ELF1 1B ELF and Section Headers

Young W. Lim

2022-02-11 Fri

Outline

Based on

"Study of ELF loading and relocs", 1999
http://netwinder.osuosl.org/users/p/patb/public_html/elf_relocs.html

Compling 32-bit program on 64-bit gcc

- gcc -v
- gcc -m32 t.c
- sudo apt-get install gcc-multilib
- sudo apt-get install g++-multilib
- gcc-multilib
- g++-multilib
- gcc -m32
- objdump -m i386

(1) ELF header and program headers

- the ELF file has an header that describes
  the overall layout of the file.
- the ELF header actually points to
  another group of headers called the program headers
  - these headers describe to the operating system anything
    that might be required for it
    to load the binary into memory and execute it.
  - segments are described by program headers,
    but so are some other things required to get the executable running.

`https://www.bottomupcs.com/elf.xhtml`

(2) ELF32_Ehdr

ELF File Header

```
typedef struct {
        unsigned char e_ident[EI_NIDENT];
        Elf32_Half    e_type;
        Elf32_Half    e_machine;
        Elf32_Word    e_version;
        Elf32_Addr    e_entry;
        Elf32_Off     e_phoff;  ........ for program header table
        Elf32_Off     e_shoff;  ........ for section header table
        Elf32_Word    e_flags;
        Elf32_Half    e_ehsize;
        Elf32_Half    e_phentsize;  .... for program header table
        Elf32_Half    e_phnum;  ........ for program header table
        Elf32_Half    e_shentsize; ..... for section header table
        Elf32_Half    e_shnum;  ........ for section header table
        Elf32_Half    e_shstrndx;
} Elf32_Ehdr;
```

https://www.bottomupcs.com/elf.xhtml

(3) e_phoff, e_phentsize, e_phnum

- in the ELF (File) header definition

e_phoff	the offset in the file where the program header table starts
e_phentsize	the size of an entry of in the program header table
e_phnum	the number of entries in the program header table

- with these three fields, the file's program headers can be located and accessed

https://www.bottomupcs.com/elf.xhtml

(4) e_shoff, e_shentsize, e_shnum

- in the ELF (File) header definition

e_shoff	the offset in the file where the section header table starts
e_shentsize	the size of an entry of in the section header table
e_shnum	the number of entries in the section header table

- with these three fields, the file's program headers can be located and accessed

https://www.bottomupcs.com/elf.xhtml

(1) Elf32_Shdr

Section Header

```
typedef struct {
        Elf32_Word        sh_name;
        Elf32_Word        sh_type;
        Elf32_Word        sh_flags;
        Elf32_Addr        sh_addr;
        Elf32_Off         sh_offset;
        Elf32_Word        sh_size;
        Elf32_Word        sh_link;
        Elf32_Word        sh_info;
        Elf32_Word        sh_addralign;
        Elf32_Word        sh_entsize;
} Elf32_Shdr;
```

https://www.bottomupcs.com/elf.xhtml

(2) Elf32_Shdr v.s. Elf32_Phdr

Section Header

```
typedef struct {
        Elf32_Word      sh_name;
        Elf32_Word      sh_type;
        Elf32_Word      sh_flags;
        Elf32_Addr      sh_addr;
        Elf32_Off       sh_offset;
        Elf32_Word      sh_size;
        Elf32_Word      sh_link;
        Elf32_Word      sh_info;
        Elf32_Word      sh_addralign;
        Elf32_Word      sh_entsize;
} Elf32_Shdr;
```

Program Header

```
typedef struct {
        Elf32_Word      p_type;
        Elf32_Off       p_offset;
        Elf32_Addr      p_vaddr;
        Elf32_Addr      p_paddr;
        Elf32_Word      p_filesz;
        Elf32_Word      p_memsz;
        Elf32_Word      p_flags;
        Elf32_Word      p_align;
} ELF32_Phdr;
```

https://www.bottomupcs.com/elf.xhtml

(3) Elf32_Shdr corresponding fields

Section Header

```
typedef struct {
        Elf32_Word    sh_name;
        Elf32_Word    sh_type;          ... p_type
        Elf32_Word    sh_flags;         ... p_flags
        Elf32_Addr    sh_addr;          ... p_vaddr, p_paddr
        Elf32_Off     sh_offset;        ... p_offset
        Elf32_Word    sh_size;          ... p_filesz, p_memsz
        Elf32_Word    sh_link;
        Elf32_Word    sh_info;
        Elf32_Word    sh_addralign;     ... p_align
        Elf32_Word    sh_entsize;
} Elf32_Shdr;
```

https://www.bottomupcs.com/elf.xhtml

(4) Section header

- The section header table has all of the information necessary to locate and isolate each of the file's sections
- each entry in the section header table is a section header
  - a section header contains information characterizing the contents of the corresponding section if the file has such a section.
  - a section header is a structure of fixed size and format, consisting of the `Elf32_Shdr` fields, or members

`https://docs.oracle.com/cd/E19455-01/806-3773/elf-2/index.html`

(5) Section header table

- the section header table :
  an array of elements of `Elf32_Shdr` type
  - starts at `e_shoff` in the file
  - the table's total size : `e_shentsize * e_shnum`
  - each entry has the same size : `e_shentsize` (in bytes)
  - the number of entries : `e_shnum`

`https://man7.org/linux/man-pages/man5/elf.5.html`

(7) sh_type

- Program headers more than just segments.

sh_type	contents	shows what the entry is defining
SHT_PROGBITS	*program*	defines a program
SHT_NOBITS	*program*	defines no occupying space
SHT_DYNAMIC	*link info*	defines linking process
SHT_RELA	*link info*	defines a relocation entry with addens
SHT_REL	*link info*	defines a relocation entry without addends
SHT_SYMTAB	*link info*	defines a symbol table
SHT_SSTRTAB	*link info*	defines a string table

https://docs.oracle.com/cd/E19455-01/806-3773/elf-2/index.html

(7') sh_type

sh_type	Flags relevant to the segment
SHT_NULL	0x0
SHT_PROGBITS	0x1
SHT_SYMTAB	0x2
SHT_STRTAB	0x3
SHT_RELA	0x4
SHT_DYNAMIC	0x6
SHT_NOBITS	0x8
SHT_REL	0x9

- code segments should be marked as read and execute only,
- data sections as read and write with no execute.
- PF_MASKPROC mask are reserved for processor-specific semantics.

https://www.bottomupcs.com/elf.xhtml

(8) SHT_PROGBITS

sh_type	SHT_PROGBITS	defines a program

- SHT_PROGBITS type section contains information defined by the program, and in a *format* and with a *meaning* determined solely by the program.

https://docs.oracle.com/cd/E19455-01/806-3773/elf-2/index.html

(9) SHT_NOBITS

sh_type	SHT_NOBITS	defines a none occupying space

- SHT_NOBITS type section contains
  information defined by the program,
  and in a *format* and with a *meaning*
  determined by the program.
- SHT_NOBITS type section occupies
  no space in the file,
  but the section header offset field specifies
  the location at which the section would have begun
  if it did occupy space within the file.

https://docs.oracle.com/cd/E19455-01/806-3773/elf-2/index.html

(10) SHT_DYNAMIC (1)

sh_type	SHT_DYNAMIC	contains dynamic linking information

- SHT_DYNAMIC type section contains information for dynamic linking
  - contains the addresses / sizes for
    - procedure linkage table
    - global offset table
    - string table
    - relocation tables
    - symbol table
    - init / term function
- only one section of SHT_DYNAMIC type is allowed in a file

https://docs.oracle.com/cd/E19455-01/806-3773/elf-2/index.html

(11) SHT_DYNAMIC (2)

sh_type	SHT_DYNAMIC	contains dynamic linking information

- if an object file participates in dynamic linking,
  its program header table will have
  an element of type PT_DYNAMIC
- this segment contains the .dynamic section
  with the _DYNAMIC label

https://docs.oracle.com/cd/E19683-01/816-1386/6m7qcoblk/index.html#chapter6-42444

(12) SHT_RELA, SHT_REL

sh_type	SHT_RLEA	defines a relocation entry with addens
sh_type	SHT_RLE	defines a relocation entry without addens

- Identifies relocation entries
  with / without explicit addends
  such as type Elf32_Rela / Elf32_Rel
  for the 32–bit class of object files.
- An object file can have *multiple* relocation sections
- describes infomration about how to *modify* section contents
  for correct relocation

https://docs.oracle.com/cd/E19455-01/806-3773/elf-2/index.html
https://docs.oracle.com/cd/E19683-01/816-1386/6m7qcoblk/index.html#chapter6-42444

(13) SHT_SYMTAB

sh_type	SHT_SYMTAB	contains a complete symbol table

- a complete symbol table, usually for link editing.
- This table can also be used for dynamic linking;
- however, it can contain many unnecessary symbols.
- Only one section of this type is allowed in a file

https://docs.oracle.com/cd/E19683-01/816-1386/6m7qcoblk/index.html#chapter6-42444

(14) SHT_STRTAB

sh_type	SHT_STRTAB	contains a string table

- Is a string table. A file can have multiple string table sections.

https://docs.oracle.com/cd/E19683-01/816-1386/6m7qcoblk/index.html#chapter6-42444

(15) sh_size

- Specifies the size, in byte units, of the section.
- Even if the section type is SHT_NOBITS,
  sh_size can be nonzero (greater than 0) ;
  - however, the corresponding section still occupies no space in the file.

https://docs.oracle.com/cd/E19683-01/816-1386/6m7qcoblk/index.html#chapter6-42444

(16) sh_offset

- specifies the byte offset from the *beginning* of the *file* to the first byte in the section
- If the section type is SHT_NOBITS, the corresponding section occupies no space in the file.
  - In this case, sh_offset specifies the location at which the section would have begun if it did occupy space within the file.

https://docs.oracle.com/cd/E19683-01/816-1386/6m7qcoblk/index.html#chapter6-42444

(17) sh_addr

- Address where the first byte resides
  if the section appears in the memory image of a process;
- a value of 0 indicates the section does not appear
  in the memory image of a process.

https://docs.oracle.com/cd/E19683-01/816-1386/6m7qcoblk/index.html#chapter6-42444

(18) sh_flags (1)

sh_type	Flags relevant to the segment
SHF_OS_NONCONFORMING	0x100
SHF_GROUP	0x200
SHF_MSKOS	0x0ff00000
SHF_ORDERED	0x40000000
SHF_EXCLUDE	0x80000000
SHF_MASKPROC	0xf0000000

https://docs.oracle.com/cd/E19683-01/816-1386/6m7qcoblk/index.html#chapter6-42444

(19) sh_flags (2)

- SHF_WRITE
  Identifies a section that should be writable during process execution.
- SHF_ALLOC
  Identifies a section that occupies memory during process execution. Some control sections do not reside in the memory image of an object file. This attribute is off for those sections.
- SHF_EXECINSTR
  Identifies a section that contains executable machine instructions.

https://docs.oracle.com/cd/E19683-01/816-1386/6m7qcoblk/index.html#chapter6-42444

(20) sh_flags (3)

- SHF_MERGE
  - Identifies a section containing data that may be merged to eliminate duplication.
  - Unless the SHF_STRINGS flag is also set, the data elements in the section are of a uniform size.
  - The size of each element is specified in the section header's sh_entsize field.
  - If the SHF_STRINGS flag is also set, the data elements consist of null-terminated character strings.
  - The size of each character is specified in the section header's sh_entsize field.

https://docs.oracle.com/cd/E19683-01/816-1386/6m7qcoblk/index.html#chapter6-42444

(21) sh_flags (4)

- SHF_STRINGS
  Identifies a section that consists of null-terminated character strings. The size of each character is specified in the section header's sh_entsize field.
- SHF_INFO_LINK
  This section headers sh_info field holds a section header table index.

https://docs.oracle.com/cd/E19683-01/816-1386/6m7qcoblk/index.html#chapter6-42444

(22) sh_flags (5)

- SHF_LINK_ORDER (1)
  - this section adds special ordering requirements to the link-editor
  - the requirements apply if the sh_link field of this section's header references *another section*, the linked-to section.
  - if this section is combined with *other sections* in the output file, the section appears in the same relative order with respect to those sections.
  - Similarly the linked-to section appears with respect to sections the linked-to section is combined with.

https://docs.oracle.com/cd/E19683-01/816-1386/6m7qcoblk/index.html#chapter6-42444

(23) sh_flags (6)

- SHF_LINK_ORDER (2)
  - The special sh_link values SHN_BEFORE and SHN_AFTER imply that the sorted section is to precede or follow, respectively, all other sections in the set being ordered.
  - Input file link-line order is preserved if multiple sections in an ordered set have one of these special values.

https://docs.oracle.com/cd/E19683-01/816-1386/6m7qcoblk/index.html#chapter6-42444

(24) sh_flags (7)

- SHF_LINK_ORDER (3)
  - A typical use of this flag is to build a table
    that references text or data sections in address order.
  - In the absence of the sh_link ordering information,
    sections from a single input file combined
    within one section of the output file
    will be contiguous and have the same relative ordering
    as they did in the input file.
  - The contributions from multiple input files
    appear in link-line order.

https://docs.oracle.com/cd/E19683-01/816-1386/6m7qcoblk/index.html#chapter6-42444

(25) sh_link (1)

- sh_type = SHT_DYNAMIC
  - sh_link : the section header index of the associated string table
  - sh_info : 0
- sh_type = SHT_HASH
  - sh_link : the section header index of the associated symbol table
  - sh_info : 0

https://docs.oracle.com/cd/E19683-01/816-1386/6m7qcoblh/index.html

(26) sh_link (2)

- sh_type = SHT_REL / SHT_RELA
  - sh_link : the section header index of the associated symbol table
  - sh_info : the section header index of the section to which the relocation applies
- sh_type = SHT_SYMTAB / SHT_DYNSYM
  - sh_link : the section header index of the associated string table
  - sh_info : one greater than the symbol table index of the last local symbol (binding STB_LOCAL)

https://docs.oracle.com/cd/E19683-01/816-1386/6m7qcoblh/index.html

(26) sh_link (3)

- sh_type = SHT_GROUP
  - sh_link : the section header index of the associated symbol table
  - sh_info : the section header index of an entry in the associated symbol table. The name of the specified symbol table entry provides a signature for the section group

https://docs.oracle.com/cd/E19683-01/816-1386/6m7qcoblh/index.html

Section groups (1)

- Some sections occur in interrelated groups.
- For example,
  - an out-of-line definition of an inline function might require,
  - in addition to the section containing its executable instructions,
  - a read-only data section containing literals referenced,
  - one or more debugging information sections and
  - other informational sections.

`https://docs.oracle.com/cd/E19683-01/816-1386/6m7qcoblh/index.html`

Section groups (2)

- Furthermore, there may be internal references among these sections that would not make sense
  - if one of the sections were removed or
  - replaced by a duplicate from another object.
- Therefore, such groups must be included or omitted from the linked object as a unit.

`https://docs.oracle.com/cd/E19683-01/816-1386/6m7qcoblh/index.html`

Section groups (3)

- A section of type SHT_GROUP defines such a grouping of sections. The name of a symbol from one of the containing object's symbol tables provides a signature for the section group. The section header of the SHT_GROUP section specifies the identifying symbol entry. The sh_link member contains the section header index of the symbol table section that contains the entry. The sh_info member contains the symbol table index of the identifying entry. The sh_flags member of the section header contains 0. The name of the section (sh_name) is not specified.

https://docs.oracle.com/cd/E19683-01/816-1386/6m7qcoblh/index.html