Advanced Programming in the UNIX Environment

Week 11, Segment 1:
The Executable and Linkable Format (ELF)

Department of Computer Science
Stevens Institute of Technology

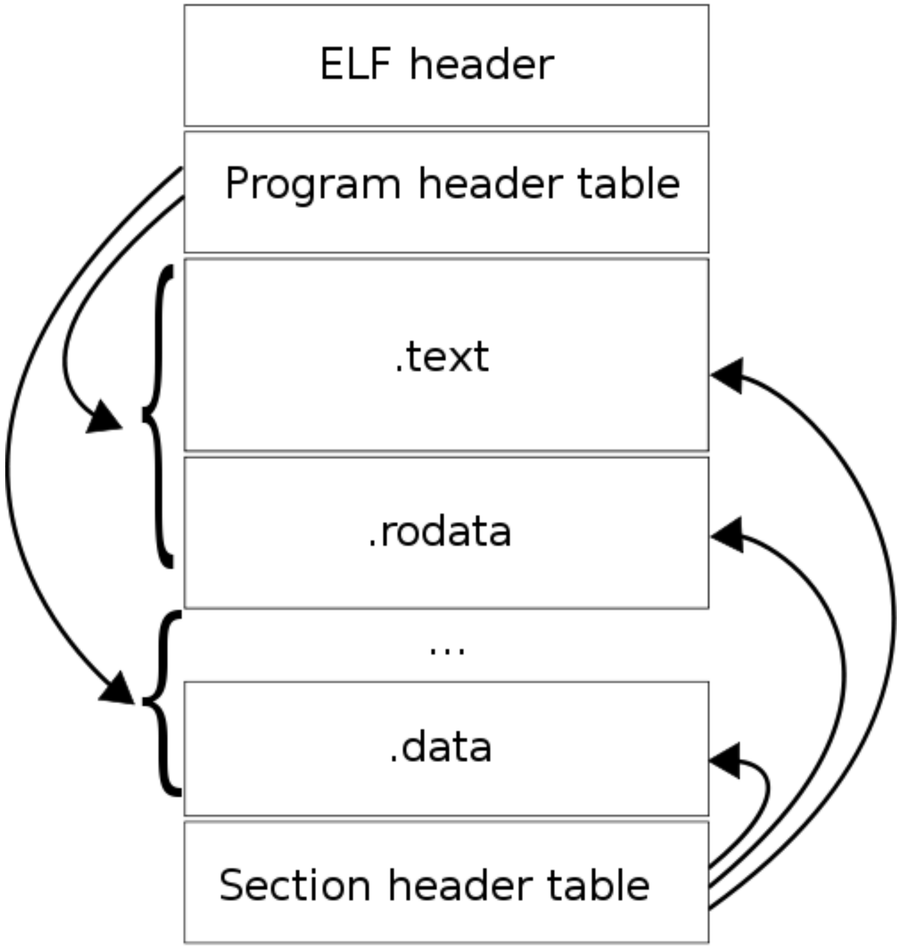
Jan Schaumann
jschauma@stevens.edu
https://stevens.netmeister.org/631/

The *E*xecutable and *L*inkable *F*ormat

Compilers produce, and linkers and loaders operate on *object files*. Just like other files, they have specific formats such as *e.g.*, assembler output (`a.out`), Common Object File Format (`COFF`), `Mach-O`, or `ELF`.

- executable – just what it sounds like (*e.g.*, `a.out`)
- core – virtual address space and register state of a process; debugging information (`a.out.core`)
- relocatable file – can be linked together with others to produce a shared library or an executable (*e.g.*, `foo.o`)
- shared object file – position independent code; used by the dynamic linker to create a process image (*e.g.*, `libfoo.so`)

The *E*xecutable and *L*inkable *F*ormat



The *E*xecutable and *L*inkable *F*ormat

Terminal — 80×24

```
typedef struct {
        unsigned char   e_ident[ELF_NIDENT];    /* Id bytes */
        Elf64_Half      e_type;                 /* file type */
        Elf64_Half      e_machine;              /* machine type */
        Elf64_Word      e_version;              /* version number */
        Elf64_Addr      e_entry;                /* entry point */
        Elf64_Off       e_phoff;                /* Program hdr offset */
        Elf64_Off       e_shoff;                /* Section hdr offset */
        Elf64_Word      e_flags;                /* Processor flags */
        Elf64_Half      e_ehsize;               /* sizeof ehdr */
        Elf64_Half      e_phentsize;            /* Program header entry size */
        Elf64_Half      e_phnum;                /* Number of program headers */
        Elf64_Half      e_shentsize;            /* Section header entry size */
        Elf64_Half      e_shnum;                /* Number of section headers */
        Elf64_Half      e_shstrndx;             /* String table index */
} Elf64_Ehdr;

/* e_ident offsets */
#define EI_MAG0         0       /* '\177' */
#define EI_MAG1         1       /* 'E'    */
#define EI_MAG2         2       /* 'L'    */
#define EI_MAG3         3       /* 'F'    */
#define EI_CLASS        4       /* File class */
                                                130,1     4385     9%
```

```
$ cc -Wall -Werror -Wextra -c main.c
$ file main.o
main.o: ELF 64-bit LSB relocatable, x86-64, version 1 (SYSV), not stripped
$
```

The filesystem tests are based on examining the return from a stat(2) system call. The program checks to see if the file is empty, or if it's some sort of special file. Any known file types appropriate to the system you are running on (sockets, symbolic links, or named pipes (FIFOs) on those systems that implement them) are intuited if they are defined in the system header file <sys/stat.h>.

The magic tests are used to check for files with data in particular fixed formats. The canonical example of this is a binary executable (compiled program) a.out file, whose format is defined in <elf.h>, <a.out.h> and possibly <exec.h> in the standard include directory. These files have a "magic number" stored in a particular place near the beginning of the file that tells the UNIX operating system that the file is a binary executable, and which of several types thereof. The concept of a "magic" has been applied by extension to data files. Any file with some invariant identifier at a small fixed offset into the file can usually be described in this way. The information identifying these files is read from the compiled magic file /usr/share/misc/magic.mgc, or the files in the directory /usr/share/misc/magic if the compiled file does not exist. In addition, if $HOME/.magic.mgc or $HOME/.magic exists, it will be used in preference to the system magic files.

--More--(14%)

Terminal — 80×24

MAGIC(5) File Formats Manual MAGIC(5)

NAME

magic - file command's magic pattern file

DESCRIPTION

This manual page documents the format of magic files as used by the file(1) command, version 5.37. The file(1) command identifies the type of a file using, among other tests, a test for whether the file contains certain "magic patterns". The database of these "magic patterns" is usually located in a binary file in /usr/share/misc/magic.mgc or a directory of source text magic pattern fragment files in /usr/share/misc/magic. The database specifies what patterns are to be tested for, what message or MIME type to print if a particular pattern is found, and additional information to extract from the file.

The format of the source fragment files that are used to build this database is as follows: Each line of a fragment file specifies a test to be performed. A test compares the data starting at a particular offset in the file with a byte value, a string or a numeric value. If the test succeeds, a message is printed. The line consists of the following fields:

--More--(4%)

NAME

magic_open, magic_close, magic_error, magic_errno, magic_descriptor, magic_file, magic_buffer, magic_getflags, magic_setflags, magic_check, magic_compile, magic_list, magic_load, magic_load_buffers, magic_setparam, magic_getparam, magic_version - Magic number recognition library

LIBRARY

Magic Number Recognition Library (libmagic, -lmagic)

SYNOPSIS

```
#include <magic.h>

magic_t
magic_open(int flags);

void
magic_close(magic_t cookie);

const char *
magic_error(magic_t cookie);
```

```
$ cc -Wall -Werror -Wextra -c main.c
$ file main.o
main.o: ELF 64-bit LSB relocatable, x86-64, version 1 (SYSV), not stripped
$ hexdump -C main.o | head -2
00000000  7f 45 4c 46 02 01 01 00  00 00 00 00 00 00 00 00  |.ELF............|
00000010  01 00 3e 00 01 00 00 00  00 00 00 00 00 00 00 00  |..>.............|
$
```

```
$ cc -Wall -Werror -Wextra -c main.c
$ file main.o
main.o: ELF 64-bit LSB relocatable, x86-64, version 1 (SYSV), not stripped
$ hexdump -C main.o | head -2
00000000  7f 45 4c 46 02 01 01 00  00 00 00 00 00 00 00 00  |.ELF............|
00000010  01 00 3e 00 01 00 00 00  00 00 00 00 00 00 00 00  |..>.............|
$
```

```
$ cc -Wall -Werror -Wextra -c main.c
$ file main.o
main.o: ELF 64-bit LSB relocatable, x86-64, version 1 (SYSV), not stripped
$ hexdump -C main.o | head -2
00000000  7f 45 4c 46 02 01 01 00  00 00 00 00 00 00 00 00  |.ELF............|
00000010  01 00 3e 00 01 00 00 00  00 00 00 00 00 00 00 00  |..>.............|
$
```

```
$ cc -Wall -Werror -Wextra -c main.c
$ file main.o
main.o: ELF 64-bit LSB relocatable, x86-64, version 1 (SYSV), not stripped
$ hexdump -C main.o | head -2
00000000  7f 45 4c 46 02 01 01 00  00 00 00 00 00 00 00 00  |.ELF............|
00000010  01 00 3e 00 01 00 00 00  00 00 00 00 00 00 00 00  |..>.............|
$
```

```
$ cc -Wall -Werror -Wextra -c main.c
$ file main.o
main.o: ELF 64-bit LSB relocatable, x86-64, version 1 (SYSV), not stripped
$ hexdump -C main.o | head -2
00000000  7f 45 4c 46 02 01 01 00  00 00 00 00 00 00 00 00  |.ELF............|
00000010  01 00 3e 00 01 00 00 00  00 00 00 00 00 00 00 00  |..>.............|
$
```

```
$ cc -Wall -Werror -Wextra -c main.c
$ file main.o
main.o: ELF 64-bit LSB relocatable, x86-64, version 1 (SYSV), not stripped
$ hexdump -C main.o | head -2
00000000  7f 45 4c 46 02 01 01 00  00 00 00 00 00 00 00 00  |.ELF............|
00000010  01 00 3e 00 01 00 00 00  00 00 00 00 00 00 00 00  |..>.............|
$
```

```
$ cc -Wall -Werror -Wextra -c main.c
$ file main.o
main.o: ELF 64-bit LSB relocatable, x86-64, version 1 (SYSV), not stripped
$ hexdump -C main.o | head -2
00000000  7f 45 4c 46 02 01 01 00  00 00 00 00 00 00 00 00  |.ELF............|
00000010  01 00 3e 00 01 00 00 00  00 00 00 00 00 00 00 00  |..>.............|
$
```

```
$ hexdump -C main.o | head -2
00000000  7f 45 4c 46 02 01 01 00  00 00 00 00 00 00 00 00  |.ELF............|
00000010  01 00 3e 00 01 00 00 00  00 00 00 00 00 00 00 00  |..>.............|
$ readelf -h main.o
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:                              REL (Relocatable file)
  Machine:                           Advanced Micro Devices X86-64
  Version:                           0x1
  Entry point address:               0x0
```

```
$ hexdump -C main.o | head -2
00000000  7f 45 4c 46 02 01 01 00  00 00 00 00 00 00 00 00  |.ELF............|
00000010  01 00 3e 00 01 00 00 00  00 00 00 00 00 00 00 00  |..>.............|
$ readelf -h main.o
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:                              REL (Relocatable file)
  Machine:                           Advanced Micro Devices X86-64
  Version:                           0x1
  Entry point address:               0x0
```

```
$ hexdump -C main.o | head -2
00000000  7f 45 4c 46 02 01 01 00  00 00 00 00 00 00 00 00  |.ELF............|
00000010  01 00 3e 00 01 00 00 00  00 00 00 00 00 00 00 00  |..>.............|
$ readelf -h main.o
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:                              REL (Relocatable file)
  Machine:                           Advanced Micro Devices X86-64
  Version:                           0x1
  Entry point address:               0x0
```

```
$ hexdump -C main.o | head -2
00000000  7f 45 4c 46 02 01 01 00  00 00 00 00 00 00 00 00  |.ELF............|
00000010  01 00 3e 00 01 00 00 00  00 00 00 00 00 00 00 00  |..>.............|
$ readelf -h main.o
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:                              REL (Relocatable file)
  Machine:                           Advanced Micro Devices X86-64
  Version:                           0x1
  Entry point address:               0x0
```

```
$ hexdump -C main.o | head -2
00000000  7f 45 4c 46 02 01 01 00  00 00 00 00 00 00 00 00  |.ELF............|
00000010  01 00 3e 00 01 00 00 00  00 00 00 00 00 00 00 00  |..>.............|
$ readelf -h main.o
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:                              REL (Relocatable file)
  Machine:                           Advanced Micro Devices X86-64
  Version:                           0x1
  Entry point address:               0x0
```

```
$ hexdump -C main.o | head -2
00000000  7f 45 4c 46 02 01 01 00  00 00 00 00 00 00 00 00  |.ELF............|
00000010  01 00 3e 00 01 00 00 00  00 00 00 00 00 00 00 00  |..>.............|
$ readelf -h main.o
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:                              REL (Relocatable file)
  Machine:                           Advanced Micro Devices X86-64
  Version:                           0x1
  Entry point address:               0x0
```

```
$ hexdump -C main.o | head -2
00000000  7f 45 4c 46 02 01 01 00  00 00 00 00 00 00 00 00  |.ELF............|
00000010  01 00 3e 00 01 00 00 00  00 00 00 00 00 00 00 00  |..>.............|
$ readelf -h main.o
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:                              REL (Relocatable file)
  Machine:                           Advanced Micro Devices X86-64
  Version:                           0x1
  Entry point address:               0x0
```

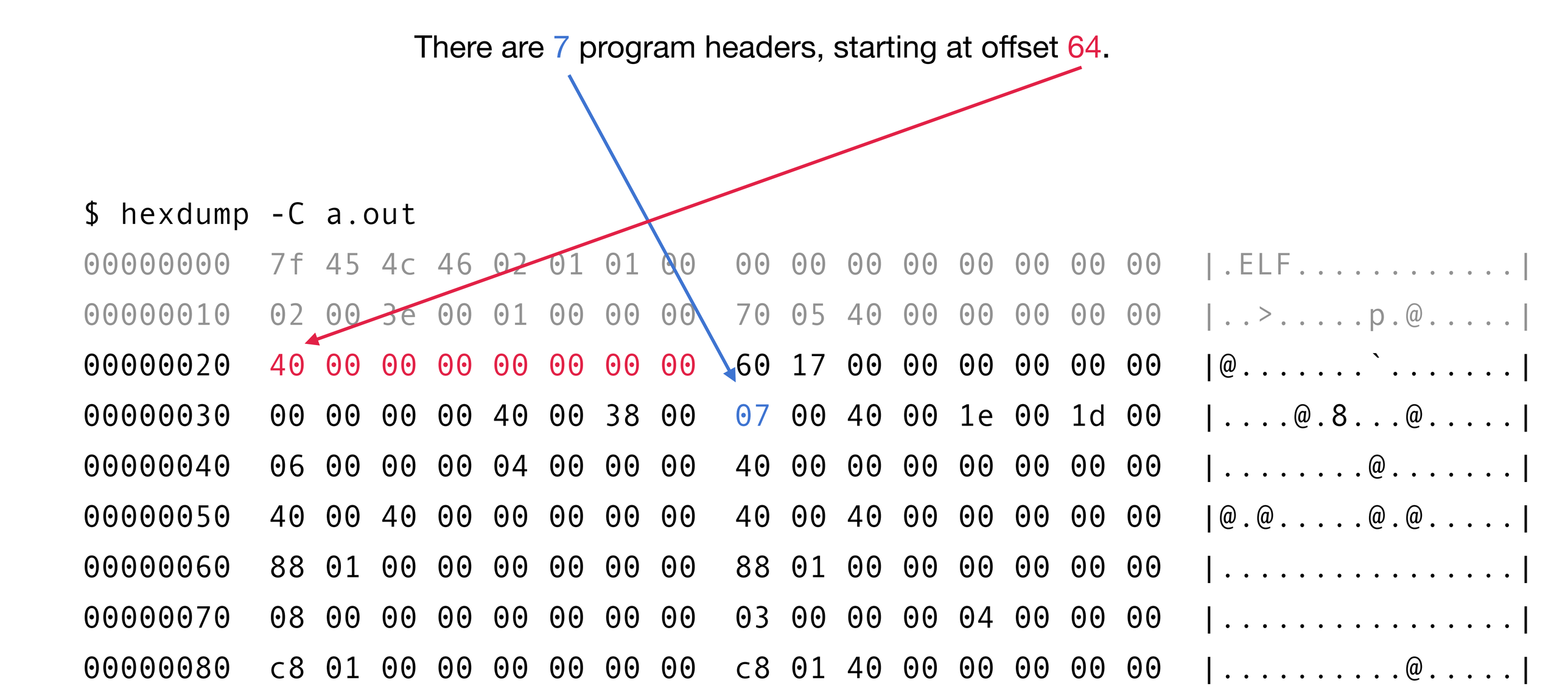
```
$ hexdump -C main.o | head -2
00000000  7f 45 4c 46 02 01 01 00  00 00 00 00 00 00 00 00  |.ELF............|
00000010  01 00 3e 00 01 00 00 00  00 00 00 00 00 00 00 00  |..>.............|
$ readelf -h main.o
ELF Header:
  Magic:   7f 45 4c 46 02 01 01 00 00 00 00 00 00 00 00 00
  Class:                             ELF64
  Data:                              2's complement, little endian
  Version:                           1 (current)
  OS/ABI:                            UNIX - System V
  ABI Version:                       0
  Type:                              REL (Relocatable file)
  Machine:                           Advanced Micro Devices X86-64
  Version:                           0x1
  Entry point address:               0x0
```

```
$ hexdump -C main.o | head -2
00000000  7f 45 4c 46 02 01 01 00  00 00 00 00 00 00 00 00  |.ELF............|
00000010  01 00 3e 00 01 00 00 00  00 00 00 00 00 00 00 00  |..>.............|
$ readelf -h main.o
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:                              REL (Relocatable file)
  Machine:                           Advanced Micro Devices X86-64
  Version:                           0x1
  Entry point address:               0x0
```

```
$ hexdump -C main.o | head -2
00000000  7f 45 4c 46 02 01 01 00  00 00 00 00 00 00 00 00  |.ELF............|
00000010  01 00 3e 00 01 00 00 00  00 00 00 00 00 00 00 00  |..>.............|
$ readelf -h main.o
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:                              REL (Relocatable file)
  Machine:                           Advanced Micro Devices X86-64
  Version:                           0x1
  Entry point address:               0x0
```

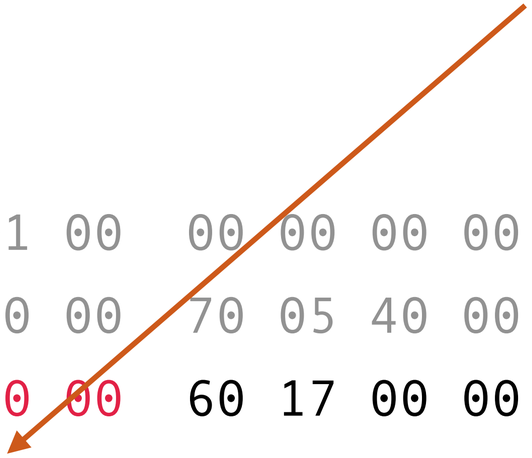
```
$ cc main.o
$ hexdump -C a.out | head -2
00000000  7f 45 4c 46 02 01 01 00  00 00 00 00 00 00 00 00  |.ELF............|
00000010  02 00 3e 00 01 00 00 00  70 05 40 00 00 00 00 00  |..>.....p.@.....|
$ readelf -h a.out
ELF Header:
  Magic:   7f 45 4c 46 02 01 01 00 00 00 00 00 00 00 00 00
  Class:                             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:               0x400570
```

```
$ hexdump -C a.out
00000000  7f 45 4c 46 02 01 01 00  00 00 00 00 00 00 00 00  |.ELF............|
00000010  02 00 3e 00 01 00 00 00  70 05 40 00 00 00 00 00  |..>.....p.@.....|
00000020  40 00 00 00 00 00 00 00  60 17 00 00 00 00 00 00  |@.......`.......|
00000030  00 00 00 00 40 00 38 00  07 00 40 00 1e 00 1d 00  |....@.8...@.....|
00000040  06 00 00 00 04 00 00 00  40 00 00 00 00 00 00 00  |........@.......|
00000050  40 00 40 00 00 00 00 00  40 00 40 00 00 00 00 00  |@.@.....@.@.....|
00000060  88 01 00 00 00 00 00 00  88 01 00 00 00 00 00 00  |................|
00000070  08 00 00 00 00 00 00 00  03 00 00 00 04 00 00 00  |................|
00000080  c8 01 00 00 00 00 00 00  c8 01 40 00 00 00 00 00  |..........@.....|
```


There are 7 program headers, starting at offset 64.
$ hexdump -C a.out
00000000 7f 45 4c 46 02 01 01 00 00 00 00 00 00 00 00 00 |.ELF............|
00000010 02 00 3e 00 01 00 00 00 70 05 40 00 00 00 00 00 |..>.....p.@.....|
00000020 40 00 00 00 00 00 00 00 60 17 00 00 00 00 00 00 |@.......`.......|
00000030 00 00 00 00 40 00 38 00 07 00 40 00 1e 00 1d 00 |....@.8...@.....|
00000040 06 00 00 00 04 00 00 00 40 00 00 00 00 00 00 00 |........@.......|
00000050 40 00 40 00 00 00 00 00 40 00 40 00 00 00 00 00 |@.@.....@.@.....|
00000060 88 01 00 00 00 00 00 00 88 01 00 00 00 00 00 00 |................|
00000070 08 00 00 00 00 00 00 00 03 00 00 00 04 00 00 00 |................|
00000080 c8 01 00 00 00 00 00 00 c8 01 40 00 00 00 00 00 |..........@.....|

There are 7 program headers, starting at offset 64.

The size of the ELF header is 0x38 = 56.



```
$ hexdump -C a.out
00000000  7f 45 4c 46 02 01 01 00  00 00 00 00 00 00 00 00  |.ELF............|
00000010  02 00 3e 00 01 00 00 00  70 05 40 00 00 00 00 00  |..>.....p.@.....|
00000020  40 00 00 00 00 00 00 00  60 17 00 00 00 00 00 00  |@.......`.......|
00000030  00 00 00 00 40 00 38 00  07 00 40 00 1e 00 1d 00  |....@.8...@.....|
00000040  06 00 00 00 04 00 00 00  40 00 00 00 00 00 00 00  |........@.......|
00000050  40 00 40 00 00 00 00 00  40 00 40 00 00 00 00 00  |@.@.....@.@.....|
00000060  88 01 00 00 00 00 00 00  88 01 00 00 00 00 00 00  |................|
00000070  08 00 00 00 00 00 00 00  03 00 00 00 04 00 00 00  |................|
00000080  c8 01 00 00 00 00 00 00  c8 01 40 00 00 00 00 00  |..........@.....|
```

There are 7 program headers, starting at offset 64.

The size of the ELF header is 0x38 = 56.

Program Header Table (PHDR) with segment permissions read at offset 0x40 = 64

```
$ hexdump -C a.out
00000000  7f 45 4c 46 02 01 01 00  00 00 00 00 00 00 00 00  |.ELF............|
00000010  02 00 3e 00 01 00 00 00  70 05 40 00 00 00 00 00  |..>.....p.@.....|
00000020  40 00 00 00 00 00 00 00  60 17 00 00 00 00 00 00  |@.......`.......|
00000030  00 00 00 00 40 00 38 00  07 00 40 00 1e 00 1d 00  |....@.8...@.....|
00000040  06 00 00 00 04 00 00 00  40 00 00 00 00 00 00 00  |........@.......|
[...]
00000070  08 00 00 00 00 00 00 00  03 00 00 00 04 00 00 00  |................|
00000080  c8 01 00 00 00 00 00 00  c8 01 40 00 00 00 00 00  |..........@.....|
```

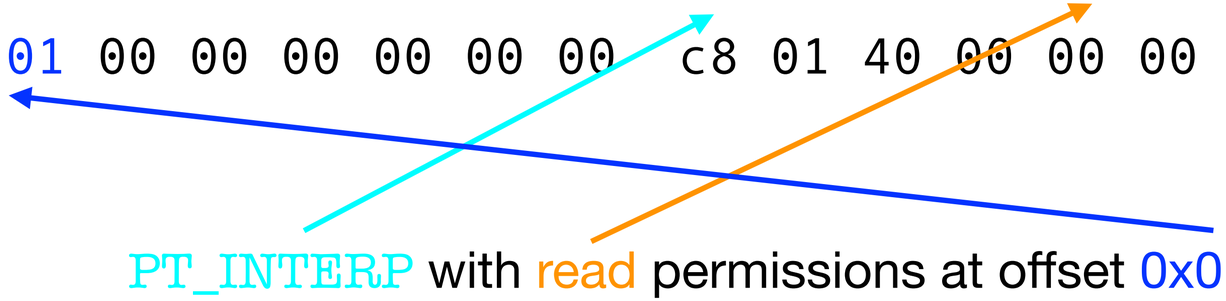
Next header at offset 0x38 + 0x40 = 0x78

There are 7 program headers, starting at offset 64.

The size of the ELF header is 0x38 = 56.

Program Header Table (PHDR) with segment permissions read at offset 0x40 = 64

```
$ hexdump -C a.out
00000000  7f 45 4c 46 02 01 01 00  00 00 00 00 00 00 00 00  |.ELF............|
00000010  02 00 3e 00 01 00 00 00  70 05 40 00 00 00 00 00  |..>.....p.@.....|
00000020  40 00 00 00 00 00 00 00  60 17 00 00 00 00 00 00  |@.......`.......|
00000030  00 00 00 00 40 00 38 00  07 00 40 00 1e 00 1d 00  |....@.8...@.....|
00000040  06 00 00 00 04 00 00 00  40 00 00 00 00 00 00 00  |........@.......|
[...]
00000070  08 00 00 00 00 00 00 00  03 00 00 00 04 00 00 00  |................|
00000080  c8 01 00 00 00 00 00 00  c8 01 40 00 00 00 00 00  |..........@.....|
```



PT_INTERP with read permissions at offset 0x01c8.

There are 7 program headers, starting at offset 64.

The size of the ELF header is 0x38 = 56.

Program Header Table (PHDR) with segment permissions read at offset 0x40 = 64

```
$ hexdump -C a.out
00000000  7f 45 4c 46 02 01 01 00  00 00 00 00 00 00 00 00  |.ELF............|
00000010  02 00 3e 00 01 00 00 00  70 05 40 00 00 00 00 00  |..>.....p.@.....|
00000020  40 00 00 00 00 00 00 00  60 17 00 00 00 00 00 00  |@.......`.......|
00000030  00 00 00 00 40 00 38 00  07 00 40 00 1e 00 1d 00  |....@.8...@.....|
00000040  06 00 00 00 04 00 00 00  40 00 00 00 00 00 00 00  |........@.......|
[...]
00000070  08 00 00 00 00 00 00 00  03 00 00 00 04 00 00 00  |................|
00000080  c8 01 00 00 00 00 00 00  c8 01 40 00 00 00 00 00  |..........@.....|
[...]
000001c0  04 00 00 00 00 00 00 00  2f 75 73 72 2f 6c 69 62  |......../usr/lib|
000001d0  65 78 65 63 2f 6c 64 2e  65 6c 66 5f 73 6f 00 00  |exec/ld.elf_so..|
```

```
$ readelf -l a.out

Elf file type is EXEC (Executable file)
Entry point 0x400570
There are 7 program headers, starting at offset 64

Program Headers:
  Type           Offset             VirtAddr           PhysAddr
                 FileSiz            MemSiz              Flags  Align
  PHDR           0x0000000000000040 0x0000000000400040 0x0000000000400040
                 0x0000000000000188 0x0000000000000188  R      0x8
  INTERP         0x00000000000001c8 0x00000000004001c8 0x00000000004001c8
                 0x0000000000000017 0x0000000000000017  R      0x1
      [Requesting program interpreter: /usr/libexec/ld.elf_so]
```

Exercises

- Repeat the `hexdump(1)`/`readelf(1)` analysis for a core dump (*e.g.*, `a.out.core`) and a shared library (*e.g.*, `/lib/libc.so`). What fields are different?
- Compile a program into an i386 binary and compare the `readelf(1)` output.
- Revisit Week 06, Segment 2: compile a program with an alternate entry function and see how the ELF header changes.
- Compare the use of the `readelf(1)` and `objdump(1)` utilities to analyze ELF files.
- Read `elf(3)` and explore `/usr/include/elf.h` to view supported ELF header fields.

Links

- https://stevens.netmeister.org/631/elf.html
- https://www.thegeekstuff.com/2012/07/elf-object-file-format/
- https://is.gd/v8eVFI
- https://jvns.ca/blog/2014/09/06/how-to-read-an-executable/
- http://blog.k3170makan.com/2018/09/introduction-to-elf-format-elf-header.html