Lecture 4:
ELF, Compiler, Linker, and Loader

Sanchuan Chen

schen@auburn.edu

8/31/2023

Executable and Linkable Format

- Standard binary format for object files
- Derived from AT&T System V Unix, now supported by UNIX/Linux
- A unified format for
  - **Relocatable object files (.o)**: Created by compilers or assemblers. Need to be processed by the linker before running.
  - **Executable object files**: Have all relocation done and all symbol resolved except perhaps shared library symbols that must be resolved at run time.
  - **Shared object files (.so)**: Shared library containing both symbol information for the linker and directly runnable code for run time

More in the specification: https://refspecs.linuxfoundation.org/elf/elf.pdf

Example (hexdump -C a.out && readelf -e a.out)

```
00000000  7f 45 4c 46 01 01 01 00  00 00 00 00 00 00 00 00
00000010  02 00 03 00 01 00 00 00  80 9a 04 08 34 00 00 00   ELF Header
00000020  78 64 01 00 00 00 00 00  34 00 20 00 08 00 28 00
00000030  1c 00 1b 00 06 00 00 00  34 00 00 00 34 80 04 08
00000040  34 80 04 08 00 01 00 00  00 01 00 00 05 00 00 00
00000050  04 00 00 00 03 00 00 00  34 01 00 00 34 81 04 08   Program Header
00000060  34 81 04 08 13 00 00 00  13 00 00 00 04 00 00 00
                        ...
00000130  04 00 00 00 2f 6c 69 62  2f 6c 64 2d 6c 69 6e 75   sec[1].interp
00000140  78 2e 73 6f 2e 32 00 00  04 00 00 00 10 00 00 00   sec[2].note.ABI-tag
00000150  01 00 00 00 47 4e 55 00  00 00 00 00 02 00 00 00
00000160  06 00 00 00 08 00 00 00  61 00 00 00 68 00 00 00   ...
                        ...
000014b0  ff 35 08 e1 05 08 ff 25  0c e1 05 08 00 00 00 00
000014c0  ff 25 10 e1 05 08 68 00  00 00 00 e9 e0 ff ff ff   sec[12].plt
                        ...
00001a70  ff 25 7c e2 05 08 68 d8  02 00 00 e9 30 fa ff ff
00001a80  31 ed 5e 89 e1 83 e4 f0  50 54 52 68 60 9e 05 08   sec[13].text
                        ...                                  ...
00016390  6c 73 00 00 52 a8 bd 5a  00 2e 73 68 73 74 72 74
000163a0  61 62 00 2e 69 6e 74 65  72 70 00 2e 6e 6f 74 65   sec[27].shstrtab
                        ...
00016470  75 67 6c 69 6e 6b 00 00  00 00 00 00 00 00 00 00
00016480  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
                        ...
000164a0  0b 00 00 00 01 00 00 00  02 00 00 00 34 81 04 08
000164b0  34 01 00 00 13 00 00 00  00 00 00 00 00 00 00 00   Section Header
                        ...
000168c0  98 63 01 00 df 00 00 00  00 00 00 00 00 00 00 00
000168d0  01 00 00 00 00 00 00 00
```

```
                    Section Headers:
[Nr] Name              Type     Addr     Off    Size   ES Flg Lk Inf Al
[ 0]                   NULL     00000000 000000 000000 00      0   0  0
[ 1] .interp           PROGBITS 08048134 000134 000013 00   A  0   0  1
[ 2] .note.ABI-tag     NOTE     08048148 000148 000020 00   A  0   0  4
                                  ...
[12] .plt              PROGBITS 080494b0 0014b0 0005d0 04  AX  0   0  4
[13] .text             PROGBITS 08049a80 001a80 0104bc 00  AX  0   0 16
[14] .fini             PROGBITS 08059f3c 011f3c 00001c 00  AX  0   0  4
[15] .rodata           PROGBITS 08059f60 011f60 003e4c 00   A  0   0 32
                                  ...
[27] .shstrtab         STRTAB   00000000 016398 0000df 00      0   0  1
```

ELF header

- Magic number, type (.o, exec, .so), machine, byte ordering, etc.

```
typedef struct elf32_hdr{
    unsigned char e_ident[EI_NIDENT];
    Elf32_Half e_type; /* ET_EXEC ET_DYN */
    Elf32_Half e_machine;
    Elf32_Word e_version;
    Elf32_Addr e_entry; /* Entry point */
    Elf32_Off e_phoff;
    Elf32_Off e_shoff;
    Elf32_Word e_flags;
    Elf32_Half e_ehsize;
    Elf32_Half e_phentsize;
    Elf32_Half e_phnum;
    Elf32_Half e_shentsize;
    Elf32_Half e_shnum;
    Elf32_Half e_shstrndx;
} Elf32_Ehdr;
```

.interp

- An ELF interpreter, normally /lib/ld-linux.so.2

Program header table

- page size, virtual addresses for memory segments (sections), segment sizes.

.text section

- code

.data section

- initialized (static) data

.bss section

- uninitialized (static) data
- "Block Started by Symbol" / "Better Save Space"
- has section header but occupies no space

.symtab section

- symbol table
- procedure and static variable names
- section names

.rel.text section

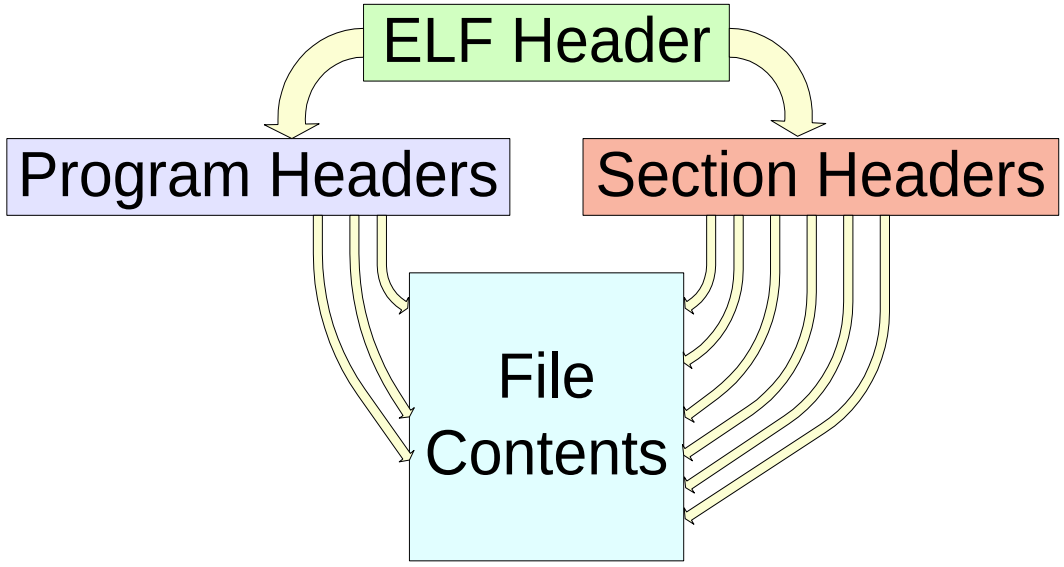
- relocation info for .text section
- addresses of instructions that will need to be modified in the executable
- instructions for modifying.

.rel.data section

- relocation info for .data section
- addresses of pointer data that will need to be modified in the merged executable

.debug section

- info for symbolic debugging (gcc -g)

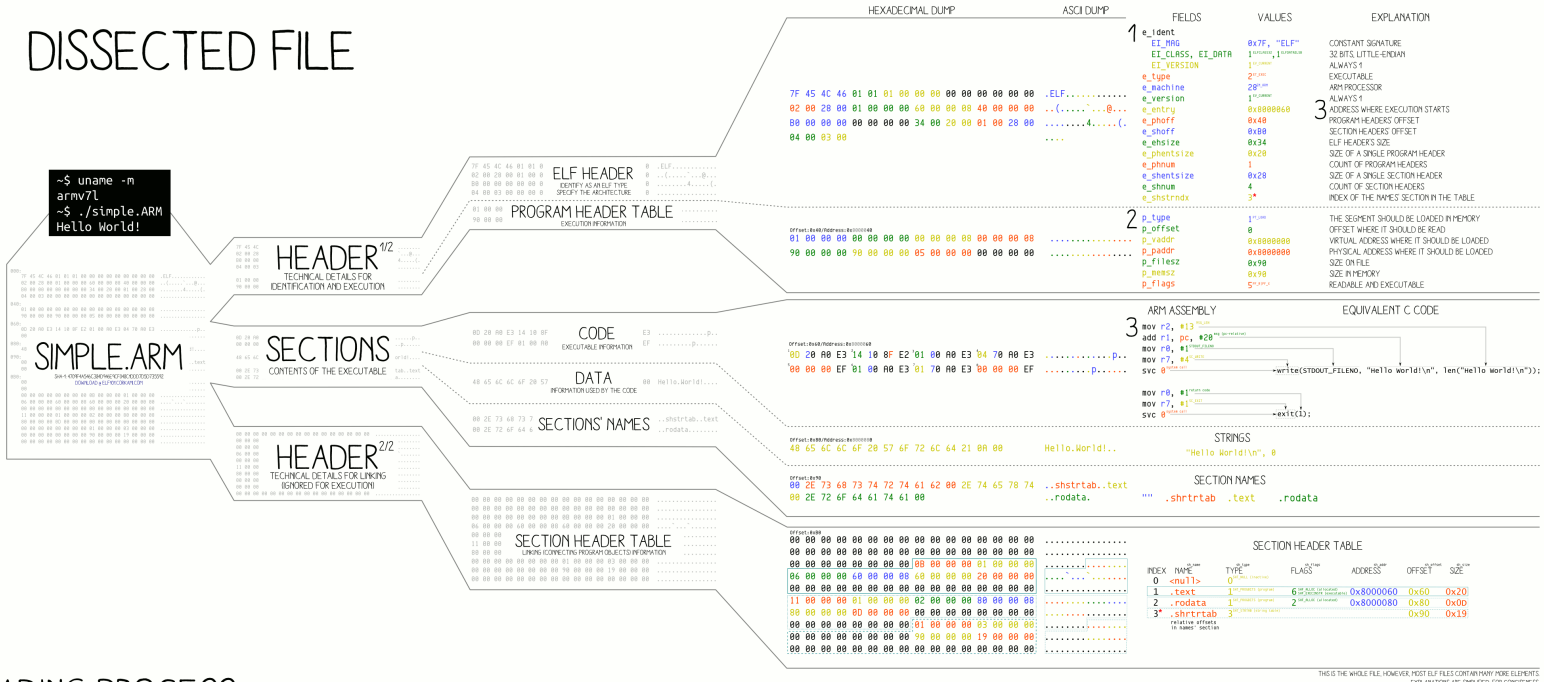


- A single segment usually consist of several sections.
  - E.g., a loadable read-only segment could contain sections for executable code, read-only data, and symbols for the dynamic linker.
- Sections are intended for further processing by the **linker**, while the segments are intended to be mapped into memory by the **loader**.

ELF101 a Linux executable walk-through

ANGE ALBERTINI
CORKAMI.COM

DISSECTED FILE



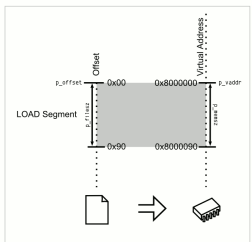
LOADING PROCESS

1 HEADER

THE ELF HEADER IS PARSED
THE PROGRAM HEADER IS PARSED
(SECTIONS ARE NOT USED)

2 MAPPING

THE FILE IS MAPPED IN MEMORY
ACCORDING TO ITS SEGMENT(S)



3 EXECUTION

ENTRY IS CALLED
SYSCALLS ARE ACCESSED VIA:
- SYSCALL NUMBER IN THE R7 REGISTER
- CALLING INSTRUCTION SVC

TRIVIA

THE ELF WAS FIRST SPECIFIED BY U.S.L. AND U.I.
FOR UNIX SYSTEM V, IN 1989

THE ELF IS USED, AMONG OTHERS, IN:
- LINUX, ANDROID, *BSD, SOLARIS, BEOS
- PSP, PLAYSTATION 2-4, DREAMCAST, GAMECUBE, WII
- VARIOUS OSES MADE BY SAMSUNG, ERICSSON, NOKIA,
- MICROCONTROLLERS FROM ATMEL, TEXAS INSTRUMENTS

VERSION 1.0A
2013/12/06

Why Compiler?

- Compiler parses the source code
- Compiler knows exactly the behavior of the program
- Compiler has to know the machine details (when generating the code)
- Compiler-based security solutions, e.g., **vulnerability detection**
  - Stack overflow
  - Integer overflow
  - Heap overflow
  - format string
  - Double free
  - Use-after-free

GNU Compiler Collection (GCC)

- **Developer(s)** GNU Project
- **Initial release** May 23, 1987
- **Newest version** GCC 13.2
- **Written** in C, C++
- **Operating system** Cross-platform
- **Platform** GNU
- **Type** Compiler
- **License** GNU General Public License (version 3 or later)
- **Website** gcc.gnu.org

Compilation system includes the phases

- Preprocessor
- Compiler
  - e.g., Optimizer
- Assembler
- Linker

Compiler Driver coordinates these phases

```
GCC(1)

NAME
       gcc - GNU project C and C++ compiler

SYNOPSIS
       gcc [-c|-S|-E] [-std=standard]
           [-g] [-pg] [-Olevel]
           [-Wwarn...] [-Wpedantic]
           [-Idir...] [-Ldir...]
           [-Dmacro[=defn]...] [-Umacro]
           [-foption...] [-mmachine-option...]
           [-o outfile] [@file] infile...

DESCRIPTION
       When you invoke GCC, it normally does preprocessing,
       compilation, assembly and linking.
```

Compiler Driver command example

```
schen@linux:~/comp6700/lec04$ gcc -m32 -fno-pic helloworld.c -v
Using built-in specs.
COLLECT_GCC=gcc
COLLECT_LTO_WRAPPER=/usr/lib/gcc/x86_64-linux-gnu/9/lto-wrapper
OFFLOAD_TARGET_NAMES=nvptx-none:hsa
OFFLOAD_TARGET_DEFAULT=1
Target: x86_64-linux-gnu
Configured with: ../src/configure -v --with-pkgversion='Ubuntu 9.3.0-17ubuntu1~20.04'
--with-bugurl=file:///usr/share/doc/gcc-9/README.Bugs
--enable-languages=c,ada,c++,go,brig,d,fortran,objc,obj-c++,gm2 --prefix=/usr
--with-gcc-major-version-only --program-suffix=-9 --program-prefix=x86_64-linux-gnu-
--enable-shared --enable-linker-build-id --libexecdir=/usr/lib
--without-included-gettext --enable-threads=posix --libdir=/usr/lib --enable-nls
--enable-clocale=gnu --enable-libstdcxx-debug --enable-libstdcxx-time=yes
--with-default-libstdcxx-abi=new --enable-gnu-unique-object --disable-vtable-verify
--enable-plugin --enable-default-pie --with-system-zlib
--with-target-system-zlib=auto --enable-objc-gc=auto --enable-multiarch
--disable-werror --with-arch-32=i686 --with-abi=m64
--with-multilib-list=m32,m64,mx32 --enable-multilib --with-tune=generic
--enable-offload-targets=nvptx-none=/build/gcc-9-HskZEa/gcc-9-9.3.0/debian/tmp-nvptx
/usr,hsa --without-cuda-driver --enable-checking=release --build=x86_64-linux-gnu
--host=x86_64-linux-gnu --target=x86_64-linux-gnu
Thread model: posix
gcc version 9.3.0 (Ubuntu 9.3.0-17ubuntu1~20.04)
COLLECT_GCC_OPTIONS='-m32' '-fno-pic' '-v' '-mtune=generic' '-march=i686'
 /usr/lib/gcc/x86_64-linux-gnu/9/cc1 -quiet -v -imultilib 32 -imultiarch
 i386-linux-gnu helloworld.c -quiet -dumpbase helloworld.c -m32 -mtune=generic
 -march=i686 -auxbase helloworld -version -fno-pic -fasynchronous-unwind-tables
 -fstack-protector-strong -Wformat -Wformat-security -fstack-clash-protection
 -fcf-protection -o /tmp/ccWCdCj6.s
GNU C17 (Ubuntu 9.3.0-17ubuntu1~20.04) version 9.3.0 (x86_64-linux-gnu)
   compiled by GNU C version 9.3.0, GMP version 6.2.0, MPFR version 4.0.2, MPC
   version 1.1.0, isl version isl-0.22.1-GMP
```

Compiler Driver command example (continued)

```
GGC heuristics: --param ggc-min-expand=100 --param ggc-min-heapsize=131072
ignoring nonexistent directory "/usr/local/include/i386-linux-gnu"
ignoring nonexistent directory "/usr/lib/gcc/x86_64-linux-gnu/9/include-fixed"
ignoring nonexistent directory
"/usr/lib/gcc/x86_64-linux-gnu/9/../../../../x86_64-linux-gnu/include"
ignoring nonexistent directory "/usr/include/i386-linux-gnu"
#include "..." search starts here:
#include <...> search starts here:
 /usr/lib/gcc/x86_64-linux-gnu/9/include
 /usr/local/include
 /usr/include
End of search list.
GNU C17 (Ubuntu 9.3.0-17ubuntu1~20.04) version 9.3.0 (x86_64-linux-gnu)
   compiled by GNU C version 9.3.0, GMP version 6.2.0, MPFR version 4.0.2, MPC
   version 1.1.0, isl version isl-0.22.1-GMP

GGC heuristics: --param ggc-min-expand=100 --param ggc-min-heapsize=131072
Compiler executable checksum: bbf13931d8de1abe14040c9909cb6969
COLLECT_GCC_OPTIONS='-m32' '-fno-pic' '-v' '-mtune=generic' '-march=i686'
 as -v --32 -o /tmp/ccRHAlr7.o /tmp/ccWCdCj6.s
GNU assembler version 2.34 (x86_64-linux-gnu) using BFD version (GNU Binutils for
Ubuntu) 2.34
COMPILER_PATH=/usr/lib/gcc/x86_64-linux-gnu/9/:/usr/lib/gcc/x86_64-linux-gnu/9/:/usr
/lib/gcc/x86_64-linux-gnu/:/usr/lib/gcc/x86_64-linux-gnu/9/:/usr/lib/gcc/x86_64-linu
x-gnu/
LIBRARY_PATH=/usr/lib/gcc/x86_64-linux-gnu/9/32/:/usr/lib/gcc/x86_64-linux-gnu/9/../
../../i386-linux-gnu/:/usr/lib/gcc/x86_64-linux-gnu/9/../../../../lib32/:/lib/i386-l
inux-gnu/:/lib/../lib32/:/usr/lib/i386-linux-gnu/:/usr/lib/../lib32/:/usr/lib/gcc/x8
6_64-linux-gnu/9/:/usr/lib/gcc/x86_64-linux-gnu/9/../../../i386-linux-gnu/:/usr/lib/
gcc/x86_64-linux-gnu/9/../../../:/lib/i386-linux-gnu/:/lib/:/usr/lib/i386-linux-gnu/
:/usr/lib/
```

Compiler Driver command example (continued)

```
COLLECT_GCC_OPTIONS='-m32' '-fno-pic' '-v' '-mtune=generic' '-march=i686'
 /usr/lib/gcc/x86_64-linux-gnu/9/collect2 -plugin
 /usr/lib/gcc/x86_64-linux-gnu/9/liblto_plugin.so
 -plugin-opt=/usr/lib/gcc/x86_64-linux-gnu/9/lto-wrapper
 -plugin-opt=-fresolution=/tmp/ccQIh2N8.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 --build-id
 --eh-frame-hdr -m elf_i386 --hash-style=gnu --as-needed -dynamic-linker
 /lib/ld-linux.so.2 -pie -z now -z relro
 /usr/lib/gcc/x86_64-linux-gnu/9/../../../../lib32/Scrt1.o
 /usr/lib/gcc/x86_64-linux-gnu/9/../../../../lib32/crti.o
 /usr/lib/gcc/x86_64-linux-gnu/9/32/crtbeginS.o -L/usr/lib/gcc/x86_64-linux-gnu/9/32
 -L/usr/lib/gcc/x86_64-linux-gnu/9/../../../i386-linux-gnu
 -L/usr/lib/gcc/x86_64-linux-gnu/9/../../../../lib32 -L/lib/i386-linux-gnu
 -L/lib/../lib32 -L/usr/lib/i386-linux-gnu -L/usr/lib/../lib32
 -L/usr/lib/gcc/x86_64-linux-gnu/9
 -L/usr/lib/gcc/x86_64-linux-gnu/9/../../../i386-linux-gnu
 -L/usr/lib/gcc/x86_64-linux-gnu/9/../../.. -L/lib/i386-linux-gnu
 -L/usr/lib/i386-linux-gnu /tmp/ccRHAlr7.o -lgcc --push-state --as-needed -lgcc_s
 --pop-state -lc -lgcc --push-state --as-needed -lgcc_s --pop-state
 /usr/lib/gcc/x86_64-linux-gnu/9/32/crtendS.o
 /usr/lib/gcc/x86_64-linux-gnu/9/../../../../lib32/crtn.o
COLLECT_GCC_OPTIONS='-m32' '-fno-pic' '-v' '-mtune=generic' '-march=i686'
```

Linker

In computer science, a linker or link editor is a program that takes one or more objects generated by a compiler and combines them into a single executable program. It can happen in various stages.

1. Linking can be done at compile time
2. at load time (by loaders)
3. at run time (by application programs)

Linker

Compilation refers to the processing of source code files (.c, .cc, or .cpp) and the creation of an 'object' file. This step doesn't create anything the user can actually run. Instead, the compiler merely produces the machine language instructions that correspond to the source code file that was compiled.

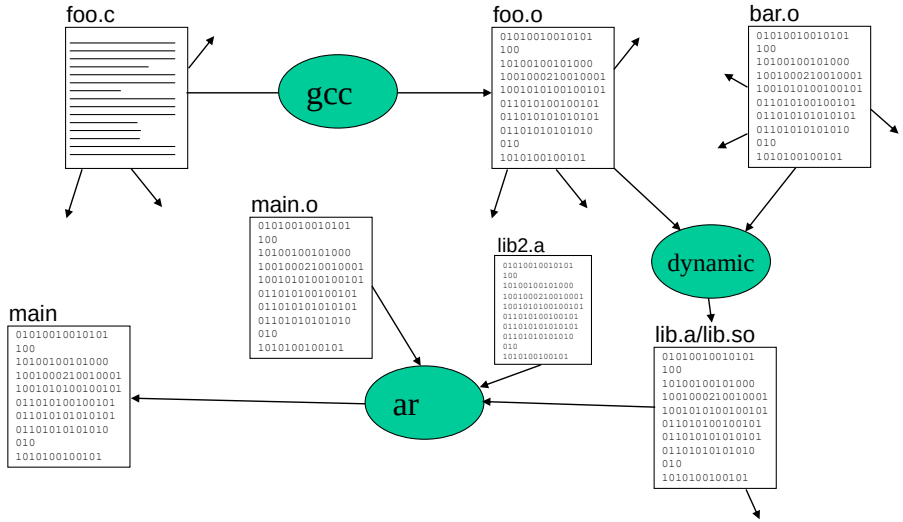
Linking refers to the creation of a single executable file from multiple object files. In this step, it is common that the linker will complain about undefined functions (commonly, main itself).

Why Linking: (1) Modularity

- Program can be written as a collection of smaller source files, rather than one monolithic mass.
- Can build libraries of common functions (more on this later)
  - e.g., Math library, standard C library

Why Linking: (2) Efficiency

- Time: Separate compilation
  - Change one source file, compile, and then relink.
  - No need to recompile other source files.
- Space: Libraries
  - Common functions can be aggregated into a single file...
  - Yet executable files and running memory images contain only code for the functions they actually use.
  - Non-shared v.s. Shared Library




```
$ ar -tv libc.a
rw-r--r-- 0/0   2488 Dec 31 19:00 1969 init-first.o
rw-r--r-- 0/0  12992 Dec 31 19:00 1969 libc-start.o
rw-r--r-- 0/0    608 Dec 31 19:00 1969 sysdep.o
rw-r--r-- 0/0   2704 Dec 31 19:00 1969 version.o
rw-r--r-- 0/0   2432 Dec 31 19:00 1969 check_fds.o
rw-r--r-- 0/0   3432 Dec 31 19:00 1969 libc-tls.o
rw-r--r-- 0/0   2184 Dec 31 19:00 1969 elf-init.o
rw-r--r-- 0/0   1144 Dec 31 19:00 1969 dso_handle.o
rw-r--r-- 0/0   1104 Dec 31 19:00 1969 errno.o
rw-r--r-- 0/0   1384 Dec 31 19:00 1969 errno-loc.o
rw-r--r-- 0/0   3152 Dec 31 19:00 1969 iconv_open.o
rw-r--r-- 0/0   3104 Dec 31 19:00 1969 iconv.o
rw-r--r-- 0/0   1480 Dec 31 19:00 1969 iconv_close.o
rw-r--r-- 0/0   4688 Dec 31 19:00 1969 gconv_open.o
rw-r--r-- 0/0   2648 Dec 31 19:00 1969 gconv.o
rw-r--r-- 0/0   1576 Dec 31 19:00 1969 gconv_close.o
rw-r--r-- 0/0  18192 Dec 31 19:00 1969 gconv_db.o
rw-r--r-- 0/0  14408 Dec 31 19:00 1969 gconv_conf.o
...
```

main.c

```
extern void swap();

int buf[2] = {1, 2};
int main()
{
    swap();
    return 0;
}
```

swap.c

```
extern int buf[];

int *bufp0 = &buf[0];
static int *bufp1;

void swap()
{
    int temp;

    bufp1 = &buf[1];
    temp = *bufp0;
    *bufp0 = *bufp1;
    *bufp1 = temp;
}
```

main.o

```
schen@linux:~/comp6700/lec04$ objdump -r -d main.o
00000000 <main>:
   0: f3 0f 1e fb           endbr32
   4: 55                    push   %ebp
   5: 89 e5                 mov    %esp,%ebp
   7: 83 e4 f0              and    $0xfffffff0,%esp
   a: e8 fc ff ff ff        call   b <main+0xb>
     b: R_386_PC32 swap
   f: b8 00 00 00 00        mov    $0x0,%eax
  14: c9                    leave
  15: c3                    ret


schen@linux:~/comp6700/lec04$ objdump -r  main.o

main.o:     file format elf32-i386

RELOCATION RECORDS FOR [.text]:
OFFSET   TYPE              VALUE
0000000b R_386_PC32        swap
```

Dynamic Linking

```
NAME
       ld.so, ld-linux.so - dynamic linker/loader

SYNOPSIS
       The  dynamic  linker  can  be  run either indirectly by running some dynamically
       linked program or shared object (in which case no command-line options to the
       dynamic linker can be passed  and,  in the  ELF  case,  the  dynamic linker
       which is stored in the .interp section of the program is executed) ...
```

test_foo.c

```
#include <stdio.h>

extern int foo(void);

int main () {
    printf("foo returns %d\n",foo());
    return 0;
}
```

foo.c

```
static int num = 6700;

int foo(void) {
    return num;
}
```

test_foo

```
$ gcc -m32 -fPIC -shared -o libfoo.so foo.c
$ gcc  -m32 -no-pie -fno-pic -ggdb3 -Wall -Wpedantic
-fno-stack-protector test_foo.c -o test_foo libfoo.so

$ export LD_LIBRARY_PATH=$LD_LIBRARY_PATH$:.

$ ldd test_foo
   linux-gate.so.1 (0xf7ed9000)
   libfoo.so (0xf7ecf000)
   libc.so.6 => /lib/i386-linux-gnu/libc.so.6 (0xf7ccc000)
   /lib/ld-linux.so.2 (0xf7eda000)
```

```
08049030 <.plt>:
 8049030: ff 35 04 c0 04 08     pushl  0x804c004
 8049036: ff 25 08 c0 04 08     jmp    *0x804c008
 804903c: 0f 1f 40 00           nopl   0x0(%eax)
 8049040: f3 0f 1e fb           endbr32
 8049044: 68 00 00 00 00        push   $0x0
 8049049: e9 e2 ff ff ff        jmp    8049030 <printf@plt-0x40>
 804904e: 66 90                 xchg   %ax,%ax
 8049050: f3 0f 1e fb           endbr32
 8049054: 68 08 00 00 00        push   $0x8
 8049059: e9 d2 ff ff ff        jmp    8049030 <printf@plt-0x40>
 804905e: 66 90                 xchg   %ax,%ax
 8049060: f3 0f 1e fb           endbr32
 8049064: 68 10 00 00 00        push   $0x10
 8049069: e9 c2 ff ff ff        jmp    8049030 <printf@plt-0x40>
 804906e: 66 90                 xchg   %ax,%ax

Disassembly of section .plt.sec:

08049070 <printf@plt>:
 8049070: f3 0f 1e fb           endbr32
 8049074: ff 25 0c c0 04 08     jmp    *0x804c00c
 804907a: 66 0f 1f 44 00 00     nopw   0x0(%eax,%eax,1)

08049080 <__libc_start_main@plt>:
 8049080: f3 0f 1e fb           endbr32
 8049084: ff 25 10 c0 04 08     jmp    *0x804c010
 804908a: 66 0f 1f 44 00 00     nopw   0x0(%eax,%eax,1)

08049090 <foo@plt>:
 8049090: f3 0f 1e fb           endbr32
 8049094: ff 25 14 c0 04 08     jmp    *0x804c014
 804909a: 66 0f 1f 44 00 00     nopw   0x0(%eax,%eax,1)
```

```
080491b6 <main>:
 80491b6: f3 0f 1e fb           endbr32
 80491ba: 8d 4c 24 04           lea    0x4(%esp),%ecx
 80491be: 83 e4 f0              and    $0xfffffff0,%esp
 80491c1: ff 71 fc              pushl  -0x4(%ecx)
 80491c4: 55                    push   %ebp
 80491c5: 89 e5                 mov    %esp,%ebp
 80491c7: 51                    push   %ecx
 80491c8: 83 ec 04              sub    $0x4,%esp
 80491cb: e8 c0 fe ff ff        call   8049090 <foo@plt>
 80491d0: 83 ec 08              sub    $0x8,%esp
 80491d3: 50                    push   %eax
 80491d4: 68 08 a0 04 08        push   $0x804a008
 80491d9: e8 92 fe ff ff        call   8049070 <printf@plt>
 ...
 $ readelf -e test_foo
ELF Header:
  Magic:   7f 45 4c 46 01 01 01 00 00 00 00 00 00 00 00 00
  Class:                             ELF32
  Data:                              2's complement, little endian
  Version:                           1 (current)
  OS/ABI:                            UNIX - System V
  ABI Version:                       0
  Type:                              EXEC (Executable file)
  Machine:                           Intel 80386
  Version:                           0x1
  Entry point address:               0x80490a0
...
```

```
Section Headers:
  [Nr] Name              Type            Addr     Off    Size   ES Flg Lk Inf Al
  [ 0]                   NULL            00000000 000000 000000 00      0   0  0
  [ 1] .interp           PROGBITS        080481b4 0001b4 000013 00   A  0   0  1
  [ 2] .note.gnu.build-i NOTE            080481c8 0001c8 000024 00   A  0   0  4
  [ 3] .note.gnu.propert NOTE            080481ec 0001ec 00001c 00   A  0   0  4
  [ 4] .note.ABI-tag     NOTE            08048208 000208 000020 00   A  0   0  4
  [ 5] .gnu.hash         GNU_HASH        08048228 000228 000020 04   A  6   0  4
  [ 6] .dynsym           DYNSYM          08048248 000248 000060 10   A  7   1  4
  [ 7] .dynstr           STRTAB          080482a8 0002a8 00005a 00   A  0   0  1
  [ 8] .gnu.version      VERSYM          08048302 000302 00000c 02   A  6   0  2
  [ 9] .gnu.version_r    VERNEED         08048310 000310 000020 00   A  7   1  4
  [10] .rel.dyn          REL             08048330 000330 000008 08   A  6   0  4
  [11] .rel.plt          REL             08048338 000338 000018 08  AI  6  24  4
  [12] .init             PROGBITS        08049000 001000 000024 00  AX  0   0  4
  [13] .plt              PROGBITS        08049030 001030 000040 04  AX  0   0 16
  [14] .plt.sec          PROGBITS        08049070 001070 000030 10  AX  0   0 16
  [15] .text             PROGBITS        080490a0 0010a0 0001c9 00  AX  0   0 16
  [16] .fini             PROGBITS        0804926c 00126c 000018 00  AX  0   0  4
  [17] .rodata           PROGBITS        0804a000 002000 000018 00   A  0   0  4
  [18] .eh_frame_hdr     PROGBITS        0804a018 002018 00004c 00   A  0   0  4
  [19] .eh_frame         PROGBITS        0804a064 002064 00011c 00   A  0   0  4
  [20] .init_array       INIT_ARRAY      0804bf04 002f04 000004 04  WA  0   0  4
  [21] .fini_array       FINI_ARRAY      0804bf08 002f08 000004 04  WA  0   0  4
  [22] .dynamic          DYNAMIC         0804bf0c 002f0c 0000f0 08  WA  7   0  4
  [23] .got              PROGBITS        0804bffc 002ffc 000004 04  WA  0   0  4
  [24] .got.plt          PROGBITS        0804c000 003000 000018 04  WA  0   0  4
  [25] .data             PROGBITS        0804c018 003018 000008 00  WA  0   0  4
  [26] .bss              NOBITS          0804c020 003020 000004 00  WA  0   0  1

 Contents of section .got.plt:
 804c000 0cbf0408 00000000 00000000 40900408  ............@...
 804c010 50900408 60900408                    P...`...
```

Loader

Compiler: A language translator that converts a complete program into machine language to produce a program that the computer can process in its entirety.

Linker: Utility program which takes one or more compiled object files and combines them into an executable file or another object file.

Loader: loads the executable code into memory ,creates the program and data stack , initializes the registers and starts the code running.

Statically Linked Binary

```
schen@linux:~/comp6700/lec04$ strace ./hellos
execve("./hellos", ["./hellos"], 0x7fffbd0c2100 /* 27 vars */) = 0
strace: [ Process PID=727737 runs in 32 bit mode. ]
arch_prctl(0x3001 /* ARCH_??? */, 0xfff3b560) = -1 EINVAL (Invalid argument)
brk(NULL)                                 = 0x8f25000
brk(0x8f25d40)                            = 0x8f25d40
set_thread_area({entry_number=-1, base_addr=0x8f25840,
    limit=0x0fffff, seg_32bit=1, contents=0, read_exec_only=0,
    limit_in_pages=1, seg_not_present=0, useable=1}) = 0 (entry_number=12)
uname({sysname="Linux", nodename="linux", ...}) = 0
readlink("/proc/self/exe", "/home/schen/comp6700/lec04/hellos", 4096) = 31
brk(0x8f46d40)                            = 0x8f46d40
brk(0x8f47000)                            = 0x8f47000
access("/etc/ld.so.nohwcap", F_OK)        = -1 ENOENT (No such file or directory)
mprotect(0x80e3000, 8192, PROT_READ)      = 0
fstat64(1, {st_mode=S_IFCHR|0620, st_rdev=makedev(0x88, 0x1), ...}) = 0
write(1, "Hello World!\n", 13Hello World!
)           = 13
exit_group(0)                             = ?
+++ exited with 0 +++
```

Dynamically Linked Binary

```
schen@linux:~/comp6700/lec04$ strace ./hello
execve("./hello", ["./hello"], 0x7fff46583a90 /* 55 vars */) = 0
brk(NULL)                                       = 0x560a7251c000
arch_prctl(0x3001 /* ARCH_??? */, 0x7ffd48aa4a10) = -1 EINVAL (Invalid argument)
mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7f04ea3de000
access("/etc/ld.so.preload", R_OK)      = -1 ENOENT (No such file or directory)
openat(AT_FDCWD, "glibc-hwcaps/x86-64-v3/libc.so.6", O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or ...
openat(AT_FDCWD, "glibc-hwcaps/x86-64-v2/libc.so.6", O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or ...
openat(AT_FDCWD, "tls/haswell/x86_64/libc.so.6", O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or ...
openat(AT_FDCWD, "tls/haswell/libc.so.6", O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or directory)
openat(AT_FDCWD, "tls/x86_64/libc.so.6", O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or directory)
openat(AT_FDCWD, "tls/libc.so.6", O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or directory)
openat(AT_FDCWD, "haswell/x86_64/libc.so.6", O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or directory)
openat(AT_FDCWD, "haswell/libc.so.6", O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or directory)
openat(AT_FDCWD, "x86_64/libc.so.6", O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or directory)
openat(AT_FDCWD, "libc.so.6", O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or directory)
openat(AT_FDCWD, "./glibc-hwcaps/x86-64-v3/libc.so.6", O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file ...
openat(AT_FDCWD, "./glibc-hwcaps/x86-64-v2/libc.so.6", O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file ...
openat(AT_FDCWD, "./tls/haswell/x86_64/libc.so.6", O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or ...
openat(AT_FDCWD, "./tls/haswell/libc.so.6", O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or directory)
openat(AT_FDCWD, "./tls/x86_64/libc.so.6", O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or directory)
openat(AT_FDCWD, "./tls/libc.so.6", O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or directory)
openat(AT_FDCWD, "./haswell/x86_64/libc.so.6", O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or ...
openat(AT_FDCWD, "./haswell/libc.so.6", O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or directory)
openat(AT_FDCWD, "./x86_64/libc.so.6", O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or directory)
openat(AT_FDCWD, "./libc.so.6", O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or directory)
openat(AT_FDCWD, "/etc/ld.so.cache", O_RDONLY|O_CLOEXEC) = 3
newfstatat(3, "", {st_mode=S_IFREG|0644, st_size=71307, ...}, AT_EMPTY_PATH) = 0
mmap(NULL, 71307, PROT_READ, MAP_PRIVATE, 3, 0) = 0x7f04ea3cc000
close(3)                                        = 0
openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libc.so.6", O_RDONLY|O_CLOEXEC) = 3
read(3, "\177ELF\2\1\1\3\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0P\237\2\0\0\0\0\0"..., 832) = 832
```

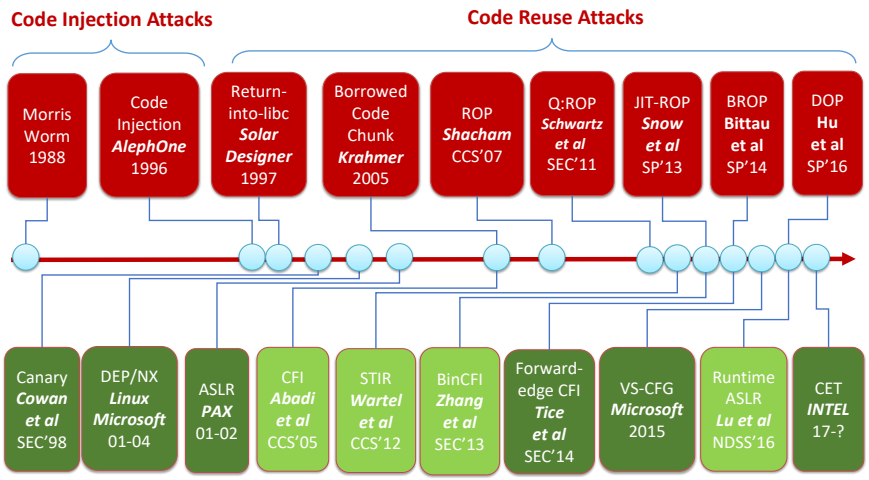
Dynamically Linked Binary (continued)

```
pread64(3, "\6\0\0\0\4\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0"..., 784, 64) = 784
pread64(3, "\4\0\0\0 \0\0\0\5\0\0\0GNU\0\2\0\0\300\4\0\0\0\3\0\0\0\0\0\0\0"..., 48, 848) = 48
pread64(3, "\4\0\0\0\24\0\0\0\3\0\0\0GNU\0i8\235HZ\227\223\333\350s\360\352,\223\340."..., 68, 896) = 68
newfstatat(3, "", {st_mode=S_IFREG|0644, st_size=2216304, ...}, AT_EMPTY_PATH) = 0
pread64(3, "\6\0\0\0\4\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0"..., 784, 64) = 784
mmap(NULL, 2260560, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7f04ea000000
mmap(0x7f04ea028000, 1658880, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x28000) = ...
mmap(0x7f04ea1bd000, 360448, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1bd000) = ...
mmap(0x7f04ea215000, 24576, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x214000) = ...
mmap(0x7f04ea21b000, 52816, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) = ...
close(3)                                = 0
mmap(NULL, 12288, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7f04ea3c9000
arch_prctl(ARCH_SET_FS, 0x7f04ea3c9740) = 0
set_tid_address(0x7f04ea3c9a10)         = 9271
set_robust_list(0x7f04ea3c9a20, 24)     = 0
rseq(0x7f04ea3ca0e0, 0x20, 0, 0x53053053) = 0
mprotect(0x7f04ea215000, 16384, PROT_READ) = 0
mprotect(0x560a708f6000, 4096, PROT_READ) = 0
mprotect(0x7f04ea418000, 8192, PROT_READ) = 0
prlimit64(0, RLIMIT_STACK, NULL, {rlim_cur=8192*1024, rlim_max=RLIM64_INFINITY}) = 0
munmap(0x7f04ea3cc000, 71307)           = 0
newfstatat(1, "", {st_mode=S_IFCHR|0620, st_rdev=makedev(0x88, 0), ...}, AT_EMPTY_PATH) = 0
getrandom("\x1f\xee\xff\xc7\x1d\x2b\xe5\x39", 8, GRND_NONBLOCK) = 8
brk(NULL)                               = 0x560a7251c000
brk(0x560a7253d000)                     = 0x560a7253d000
write(1, "Hello world!\n", 13Hello world!
)          = 13
exit_group(0)                           = ?
+++ exited with 0 +++
```

Dynamically Linked Binary with LD_PRELOAD

```
$ LD_PRELOAD=/lib/x86_64-linux-gnu/libc.so.6:/lib64/ld-linux-x86-64.so.2:linux-vdso.so.1 strace ./hello
execve("./hello", ["./hello"], 0x7ffee8804040 /* 56 vars */) = 0
brk(NULL)                                = 0x5652a8651000
arch_prctl(0x3001 /* ARCH_??? */, 0x7ffd6bcf8ac0) = -1 EINVAL (Invalid argument)
mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7f433e716000
openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libc.so.6", O_RDONLY|O_CLOEXEC) = 3
read(3, "\177ELF\2\1\1\3\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0P\237\2\0\0\0\0\0"..., 832) = 832
pread64(3, "\6\0\0\0\4\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0"..., 784, 64) = 784
pread64(3, "\4\0\0\0 \0\0\0\5\0\0\0GNU\0\2\0\0\300\4\0\0\0\3\0\0\0\0\0\0\0"..., 48, 848) = 48
pread64(3, "\4\0\0\0\24\0\0\0\3\0\0\0GNU\0i8\235HZ\227\223\333\350s\360\352,\223\340."..., 68, 896) = 68
newfstatat(3, "", {st_mode=S_IFREG|0644, st_size=2216304, ...}, AT_EMPTY_PATH) = 0
pread64(3, "\6\0\0\0\4\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0"..., 784, 64) = 784
mmap(NULL, 2260560, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7f433e400000
mmap(0x7f433e428000, 1658880, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x28000) = ...
mmap(0x7f433e5bd000, 360448, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1bd000) = ...
mmap(0x7f433e615000, 24576, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x214000) = ...
mmap(0x7f433e61b000, 52816, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) = ...
close(3)                                 = 0
access("/etc/ld.so.preload", R_OK)       = -1 ENOENT (No such file or directory)
mmap(NULL, 12288, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7f433e713000
arch_prctl(ARCH_SET_FS, 0x7f433e713740) = 0 set_tid_address(0x7f433e713a10)          = 9265
set_robust_list(0x7f433e713a20, 24)      = 0 rseq(0x7f433e7140e0, 0x20, 0, 0x53053053) = 0
mprotect(0x7f433e615000, 16384, PROT_READ) = 0
mprotect(0x5652a66d5000, 4096, PROT_READ) = 0
mprotect(0x7f433e750000, 8192, PROT_READ) = 0
prlimit64(0, RLIMIT_STACK, NULL, {rlim_cur=8192*1024, rlim_max=RLIM64_INFINITY}) = 0
newfstatat(1, "", {st_mode=S_IFCHR|0620, st_rdev=makedev(0x88, 0), ...}, AT_EMPTY_PATH) = 0
getrandom("\x35\xaa\xb2\xf2\x9d\x9d\x62\xf5", 8, GRND_NONBLOCK) = 8
brk(NULL)                                = 0x5652a8651000
brk(0x5652a8672000)                      = 0x5652a8672000
write(1, "Hello world!\n", 13Hello world!)          = 13
exit_group(0)                            = ?
+++ exited with 0 +++
```

Thank You



[1]

schen@auburn.edu
schuan.github.io

[1] Instructor appreciates the help from Prof. Zhiqiang Lin.