30 HelloWorld.o
```

```
00001d0: 0500 0000 0000 0000 0a00 0000 0500 0000  ....
00001e0: 0000 0000 0000 0000 0a00 0000 0000 0000  ....
00001f0: 0200 0000 0a00 0000 fcff ffff ffff ffff  ....
```

```
[root@localhost tmp]# readelf -S HelloWorld
```

```
[ 9] .rela.dyn            RELA             0000000000400380 00000380
      0000000000000018 0000000000000018 A        5      0      8
[10] .rela.plt            RELA             0000000000400398 00000398
      0000000000000048 0000000000000018 AI       5     22      8
```

```
[root@localhost tmp]# readelf -r HelloWorld
```

Relocation section **'*.rela.dyn*'** at offset 0x380 contains 1 entries:

Offset	Info	Type	Sym. Value	Sym. Name + Addend
000000600ff8	000300000006	R_X86_64_GLOB_DAT	0000000000000000	__gmon_start__ + 0

Relocation section **'`.rela.plt`'** at offset 0x398 contains 3 entries:

Offset	Info	Type	Sym. Value	Sym. Name + Addend
000000601018	000100000007	R_X86_64_JUMP_SLO	0000000000000000	puts@GLIBC_2.2.5 + 0
000000601020	000200000007	R_X86_64_JUMP_SLO	0000000000000000	__libc_start_main@GLIBC_2.2.5 + 0
000000601028	000300000007	R_X86_64_JUMP_SLO	0000000000000000	__gmon_start__ + 0

```
[root@localhost tmp]# xxd -s 0x398 -l 0x48 HelloWorld
```

```
0000398: 1810 6000 0000 0000 0700 0000 0100 0000  ..`.....
00003a8: 0000 0000 0000 0000 2010 6000 0000 0000  ....`.....
00003b8: 0700 0000 0200 0000 0000 0000 0000 0000  ....
00003c8: 2810 6000 0000 0000 0700 0000 0300 0000  (.`.....
00003d8: 0000 0000 0000 0000  ....
```

6. gcc链接: collect2:ld//lib64/ld-linux-x86-64.so.2

动态链接

实验:

```
[root@localhost 3]# cat Program1.c
```

```
#include "Lib.h"

int main(void) {
    foobar(1);
    return 0;
}
```

```
[root@localhost 3]# cat Program2.c
```

```
#include "Lib.h"

int main(void) {
    foobar(2);
    return 0;
}
```

```
[root@localhost 3]# cat Lib.c
```

```
#include <stdio.h>

void foobar(int i) {
    printf("Printing from Lib.so %d\n", i);
}
```

```
[root@localhost 3]# cat Lib.h
```

```

#ifndef LIB_H
#define LIB_H

void foobar(int i);

#endif

```

```

[root@localhost 3]# gcc -fPIC -shared -o Lib.so Lib.c
[root@localhost 3]# gcc -o Program1 Program1.c ./Lib.so
[root@localhost 3]# gcc -o Program2 Program2.c ./Lib.so

```

7. elf装载

program headers

```

int main(void) {
    int a = 0x12345678;

    return 0;
}

```

```
[root@localhost tmp]# readelf -l test
```

Elf file type is EXEC (Executable file)

Entry point 0x400400

There are 9 program headers, starting at offset 64

Program Headers:

Type	Offset FileSiz	VirtAddr MemSiz	PhysAddr Flags Align
PHDR	0x0000000000000040	0x0000000000400040	0x0000000000400040
	0x00000000000001f8	0x00000000000001f8	R E 8
INTERP	0x0000000000000238	0x0000000000400238	0x0000000000400238
	0x000000000000001c	0x000000000000001c	R 1
[Requesting program interpreter: /lib64/ld-linux-x86-64.so.2]			
LOAD	0x0000000000000000	0x0000000000400000	0x0000000000400000
	0x000000000000069c	0x000000000000069c	R E 200000
LOAD	0x0000000000000e18	0x0000000000600e18	0x0000000000600e18
	0x0000000000000220	0x0000000000000228	RW 200000
DYNAMIC	0x0000000000000e28	0x0000000000600e28	0x0000000000600e28
	0x00000000000001d0	0x00000000000001d0	RW 8
NOTE	0x0000000000000254	0x0000000000400254	0x0000000000400254
	0x0000000000000044	0x0000000000000044	R 4
GNU_EH_FRAME	0x0000000000000574	0x0000000000400574	0x0000000000400574
	0x0000000000000034	0x0000000000000034	R 4
GNU_STACK	0x0000000000000000	0x0000000000000000	0x0000000000000000
	0x0000000000000000	0x0000000000000000	RW 10
GNU_RELRO	0x0000000000000e18	0x0000000000600e18	0x0000000000600e18
	0x00000000000001e8	0x00000000000001e8	R 1

Section to Segment mapping:

Segment Sections...

00

01 .interp

```

02      .interp .note.ABI-tag .note.gnu.build-id .gnu.hash .dynsym .dynstr
.gnu.version .gnu.version_r .rela.dyn .rela.plt .init .plt .text .fini .rodata
.eh_frame_hdr .eh_frame
03      .init_array .fini_array .dynamic .got .got.plt .data .bss
04      .dynamic
05      .note.ABI-tag .note.gnu.build-id
06      .eh_frame_hdr
07
08      .init_array .fini_array .dynamic .got

```

```

[root@localhost tmp]# xxd -s 0x40 -l 0x1f8 /tmp/test
0000040: 0600 0000 0500 0000 4000 0000 0000 0000  .....@.....
0000050: 4000 4000 0000 0000 4000 4000 0000 0000  @.@.....@.
0000060: f801 0000 0000 0000 f801 0000 0000 0000  .....
0000070: 0800 0000 0000 0000 0300 0000 0400 0000  .....
0000080: 3802 0000 0000 0000 3802 4000 0000 0000  8.....8.@.
0000090: 3802 4000 0000 0000 1c00 0000 0000 0000  8.@.....
00000a0: 1c00 0000 0000 0000 0100 0000 0000 0000  .....
00000b0: 0100 0000 0500 0000 0000 0000 0000 0000  .....
00000c0: 0000 4000 0000 0000 0000 4000 0000 0000  ..@.....@.
00000d0: 9c06 0000 0000 0000 9c06 0000 0000 0000  .....
00000e0: 0000 2000 0000 0000 0100 0000 0600 0000  .. .....
00000f0: 180e 0000 0000 0000 180e 6000 0000 0000  .....`.....
0000100: 180e 6000 0000 0000 2002 0000 0000 0000  ..`.....
0000110: 2802 0000 0000 0000 0000 2000 0000 0000  (.....
0000120: 0200 0000 0600 0000 280e 0000 0000 0000  .....(.....
0000130: 280e 6000 0000 0000 280e 6000 0000 0000  (.`.....(.`.....
0000140: d001 0000 0000 0000 d001 0000 0000 0000  .....
0000150: 0800 0000 0000 0000 0400 0000 0400 0000  .....
0000160: 5402 0000 0000 0000 5402 4000 0000 0000  T.....T.@.
0000170: 5402 4000 0000 0000 4400 0000 0000 0000  T.@.....D.....
0000180: 4400 0000 0000 0000 0400 0000 0000 0000  D.....
0000190: 50e5 7464 0400 0000 7405 0000 0000 0000  P.td....t.....
00001a0: 7405 4000 0000 0000 7405 4000 0000 0000  t.@.....t.@.
00001b0: 3400 0000 0000 0000 3400 0000 0000 0000  4.....4.....
00001c0: 0400 0000 0000 0000 51e5 7464 0600 0000  .....Q.td....
00001d0: 0000 0000 0000 0000 0000 0000 0000 0000  .....
00001e0: 0000 0000 0000 0000 0000 0000 0000 0000  .....
00001f0: 0000 0000 0000 0000 1000 0000 0000 0000  .....
0000200: 52e5 7464 0400 0000 180e 0000 0000 0000  R.td.....
0000210: 180e 6000 0000 0000 180e 6000 0000 0000  ..`.....`.....
0000220: e801 0000 0000 0000 e801 0000 0000 0000  .....
0000230: 0100 0000 0000 0000  .....

```

```

[root@localhost tmp]# xxd -s 0x40 -l 56 /tmp/test
0000040: 0600 0000 0500 0000 4000 0000 0000 0000  .....@.....
0000050: 4000 4000 0000 0000 4000 4000 0000 0000  @.@.....@.
0000060: f801 0000 0000 0000 f801 0000 0000 0000  .....
0000070: 0800 0000 0000 0000  .....

```

```
> ELF二进制文件load到内存并执行，内核源码：
> https://github.com/GabrielJiang-
J/study_information/blob/master/%E4%BA%8C%E8%BF%9B%E5%88%B6%E5%88%86%E6%9E%90/li
nux/elf-64-gen.pdf
> linux-2.6.34/fs/binfmt_elf.c:elf_format.load_binary
> linux-2.6.34/arch/ia64/kernel/process.c:sys_execve
```

8. elf执行