Link 5.B Relocation

Young W. Lim

2019-03-22 Fri

Outline

1. Based on
2. x86 relocation types
   - ELF relocation types summary
   - ELF relocation types
3. Disassembled listings of `swap.o`
   - `objdump -d swap.o`
   - `objdump -dr swap.o`
4. Disassembled listings of `smain.o`
   - `objdump -d smain.o`
   - `objdump -dr smain.o`
5. Disassembled listings of `smain.out`
   - `objdump -d smain.out` compiled with `-nostdlib`
   - `objdump -d smain.out`
6. Symbol table listings
   - symbol table : `swap.o`
   - symbol table : `smain.o`
   - symbol table : `smain.out`

Based on

"Self-service Linux: Mastering the Art of Problem Determination",
Mark Wilding
"Computer Architecture: A Programmer's Perspective",
Bryant & O'Hallaron

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

ELF relocation types (A)

name	value	field	calculation
R_386_NONE	0	None	None
R_386_32	1	word32	S+A
R_386_PC32	2	word32	S+A-P
R_386_GOT32	3	word32	G+A
R_386_PLT32	4	word32	L+A-P
R_386_COPY	5	None	None
R_386_GLOB_DAT	6	word32	S
R_386_JMP_SLOT	7	word32	S
R_386_RELATIVE	8	word32	B+A
R_386_GOTOFF	9	word32	S+A-GOT
R_386_GOTPC	10	word32	GOT+A-P
R_386_32PLT	11	word32	L+A

https://docs.oracle.com/cd/E19683-01/817-3677/chapter6-26/index.html

ELF relocation types (B)

- A represents the addend used to compute
  the value of the relocatable field.
- B represents the base address at which a shared object
  has been loaded into memory during execution.
  Generally, a shared object is built with a 0 base virtual address,
  but the execution address will be different.
- G represents the offset into the global offset table(GOT)
  at which the relocation entry's symbol will reside during execution.
- GOT represents the address of the global offset table(GOT).

https://refspecs.linuxfoundation.org/elf/x86_64-abi-0.95.pdf

ELF relocation types (C)

- L represents the place (section offset or address) of the Procedure Linkage Table (PLT) entry for a symbol.
- P represents the place (section offset or address) of the storage unit being relocated (computed using `r_offset`).
- S represents the value of the symbol whose index resides in the relocation entry

`https://refspecs.linuxfoundation.org/elf/x86_64-abi-0.95.pdf`

(1) R_386_GOT32

- Computes the distance from the base of the global offset table to the symbol's global offset table entry
- It also instructs the link editor to create a global offset table.

$G + A - P$

- G represents the offset into the global offset table(GOT) at which the relocation entry's symbol will reside during execution.
- A represents the addend used to compute the value of the relocatable field.
- P represents the place (section offset or address) of the storage unit being relocated (computed using `r_offset`).

`https://docs.oracle.com/cd/E19683-01/817-3677/chapter6-26/index.html`

(2) R_386_PLT32

- Computes the address of the symbol's procedure linkage table entry
- instructs the link editor to create a procedure linkage table.

$L + A - P$

- L represents the place (section offset or address) of the Procedure Linkage Table (PLT) entry for a symbol.
- A represents the addend used to compute the value of the relocatable field.
- P represents the place (section offset or address) of the storage unit being relocated (computed using `r_offset`).

https://docs.oracle.com/cd/E19683-01/817-3677/chapter6-26/index.html

(3) R_386_COPY

- Created by the link-editor for dynamic executables to preserve a read-only text segment.
- its offset member refers to a location in a writable segment.
- The symbol table index specifies a symbol that should exist both in the current object file and in a shared object.
- during execution, the runtime linker copies data associated with the shared object's symbol to the location specified by the offset
- None

https://docs.oracle.com/cd/E19683-01/817-3677/chapter6-26/index.html

(4) R_386_GLOB_DAT

- used to set a global offset table (GOT) entry
  to the address of the specified symbol
- the special relocation type enable you to determine
  the correspondence between symbols and
  global offset table (GOT) entries

S

- S represents the value of the symbol
  whose index resides in the relocation entry

https://docs.oracle.com/cd/E19683-01/817-3677/chapter6-26/index.html

(5) R_386_JMP_SLOT

- Created by the link editor for dynamic objects
  to provide lazy binding.
- its offset member gives the location
  of a procedure linkage table (PLT) entry.
- The runtime linker modifies the PLT entry
  to transfer control to the designated symbol address

S

- S represents the value of the symbol
  whose index resides in the relocation entry

https://docs.oracle.com/cd/E19683-01/817-3677/chapter6-26/index.html

(6).(a) R_386_RELATIVE

- created by the link editor for dynamic objects.
- its offset member gives the location within a shared object that contains a value representing a relative address.
- The runtime linker computes the corresponding virtual address by adding the virtual address at which the shared object is loaded to the relative address.
- Relocation entries for this type must specify 0 for the symbol table index.
- $B + A$

https://docs.oracle.com/cd/E19683-01/817-3677/chapter6-26/index.html

(6).(b) R_386_RELATIVE

$B + A$

- B represents the base address at which a shared object has been loaded into memory during execution. Generally, a shared object is built with a 0 base virtual address, but the execution address will be different.
- A represents the addend used to compute the value of the relocatable field.

https://docs.oracle.com/cd/E19683-01/817-3677/chapter6-26/index.html

(7) R_386_GOTOFF

- Computes the difference between a symbol's value and the address of the global offset table.
- It also instructs the link editor to create the global offset table.

$S + A - GOT$

- S represents the value of the symbol whose index resides in the relocation entry
- A represents the addend used to compute the value of the relocatable field.
- GOT represents the address of the global offset table(GOT).

https://docs.oracle.com/cd/E19683-01/817-3677/chapter6-26/index.html

(8).(a) R_386_GOTPC

- Resembles R_386_PC32, except that it uses the addr1ess of the global offset table (GOT) in its calculation.
- The symbol referenced in this relocation normally is GLOBAL_OFFSET_TABLE, which also instructs the link-editor to create the global offset table.
- $GOT + A - P$ (R_386_GOT32)
- $S + A - P$ (R_386_PC32)

https://docs.oracle.com/cd/E19683-01/817-3677/chapter6-26/index.html

(8).(b) R_386_GOTPC

$GOT + A - P$ (R_386_GOT32) / $S + A - P$ (R_386_PC32)

- GOT represents the address of the global offset table(GOT).
- A represents the addend used to compute the value of the relocatable field.
- P represents the place (section offset or address) of the storage unit being relocated (computed using `r_offset`).
- S represents the value of the symbol whose index resides in the relocation entry

`https://docs.oracle.com/cd/E19683-01/817-3677/chapter6-26/index.html`

objdump -d swap.o (1)

```
young@USys2:~/smain$ objdump -d swap.o

swap.o:     formato del fichero elf32-i386


Desensamblado de la sección .text:

00000000 <swap>:
   0:   55                      push   %ebp
   1:   89 e5                   mov    %esp,%ebp
   3:   83 ec 10                sub    $0x10,%esp
   6:   e8 fc ff ff ff          call   7 <swap+0x7>
   b:   05 01 00 00 00          add    $0x1,%eax
  10:   8b 90 00 00 00 00       mov    0x0(%eax),%edx
  16:   8b 88 00 00 00 00       mov    0x0(%eax),%ecx
  1c:   8d 49 04                lea    0x4(%ecx),%ecx
  1f:   89 0a                   mov    %ecx,(%edx)
  21:   8b 90 00 00 00 00       mov    0x0(%eax),%edx
  27:   8b 12                   mov    (%edx),%edx
  29:   89 55 fc                mov    %edx,-0x4(%ebp)
  2c:   8b 90 00 00 00 00       mov    0x0(%eax),%edx
```

objdump -d swap.o (2)

```
  32:   8b 0a                   mov    (%edx),%ecx
  34:   8b 90 00 00 00 00       mov    0x0(%eax),%edx
  3a:   8b 09                   mov    (%ecx),%ecx
  3c:   89 0a                   mov    %ecx,(%edx)
  3e:   8b 80 00 00 00 00       mov    0x0(%eax),%eax
  44:   8b 00                   mov    (%eax),%eax
  46:   8b 55 fc                mov    -0x4(%ebp),%edx
  49:   89 10                   mov    %edx,(%eax)
  4b:   90                      nop
  4c:   c9                      leave
  4d:   c3                      ret

Desensamblado de la sección .text.__x86.get_pc_thunk.ax:

00000000 <__x86.get_pc_thunk.ax>:
   0:   8b 04 24                mov    (%esp),%eax
   3:   c3                      ret
```

objdump -dr swap.o (1)

```
Desensamblado de la sección .text:

00000000 <swap>:
   0:   55                      push   %ebp
   1:   89 e5                   mov    %esp,%ebp
   3:   83 ec 10                sub    $0x10,%esp
   6:   e8 fc ff ff ff          call   7 <swap+0x7>
                        7: R_386_PC32   __x86.get_pc_thunk.ax
   b:   05 01 00 00 00          add    $0x1,%eax
                        c: R_386_GOTPC  _GLOBAL_OFFSET_TABLE_
  10:   8b 90 00 00 00 00       mov    0x0(%eax),%edx
                        12: R_386_GOT32X        p1
  16:   8b 88 00 00 00 00       mov    0x0(%eax),%ecx
                        18: R_386_GOT32X        buf
  1c:   8d 49 04                lea    0x4(%ecx),%ecx
  1f:   89 0a                   mov    %ecx,(%edx)
  21:   8b 90 00 00 00 00       mov    0x0(%eax),%edx
                        23: R_386_GOTOFF        p0
  27:   8b 12                   mov    (%edx),%edx
  29:   89 55 fc                mov    %edx,-0x4(%ebp)
  2c:   8b 90 00 00 00 00       mov    0x0(%eax),%edx
                        2e: R_386_GOT32X        p1
  32:   8b 0a                   mov    (%edx),%ecx
  34:   8b 90 00 00 00 00       mov    0x0(%eax),%edx
```

objdump -dr swap.o (2)

```
  3a:   8b 09                   mov    (%ecx),%ecx
  3c:   89 0a                   mov    %ecx,(%edx)
  3e:   8b 80 00 00 00 00       mov    0x0(%eax),%eax
                        40: R_386_GOT32X        p1
  44:   8b 00                   mov    (%eax),%eax
  46:   8b 55 fc                mov    -0x4(%ebp),%edx
  49:   89 10                   mov    %edx,(%eax)
  4b:   90                      nop
  4c:   c9                      leave
  4d:   c3                      ret

Desensamblado de la sección .text.__x86.get_pc_thunk.ax:

00000000 <__x86.get_pc_thunk.ax>:
   0:   8b 04 24                mov    (%esp),%eax
   3:   c3                      ret
```

y

objdump -d smain.o (1)

```
young@USys2:~/smain$ objdump -d smain.o

smain.o:     formato del fichero elf32-i386


Desensamblado de la sección .text:

00000000 <main>:
   0:   8d 4c 24 04             lea    0x4(%esp),%ecx
   4:   83 e4 f0                and    $0xfffffff0,%esp
   7:   ff 71 fc                pushl  -0x4(%ecx)
   a:   55                      push   %ebp
   b:   89 e5                   mov    %esp,%ebp
   d:   53                      push   %ebx
   e:   51                      push   %ecx
   f:   e8 fc ff ff ff          call   10 <main+0x10>
  14:   05 01 00 00 00          add    $0x1,%eax
  19:   89 c3                   mov    %eax,%ebx
  1b:   e8 fc ff ff ff          call   1c <main+0x1c>
```

objdump -d smain.o (2)

```
  20:   b8 00 00 00 00          mov    $0x0,%eax
  25:   59                      pop    %ecx
  26:   5b                      pop    %ebx
  27:   5d                      pop    %ebp
  28:   8d 61 fc                lea    -0x4(%ecx),%esp
  2b:   c3                      ret

Desensamblado de la sección .text.__x86.get_pc_thunk.ax:

00000000 <__x86.get_pc_thunk.ax>:
   0:   8b 04 24                mov    (%esp),%eax
   3:   c3                      ret
```

objdump -dr smain.o (1)

```
smain.o:     formato del fichero elf32-i386


Desensamblado de la sección .text:

00000000 <main>:
   0:   8d 4c 24 04             lea    0x4(%esp),%ecx
   4:   83 e4 f0                and    $0xfffffff0,%esp
   7:   ff 71 fc                pushl  -0x4(%ecx)
   a:   55                      push   %ebp
   b:   89 e5                   mov    %esp,%ebp
   d:   53                      push   %ebx
   e:   51                      push   %ecx
   f:   e8 fc ff ff ff          call   10 <main+0x10>
                        10: R_386_PC32  __x86.get_pc_thunk.ax
  14:   05 01 00 00 00          add    $0x1,%eax
                        15: R_386_GOTPC _GLOBAL_OFFSET_TABLE_
  19:   89 c3                   mov    %eax,%ebx
  1b:   e8 fc ff ff ff          call   1c <main+0x1c>
                        1c: R_386_PLT32 swap
```

objdump -dr smain.o (2)

```
  20:   b8 00 00 00 00          mov    $0x0,%eax
  25:   59                      pop    %ecx
  26:   5b                      pop    %ebx
  27:   5d                      pop    %ebp
  28:   8d 61 fc                lea    -0x4(%ecx),%esp
  2b:   c3                      ret

Desensamblado de la sección .text.__x86.get_pc_thunk.ax:

00000000 <__x86.get_pc_thunk.ax>:
   0:   8b 04 24                mov    (%esp),%eax
   3:   c3                      ret
```

objdump -d smain.out (1) main

```
000001c0 <main>:
 1c0:   8d 4c 24 04             lea    0x4(%esp),%ecx
 1c4:   83 e4 f0                and    $0xfffffff0,%esp
 1c7:   ff 71 fc                pushl  -0x4(%ecx)
 1ca:   55                      push   %ebp
 1cb:   89 e5                   mov    %esp,%ebp
 1cd:   53                      push   %ebx
 1ce:   51                      push   %ecx
 1cf:   e8 18 00 00 00          call   1ec <__x86.get_pc_thunk.ax>
 1d4:   05 20 1e 00 00          add    $0x1e20,%eax
 1d9:   89 c3                   mov    %eax,%ebx
 1db:   e8 10 00 00 00          call   1f0 <swap>
 1e0:   b8 00 00 00 00          mov    $0x0,%eax
 1e5:   59                      pop    %ecx
 1e6:   5b                      pop    %ebx
 1e7:   5d                      pop    %ebp
 1e8:   8d 61 fc                lea    -0x4(%ecx),%esp
 1eb:   c3                      ret

000001ec <__x86.get_pc_thunk.ax>:
 1ec:   8b 04 24                mov    (%esp),%eax
 1ef:   c3                      ret
```

objdump -d smain.out (2) swap

```
000001f0 <swap>:
 1f0:   55                      push   %ebp
 1f1:   89 e5                   mov    %esp,%ebp
 1f3:   83 ec 10                sub    $0x10,%esp
 1f6:   e8 f1 ff ff ff          call   1ec <__x86.get_pc_thunk.ax>
 1fb:   05 f9 1d 00 00          add    $0x1df9,%eax
 200:   8d 90 18 00 00 00       lea    0x18(%eax),%edx
 206:   8d 88 0c 00 00 00       lea    0xc(%eax),%ecx
 20c:   8d 49 04                lea    0x4(%ecx),%ecx
 20f:   89 0a                   mov    %ecx,(%edx)
 211:   8b 90 14 00 00 00       mov    0x14(%eax),%edx
 217:   8b 12                   mov    (%edx),%edx
 219:   89 55 fc                mov    %edx,-0x4(%ebp)
 21c:   8d 90 18 00 00 00       lea    0x18(%eax),%edx
 222:   8b 0a                   mov    (%edx),%ecx
 224:   8b 90 14 00 00 00       mov    0x14(%eax),%edx
 22a:   8b 09                   mov    (%ecx),%ecx
 22c:   89 0a                   mov    %ecx,(%edx)
 22e:   8d 80 18 00 00 00       lea    0x18(%eax),%eax
 234:   8b 00                   mov    (%eax),%eax
 236:   8b 55 fc                mov    -0x4(%ebp),%edx
 239:   89 10                   mov    %edx,(%eax)
 23b:   90                      nop
 23c:   c9                      leave
```

objdump -d smain.out (1) .init

```
smain.out:     formato del fichero elf32-i386


Desensamblado de la sección .init:

00000360 <_init>:
 360:   53                      push   %ebx
 361:   83 ec 08                sub    $0x8,%esp
 364:   e8 97 00 00 00          call   400 <__x86.get_pc_thunk.bx>
 369:   81 c3 73 1c 00 00       add    $0x1c73,%ebx
 36f:   8b 83 18 00 00 00       mov    0x18(%ebx),%eax
 375:   85 c0                   test   %eax,%eax
 377:   74 05                   je     37e <_init+0x1e>
 379:   e8 3a 00 00 00          call   3b8 <__gmon_start__@plt>
 37e:   83 c4 08                add    $0x8,%esp
 381:   5b                      pop    %ebx
 382:   c3                      ret
```

objdump -d smain.out (2) .plt

```
Desensamblado de la sección .plt:

00000390 <.plt>:
 390:   ff b3 04 00 00 00       pushl  0x4(%ebx)
 396:   ff a3 08 00 00 00       jmp    *0x8(%ebx)
 39c:   00 00                   add    %al,(%eax)
        ...

000003a0 <__libc_start_main@plt>:
 3a0:   ff a3 0c 00 00 00       jmp    *0xc(%ebx)
 3a6:   68 00 00 00 00          push   $0x0
 3ab:   e9 e0 ff ff ff          jmp    390 <.plt>
```

objdump -d smain.out (3) .plt.got

```
Desensamblado de la sección .plt.got:

000003b0 <__cxa_finalize@plt>:
 3b0:   ff a3 14 00 00 00       jmp    *0x14(%ebx)
 3b6:   66 90                   xchg   %ax,%ax

000003b8 <__gmon_start__@plt>:
 3b8:   ff a3 18 00 00 00       jmp    *0x18(%ebx)
 3be:   66 90                   xchg   %ax,%ax
```

objdump -d smain.out (4) .text summary

```
Desensamblado de la sección .text:

000003c0 <_start>:
00000400 <__x86.get_pc_thunk.bx>:
00000410 <deregister_tm_clones>:
00000450 <register_tm_clones>:
000004a0 <__do_global_dtors_aux>:
000004f0 <frame_dummy>:
000004f9 <__x86.get_pc_thunk.dx>:
000004fd <main>:
00000529 <__x86.get_pc_thunk.ax>:
0000052d <swap>:
00000580 <__libc_csu_init>:
```

objdump -d smain.out (5) .fini

```
Desensamblado de la sección .fini:

000005e4 <_fini>:
 5e4:   53                      push   %ebx
 5e5:   83 ec 08                sub    $0x8,%esp
 5e8:   e8 13 fe ff ff          call   400 <__x86.get_pc_thunk.bx>
 5ed:   81 c3 ef 19 00 00       add    $0x19ef,%ebx
 5f3:   83 c4 08                add    $0x8,%esp
 5f6:   5b                      pop    %ebx
 5f7:   c3                      ret
```

objdump -d smain.out (6) _start (a)

```
000003c0 <_start>:
 3c0:   31 ed                   xor    %ebp,%ebp
 3c2:   5e                      pop    %esi
 3c3:   89 e1                   mov    %esp,%ecx
 3c5:   83 e4 f0                and    $0xfffffff0,%esp
 3c8:   50                      push   %eax
 3c9:   54                      push   %esp
 3ca:   52                      push   %edx
 3cb:   e8 22 00 00 00          call   3f2 <_start+0x32>
 3d0:   81 c3 0c 1c 00 00       add    $0x1c0c,%ebx
 3d6:   8d 83 04 e6 ff ff       lea    -0x19fc(%ebx),%eax
 3dc:   50                      push   %eax
 3dd:   8d 83 a4 e5 ff ff       lea    -0x1a5c(%ebx),%eax
 3e3:   50                      push   %eax
 3e4:   51                      push   %ecx
 3e5:   56                      push   %esi
 3e6:   ff b3 1c 00 00 00       pushl  0x1c(%ebx)
```

objdump -d smain.out (7) _start (b)

```
3ec:   e8 af ff ff ff          call   3a0 <__libc_start_main@plt>
3f1:   f4                      hlt
3f2:   8b 1c 24                mov    (%esp),%ebx
3f5:   c3                      ret
3f6:   66 90                   xchg   %ax,%ax
3f8:   66 90                   xchg   %ax,%ax
3fa:   66 90                   xchg   %ax,%ax
3fc:   66 90                   xchg   %ax,%ax
3fe:   66 90                   xchg   %ax,%ax
```

objdump -d smain.out (8) __x86.get_pc_thunk.bx

```
00000400 <__x86.get_pc_thunk.bx>:
 400:   8b 1c 24                mov    (%esp),%ebx
 403:   c3                      ret
 404:   66 90                   xchg   %ax,%ax
 406:   66 90                   xchg   %ax,%ax
 408:   66 90                   xchg   %ax,%ax
 40a:   66 90                   xchg   %ax,%ax
 40c:   66 90                   xchg   %ax,%ax
 40e:   66 90                   xchg   %ax,%ax
```

objdump -d smain.out (9) deregister_tm_clones (a)

```
00000410 <deregister_tm_clones>:
 410:   e8 e4 00 00 00          call   4f9 <__x86.get_pc_thunk.dx>
 415:   81 c2 c7 1b 00 00       add    $0x1bc7,%edx
 41b:   8d 8a 38 00 00 00       lea    0x38(%edx),%ecx
 421:   8d 82 38 00 00 00       lea    0x38(%edx),%eax
 427:   39 c8                   cmp    %ecx,%eax
 429:   74 1d                   je     448 <deregister_tm_clones+0x38>
 42b:   8b 82 10 00 00 00       mov    0x10(%edx),%eax
 431:   85 c0                   test   %eax,%eax
 433:   74 13                   je     448 <deregister_tm_clones+0x38>
 435:   55                      push   %ebp
 436:   89 e5                   mov    %esp,%ebp
 438:   83 ec 14                sub    $0x14,%esp
 43b:   51                      push   %ecx
```

objdump -d smain.out (10) deregister_tm_clones (b)

```
43c:   ff d0                    call   *%eax
43e:   83 c4 10                 add    $0x10,%esp
441:   c9                       leave
442:   c3                       ret
443:   90                       nop
444:   8d 74 26 00             lea    0x0(%esi,%eiz,1),%esi
448:   f3 c3                    repz ret
44a:   8d b6 00 00 00 00        lea    0x0(%esi),%esi
```

objdump -d smain.out (11) register_tm_clones (a)

```
00000450 <register_tm_clones>:
 450:   e8 a4 00 00 00          call   4f9 <__x86.get_pc_thunk.dx>
 455:   81 c2 87 1b 00 00       add    $0x1b87,%edx
 45b:   55                      push   %ebp
 45c:   8d 8a 38 00 00 00       lea    0x38(%edx),%ecx
 462:   8d 82 38 00 00 00       lea    0x38(%edx),%eax
 468:   29 c8                   sub    %ecx,%eax
 46a:   89 e5                   mov    %esp,%ebp
 46c:   53                      push   %ebx
 46d:   c1 f8 02                sar    $0x2,%eax
 470:   89 c3                   mov    %eax,%ebx
 472:   83 ec 04                sub    $0x4,%esp
 475:   c1 eb 1f                shr    $0x1f,%ebx
 478:   01 d8                   add    %ebx,%eax
 47a:   d1 f8                   sar    %eax
```

objdump -d smain.out (12) register_tm_clones (b)

```
47c:   74 14                   je     492 <register_tm_clones+0x42>
47e:   8b 92 20 00 00 00       mov    0x20(%edx),%edx
484:   85 d2                   test   %edx,%edx
486:   74 0a                   je     492 <register_tm_clones+0x42>
488:   83 ec 08                sub    $0x8,%esp
48b:   50                      push   %eax
48c:   51                      push   %ecx
48d:   ff d2                   call   *%edx
48f:   83 c4 10                add    $0x10,%esp
492:   8b 5d fc                mov    -0x4(%ebp),%ebx
495:   c9                      leave
496:   c3                      ret
497:   89 f6                   mov    %esi,%esi
499:   8d bc 27 00 00 00 00    lea    0x0(%edi,%eiz,1),%edi
```

objdump -d smain.out (13) __do_global_dtors_aux

```
000004a0 <__do_global_dtors_aux>:
 4a0:   55                      push   %ebp
 4a1:   89 e5                   mov    %esp,%ebp
 4a3:   53                      push   %ebx
 4a4:   e8 57 ff ff ff          call   400 <__x86.get_pc_thunk.bx>
 4a9:   81 c3 33 1b 00 00       add    $0x1b33,%ebx
 4af:   83 ec 04                sub    $0x4,%esp
 4b2:   80 bb 38 00 00 00 00    cmpb   $0x0,0x38(%ebx)
 4b9:   75 27                   jne    4e2 <__do_global_dtors_aux+0x42>
 4bb:   8b 83 14 00 00 00       mov    0x14(%ebx),%eax
 4c1:   85 c0                   test   %eax,%eax
 4c3:   74 11                   je     4d6 <__do_global_dtors_aux+0x36>
 4c5:   83 ec 0c                sub    $0xc,%esp
 4c8:   ff b3 28 00 00 00       pushl  0x28(%ebx)
 4ce:   e8 dd fe ff ff          call   3b0 <__cxa_finalize@plt>
 4d3:   83 c4 10                add    $0x10,%esp
 4d6:   e8 35 ff ff ff          call   410 <deregister_tm_clones>
 4db:   c6 83 38 00 00 00 01    movb   $0x1,0x38(%ebx)
 4e2:   8b 5d fc                mov    -0x4(%ebp),%ebx
 4e5:   c9                      leave
 4e6:   c3                      ret
 4e7:   89 f6                   mov    %esi,%esi
 4e9:   8d bc 27 00 00 00 00    lea    0x0(%edi,%eiz,1),%edi
```

objdump -d smain.out (14) frame_dummy, __x86.get_pc_thunk.dx

```
000004f0 <frame_dummy>:
 4f0:   55                      push   %ebp
 4f1:   89 e5                   mov    %esp,%ebp
 4f3:   5d                      pop    %ebp
 4f4:   e9 57 ff ff ff          jmp    450 <register_tm_clones>

000004f9 <__x86.get_pc_thunk.dx>:
 4f9:   8b 14 24                mov    (%esp),%edx
 4fc:   c3                      ret
```

objdump -d smain.out (15) main

```
000004fd <main>:
 4fd:   8d 4c 24 04             lea    0x4(%esp),%ecx
 501:   83 e4 f0                and    $0xfffffff0,%esp
 504:   ff 71 fc                pushl  -0x4(%ecx)
 507:   55                      push   %ebp
 508:   89 e5                   mov    %esp,%ebp
 50a:   53                      push   %ebx
 50b:   51                      push   %ecx
 50c:   e8 18 00 00 00          call   529 <__x86.get_pc_thunk.ax>
 511:   05 cb 1a 00 00          add    $0x1acb,%eax
 516:   89 c3                   mov    %eax,%ebx
 518:   e8 10 00 00 00          call   52d <swap>
 51d:   b8 00 00 00 00          mov    $0x0,%eax
 522:   59                      pop    %ecx
 523:   5b                      pop    %ebx
 524:   5d                      pop    %ebp
 525:   8d 61 fc                lea    -0x4(%ecx),%esp
 528:   c3                      ret
```

objdump -d smain.out (16) __x86.get_pc_thunk.ax

```
00000529 <__x86.get_pc_thunk.ax>:
 529:   8b 04 24                mov    (%esp),%eax
 52c:   c3                      ret
```

objdump -d smain.out (17) swap (a)

```
0000052d <swap>:
 52d:   55                      push   %ebp
 52e:   89 e5                   mov    %esp,%ebp
 530:   83 ec 10                sub    $0x10,%esp
 533:   e8 f1 ff ff ff          call   529 <__x86.get_pc_thunk.ax>
 538:   05 a4 1a 00 00          add    $0x1aa4,%eax
 53d:   8d 90 3c 00 00 00       lea    0x3c(%eax),%edx
 543:   8d 88 2c 00 00 00       lea    0x2c(%eax),%ecx
 549:   8d 49 04                lea    0x4(%ecx),%ecx
 54c:   89 0a                   mov    %ecx,(%edx)
 54e:   8b 90 34 00 00 00       mov    0x34(%eax),%edx
 554:   8b 12                   mov    (%edx),%edx
 556:   89 55 fc                mov    %edx,-0x4(%ebp)
 559:   8d 90 3c 00 00 00       lea    0x3c(%eax),%edx
 55f:   8b 0a                   mov    (%edx),%ecx
 561:   8b 90 34 00 00 00       mov    0x34(%eax),%edx
 567:   8b 09                   mov    (%ecx),%ecx
 569:   89 0a                   mov    %ecx,(%edx)
 56b:   8d 80 3c 00 00 00       lea    0x3c(%eax),%eax
 571:   8b 00                   mov    (%eax),%eax
 573:   8b 55 fc                mov    -0x4(%ebp),%edx
 576:   89 10                   mov    %edx,(%eax)
```

objdump -d smain.out (18) swap (b)

```
578:   90                      nop
579:   c9                      leave
57a:   c3                      ret
57b:   66 90                   xchg   %ax,%ax
57d:   66 90                   xchg   %ax,%ax
57f:   90                      nop
```

objdump -d smain.out (19) __libc_csu_init (a)

```
00000580 <__libc_csu_init>:
 580:   55                      push   %ebp
 581:   57                      push   %edi
 582:   56                      push   %esi
 583:   53                      push   %ebx
 584:   e8 77 fe ff ff          call   400 <__x86.get_pc_thunk.bx>
 589:   81 c3 53 1a 00 00       add    $0x1a53,%ebx
 58f:   83 ec 0c                sub    $0xc,%esp
 592:   8b 6c 24 28             mov    0x28(%esp),%ebp
 596:   8d b3 04 ff ff ff       lea    -0xfc(%ebx),%esi
 59c:   e8 bf fd ff ff          call   360 <_init>
 5a1:   8d 83 00 ff ff ff       lea    -0x100(%ebx),%eax
 5a7:   29 c6                   sub    %eax,%esi
 5a9:   c1 fe 02                sar    $0x2,%esi
 5ac:   85 f6                   test   %esi,%esi
 5ae:   74 25                   je     5d5 <__libc_csu_init+0x55>
 5b0:   31 ff                   xor    %edi,%edi
 5b2:   8d b6 00 00 00 00       lea    0x0(%esi),%esi
 5b8:   83 ec 04                sub    $0x4,%esp
 5bb:   55                      push   %ebp
 5bc:   ff 74 24 2c             pushl  0x2c(%esp)
 5c0:   ff 74 24 2c             pushl  0x2c(%esp)
```

objdump -d smain.out (20) __libc_csu_init (b)

```
 5c4:   ff 94 bb 00 ff ff ff    call   *-0x100(%ebx,%edi,4)
 5cb:   83 c7 01                add    $0x1,%edi
 5ce:   83 c4 10                add    $0x10,%esp
 5d1:   39 fe                   cmp    %edi,%esi
 5d3:   75 e3                   jne    5b8 <__libc_csu_init+0x38>
 5d5:   83 c4 0c                add    $0xc,%esp
 5d8:   5b                      pop    %ebx
 5d9:   5e                      pop    %esi
 5da:   5f                      pop    %edi
 5db:   5d                      pop    %ebp
 5dc:   c3                      ret
 5dd:   8d 76 00                lea    0x0(%esi),%esi

000005e0 <__libc_csu_fini>:
 5e0:   f3 c3                   repz ret
```

readelf -s swap.o

```
Symbol table '.symtab' contains 17 entries:
   Num:    Valor  Tam  Tipo    Unión  Vis      Nombre Ind
     0: 00000000    0 NOTYPE  LOCAL  DEFAULT  UND
     1: 00000000    0 FILE    LOCAL  DEFAULT  ABS swap.c
     2: 00000000    0 SECTION LOCAL  DEFAULT    2
     3: 00000000    0 SECTION LOCAL  DEFAULT    4
     4: 00000000    0 SECTION LOCAL  DEFAULT    5
     5: 00000000    0 SECTION LOCAL  DEFAULT    6
     6: 00000000    0 SECTION LOCAL  DEFAULT    8
     7: 00000000    0 SECTION LOCAL  DEFAULT   10
     8: 00000000    0 SECTION LOCAL  DEFAULT   11
     9: 00000000    0 SECTION LOCAL  DEFAULT    9
    10: 00000000    0 SECTION LOCAL  DEFAULT    1
    11: 00000000    4 OBJECT  GLOBAL DEFAULT    6 p0
    12: 00000000    0 NOTYPE  GLOBAL DEFAULT  UND buf
    13: 00000004    4 OBJECT  GLOBAL DEFAULT  COM p1
    14: 00000000   78 FUNC    GLOBAL DEFAULT    2 swap
    15: 00000000    0 FUNC    GLOBAL HIDDEN     8 __x86.get_pc_thunk.ax
    16: 00000000    0 NOTYPE  GLOBAL DEFAULT  UND _GLOBAL_OFFSET_TABLE_
```

objdump -t swap.o

```
oung@USys2:~/smain$ objdump -t swap.o

swap.o:     formato del fichero elf32-i386

SYMBOL TABLE:
00000000 l    df *ABS*  00000000 swap.c
00000000 l    d  .text  00000000 .text
00000000 l    d  .data  00000000 .data
00000000 l    d  .bss   00000000 .bss
00000000 l    d  .data.rel      00000000 .data.rel
00000000 l    d  .text.__x86.get_pc_thunk.ax    00000000 .text.__x86.get_pc_thunk.a
00000000 l    d  .note.GNU-stack        00000000 .note.GNU-stack
00000000 l    d  .eh_frame      00000000 .eh_frame
00000000 l    d  .comment       00000000 .comment
00000000 l    d  .group 00000000 .group
00000000 g     O .data.rel      00000004 p0
00000000         *UND*  00000000 buf
00000004       O *COM*  00000004 p1
00000000 g     F .text  0000004e swap
00000000 g     F .text.__x86.get_pc_thunk.ax    00000000 .hidden __x86.get_pc_thunk
00000000         *UND*  00000000 _GLOBAL_OFFSET_TABLE_
```

readelf -s smain.o

```
young@USys2:~/smain$ readelf -s smain.o

Symbol table '.symtab' contains 15 entries:
   Num:    Valor  Tam  Tipo    Unión  Vis      Nombre Ind
     0: 00000000    0 NOTYPE  LOCAL  DEFAULT  UND
     1: 00000000    0 FILE    LOCAL  DEFAULT  ABS smain.c
     2: 00000000    0 SECTION LOCAL  DEFAULT    2
     3: 00000000    0 SECTION LOCAL  DEFAULT    4
     4: 00000000    0 SECTION LOCAL  DEFAULT    5
     5: 00000000    0 SECTION LOCAL  DEFAULT    6
     6: 00000000    0 SECTION LOCAL  DEFAULT    8
     7: 00000000    0 SECTION LOCAL  DEFAULT    9
     8: 00000000    0 SECTION LOCAL  DEFAULT    7
     9: 00000000    0 SECTION LOCAL  DEFAULT    1
    10: 00000000    8 OBJECT  GLOBAL DEFAULT    4 buf
    11: 00000000   44 FUNC    GLOBAL DEFAULT    2 main
    12: 00000000    0 FUNC    GLOBAL HIDDEN     6 __x86.get_pc_thunk.ax
    13: 00000000    0 NOTYPE  GLOBAL DEFAULT  UND _GLOBAL_OFFSET_TABLE_
    14: 00000000    0 NOTYPE  GLOBAL DEFAULT  UND swap
```

objdump -t smain.o

```
young@USys2:~/smain$ objdump -t smain.o

smain.o:     formato del fichero elf32-i386

SYMBOL TABLE:
00000000 l    df *ABS*  00000000 smain.c
00000000 l    d  .text  00000000 .text
00000000 l    d  .data  00000000 .data
00000000 l    d  .bss   00000000 .bss
00000000 l    d  .text.__x86.get_pc_thunk.ax    00000000 .text.__x86.get_pc_thunk.a
00000000 l    d  .note.GNU-stack        00000000 .note.GNU-stack
00000000 l    d  .eh_frame      00000000 .eh_frame
00000000 l    d  .comment       00000000 .comment
00000000 l    d  .group 00000000 .group
00000000 g     O .data  00000008 buf
00000000 g     F .text  0000002c main
00000000 g     F .text.__x86.get_pc_thunk.ax    00000000 .hidden __x86.get_pc_thunk
00000000         *UND*  00000000 _GLOBAL_OFFSET_TABLE_
00000000         *UND*  00000000 swap
```

readelf -s smain.out compiled with -nostdlib (1)

```
young@USys2:~/smain$ readelf -s smain.out

Symbol table '.dynsym' contains 1 entry:
   Num:    Valor  Tam  Tipo    Unión  Vis      Nombre Ind
     0: 00000000     0 NOTYPE  LOCAL  DEFAULT  UND

Symbol table '.symtab' contains 31 entries:
   Num:    Valor  Tam  Tipo    Unión  Vis      Nombre Ind
     0: 00000000     0 NOTYPE  LOCAL  DEFAULT  UND
     1: 00000154     0 SECTION LOCAL  DEFAULT    1
     2: 00000168     0 SECTION LOCAL  DEFAULT    2
     3: 0000018c     0 SECTION LOCAL  DEFAULT    3
     4: 000001a4     0 SECTION LOCAL  DEFAULT    4
     5: 000001b4     0 SECTION LOCAL  DEFAULT    5
     6: 000001b8     0 SECTION LOCAL  DEFAULT    6
     7: 000001c0     0 SECTION LOCAL  DEFAULT    7
     8: 00000240     0 SECTION LOCAL  DEFAULT    8
     9: 00000264     0 SECTION LOCAL  DEFAULT    9
    10: 00001f6c     0 SECTION LOCAL  DEFAULT   10
    11: 00001ff4     0 SECTION LOCAL  DEFAULT   11
    12: 00002000     0 SECTION LOCAL  DEFAULT   12
    13: 0000200c     0 SECTION LOCAL  DEFAULT   13
    14: 00000000     0 SECTION LOCAL  DEFAULT   14
```

readelf -s smain.out compiled with -nostdlib (2)

```
15: 00000000     0 FILE    LOCAL  DEFAULT  ABS smain.c
16: 00000000     0 FILE    LOCAL  DEFAULT  ABS swap.c
17: 00000000     0 FILE    LOCAL  DEFAULT  ABS
18: 00001f6c     0 OBJECT  LOCAL  DEFAULT   10 _DYNAMIC
19: 00000240     0 NOTYPE  LOCAL  DEFAULT    8 __GNU_EH_FRAME_HDR
20: 00001ff4     0 OBJECT  LOCAL  DEFAULT   11 _GLOBAL_OFFSET_TABLE_
21: 00002008     4 OBJECT  GLOBAL DEFAULT   12 p0
22: 000001f0    78 FUNC    GLOBAL DEFAULT    7 swap
23: 000001ec     0 FUNC    GLOBAL HIDDEN     7 __x86.get_pc_thunk.ax
24: 0000200c     4 OBJECT  GLOBAL DEFAULT   13 p1
25: 00000000     0 NOTYPE  GLOBAL DEFAULT  UND _start
26: 0000200c     0 NOTYPE  GLOBAL DEFAULT   13 __bss_start
27: 000001c0    44 FUNC    GLOBAL DEFAULT    7 main
28: 00002000     8 OBJECT  GLOBAL DEFAULT   12 buf
29: 0000200c     0 NOTYPE  GLOBAL DEFAULT   12 _edata
30: 00002010     0 NOTYPE  GLOBAL DEFAULT   13 _end
```

readelf -s smain.out (1)

```
young@USys2:~/smain$ readelf -s smain.out

Symbol table '.dynsym' contains 7 entries:
   Num:    Valor  Tam  Tipo    Unión  Vis      Nombre Ind
     0: 00000000    0 NOTYPE  LOCAL  DEFAULT  UND
     1: 00000000    0 NOTYPE  WEAK   DEFAULT  UND _ITM_deregisterTMCloneTab
     2: 00000000    0 FUNC    WEAK   DEFAULT  UND __cxa_finalize@GLIBC_2.1.3 (2)
     3: 00000000    0 NOTYPE  WEAK   DEFAULT  UND __gmon_start__
     4: 00000000    0 FUNC    GLOBAL DEFAULT  UND __libc_start_main@GLIBC_2.0 (3)
     5: 00000000    0 NOTYPE  WEAK   DEFAULT  UND _ITM_registerTMCloneTable
     6: 000005fc    4 OBJECT  GLOBAL DEFAULT   16 _IO_stdin_used

Symbol table '.symtab' contains 71 entries:
   Num:    Valor  Tam  Tipo    Unión  Vis      Nombre Ind
     0: 00000000    0 NOTYPE  LOCAL  DEFAULT  UND
     1: 00000154    0 SECTION LOCAL  DEFAULT    1
     2: 00000168    0 SECTION LOCAL  DEFAULT    2
     3: 00000188    0 SECTION LOCAL  DEFAULT    3
     4: 000001ac    0 SECTION LOCAL  DEFAULT    4
     5: 000001cc    0 SECTION LOCAL  DEFAULT    5
```

readelf -s smain.out (2)

```
 6: 0000023c     0 SECTION LOCAL  DEFAULT    6
 7: 000002d2     0 SECTION LOCAL  DEFAULT    7
 8: 000002e0     0 SECTION LOCAL  DEFAULT    8
 9: 00000310     0 SECTION LOCAL  DEFAULT    9
10: 00000358     0 SECTION LOCAL  DEFAULT   10
11: 00000360     0 SECTION LOCAL  DEFAULT   11
12: 00000390     0 SECTION LOCAL  DEFAULT   12
13: 000003b0     0 SECTION LOCAL  DEFAULT   13
14: 000003c0     0 SECTION LOCAL  DEFAULT   14
15: 000005e4     0 SECTION LOCAL  DEFAULT   15
16: 000005f8     0 SECTION LOCAL  DEFAULT   16
17: 00000600     0 SECTION LOCAL  DEFAULT   17
18: 00000644     0 SECTION LOCAL  DEFAULT   18
19: 00001edc     0 SECTION LOCAL  DEFAULT   19
20: 00001ee0     0 SECTION LOCAL  DEFAULT   20
21: 00001ee4     0 SECTION LOCAL  DEFAULT   21
22: 00001fdc     0 SECTION LOCAL  DEFAULT   22
23: 00002000     0 SECTION LOCAL  DEFAULT   23
24: 00002014     0 SECTION LOCAL  DEFAULT   24
25: 00000000     0 SECTION LOCAL  DEFAULT   25
```

readelf -s smain.out (3)

```
26: 00000000     0 FILE    LOCAL  DEFAULT  ABS crtstuff.c
27: 00000410     0 FUNC    LOCAL  DEFAULT   14 deregister_tm_clones
28: 00000450     0 FUNC    LOCAL  DEFAULT   14 register_tm_clones
29: 000004a0     0 FUNC    LOCAL  DEFAULT   14 __do_global_dtors_aux
30: 00002014     1 OBJECT  LOCAL  DEFAULT   24 completed.7281
31: 00001ee0     0 OBJECT  LOCAL  DEFAULT   20 __do_global_dtors_aux_fin
32: 000004f0     0 FUNC    LOCAL  DEFAULT   14 frame_dummy
33: 00001edc     0 OBJECT  LOCAL  DEFAULT   19 __frame_dummy_init_array_
34: 00000000     0 FILE    LOCAL  DEFAULT  ABS smain.c
35: 00000000     0 FILE    LOCAL  DEFAULT  ABS swap.c
36: 00000000     0 FILE    LOCAL  DEFAULT  ABS crtstuff.c
37: 0000075c     0 OBJECT  LOCAL  DEFAULT   18 __FRAME_END__
38: 00000000     0 FILE    LOCAL  DEFAULT  ABS
39: 00001ee0     0 NOTYPE  LOCAL  DEFAULT   19 __init_array_end
40: 00001ee4     0 OBJECT  LOCAL  DEFAULT   21 _DYNAMIC
41: 00001edc     0 NOTYPE  LOCAL  DEFAULT   19 __init_array_start
42: 00000600     0 NOTYPE  LOCAL  DEFAULT   17 __GNU_EH_FRAME_HDR
43: 00001fdc     0 OBJECT  LOCAL  DEFAULT   22 _GLOBAL_OFFSET_TABLE_
44: 000005e0     2 FUNC    GLOBAL DEFAULT   14 __libc_csu_fini
45: 00000000     0 NOTYPE  WEAK   DEFAULT  UND _ITM_deregisterTMCloneTab
46: 00000400     4 FUNC    GLOBAL HIDDEN    14 __x86.get_pc_thunk.bx
47: 00002000     0 NOTYPE  WEAK   DEFAULT   23 data_start
```

readelf -s smain.out (4)

```
48: 00002014     0 NOTYPE  GLOBAL DEFAULT   23 _edata
49: 00002010     4 OBJECT  GLOBAL DEFAULT   23 p0
50: 000005e4     0 FUNC    GLOBAL DEFAULT   15 _fini
51: 000004f9     0 FUNC    GLOBAL HIDDEN    14 __x86.get_pc_thunk.dx
52: 00000000     0 FUNC    WEAK   DEFAULT  UND __cxa_finalize@@GLIBC_2.1
53: 00002000     0 NOTYPE  GLOBAL DEFAULT   23 __data_start
54: 00000000     0 NOTYPE  WEAK   DEFAULT  UND __gmon_start__
55: 00002004     0 OBJECT  GLOBAL HIDDEN    23 __dso_handle
56: 000005fc     4 OBJECT  GLOBAL DEFAULT   16 _IO_stdin_used
57: 00000000     0 FUNC    GLOBAL DEFAULT  UND __libc_start_main@@GLIBC_
58: 00000580    93 FUNC    GLOBAL DEFAULT   14 __libc_csu_init
59: 0000201c     0 NOTYPE  GLOBAL DEFAULT   24 _end
60: 000003c0     0 FUNC    GLOBAL DEFAULT   14 _start
61: 000005f8     4 OBJECT  GLOBAL DEFAULT   16 _fp_hw
62: 00002008     8 OBJECT  GLOBAL DEFAULT   23 buf
63: 00002014     0 NOTYPE  GLOBAL DEFAULT   24 __bss_start
64: 000004fd    44 FUNC    GLOBAL DEFAULT   14 main
65: 00000529     0 FUNC    GLOBAL HIDDEN    14 __x86.get_pc_thunk.ax
66: 00002018     4 OBJECT  GLOBAL DEFAULT   24 p1
67: 00002014     0 OBJECT  GLOBAL HIDDEN    23 __TMC_END__
68: 00000000     0 NOTYPE  WEAK   DEFAULT  UND _ITM_registerTMCloneTable
69: 0000052d    78 FUNC    GLOBAL DEFAULT   14 swap
70: 00000360     0 FUNC    GLOBAL DEFAULT   11 _init
```

objdump -t smain.out (1)

```
young@USys2:~/smain$ objdump -t smain.out

smain.out:     formato del fichero elf32-i386

SYMBOL TABLE:
00000154 l    d  .interp        00000000              .interp
00000168 l    d  .note.ABI-tag  00000000              .note.ABI-tag
00000188 l    d  .note.gnu.build-id     00000000              .note.gnu.build-id
000001ac l    d  .gnu.hash      00000000              .gnu.hash
000001cc l    d  .dynsym        00000000              .dynsym
0000023c l    d  .dynstr        00000000              .dynstr
000002d2 l    d  .gnu.version   00000000              .gnu.version
000002e0 l    d  .gnu.version_r 00000000              .gnu.version_r
00000310 l    d  .rel.dyn       00000000              .rel.dyn
00000358 l    d  .rel.plt       00000000              .rel.plt
00000360 l    d  .init  00000000              .init
00000390 l    d  .plt   00000000              .plt
000003b0 l    d  .plt.got       00000000              .plt.got
000003c0 l    d  .text  00000000              .text
000005e4 l    d  .fini  00000000              .fini
000005f8 l    d  .rodata        00000000              .rodata
00000600 l    d  .eh_frame_hdr  00000000              .eh_frame_hdr
00000644 l    d  .eh_frame      00000000              .eh_frame
```

objdump -t smain.out (2)

```
00001edc l    d  .init_array    00000000              .init_array
00001ee0 l    d  .fini_array    00000000              .fini_array
00001ee4 l    d  .dynamic       00000000              .dynamic
00001fdc l    d  .got   00000000              .got
00002000 l    d  .data  00000000              .data
00002014 l    d  .bss   00000000              .bss
00000000 l    d  .comment       00000000              .comment
00000000 l    df *ABS*  00000000              crtstuff.c
00000410 l     F .text  00000000              deregister_tm_clones
00000450 l     F .text  00000000              register_tm_clones
000004a0 l     F .text  00000000              __do_global_dtors_aux
00002014 l     O .bss   00000001              completed.7281
00001ee0 l     O .fini_array    00000000              __do_global_dtors_aux_fini_ar
000004f0 l     F .text  00000000              frame_dummy
00001edc l     O .init_array    00000000              __frame_dummy_init_array_entr
00000000 l    df *ABS*  00000000              smain.c
00000000 l    df *ABS*  00000000              swap.c
00000000 l    df *ABS*  00000000              crtstuff.c
0000075c l     O .eh_frame      00000000              __FRAME_END__
00000000 l    df *ABS*  00000000
00001ee0 l        .init_array    00000000              __init_array_end
00001ee4 l     O .dynamic       00000000              _DYNAMIC
```

objdump -t smain.out (3)

```
00001edc l       .init_array    00000000              __init_array_start
00000600 l       .eh_frame_hdr  00000000              __GNU_EH_FRAME_HDR
00001fdc l     O .got   00000000                      _GLOBAL_OFFSET_TABLE_
000005e0 g     F .text  00000002                      __libc_csu_fini
00000000  w      *UND*  00000000                      _ITM_deregisterTMCloneTable
00000400 g     F .text  00000004                      .hidden __x86.get_pc_thunk.bx
00002000  w      .data  00000000                      data_start
00002014 g       .data  00000000                      _edata
00002010 g     O .data  00000004                      p0
000005e4 g     F .fini  00000000                      _fini
000004f9 g     F .text  00000000                      .hidden __x86.get_pc_thunk.dx
00000000  w    F *UND*  00000000                      __cxa_finalize@@GLIBC_2.1.3
00002000 g       .data  00000000                      __data_start
00000000  w      *UND*  00000000                      __gmon_start__
00002004 g     O .data  00000000                      .hidden __dso_handle
000005fc g     O .rodata        00000004              _IO_stdin_used
00000000       F *UND*  00000000                      __libc_start_main@@GLIBC_2.0
00000580 g     F .text  0000005d                      __libc_csu_init
0000201c g       .bss   00000000                      _end
000003c0 g     F .text  00000000                      _start
000005f8 g     O .rodata        00000004              _fp_hw
00002008 g     O .data  00000008                      buf
00002014 g       .bss   00000000                      __bss_start
```

objdump -t smain.out (4)

```
000004fd g     F .text  0000002c              main
00000529 g     F .text  00000000              .hidden __x86.get_pc_thunk.ax
00002018 g     O .bss   00000004              p1
00002014 g     O .data  00000000              .hidden __TMC_END__
00000000  w      *UND*  00000000              _ITM_registerTMCloneTable
0000052d g     F .text  0000004e              swap
00000360 g     F .init  00000000              _init
```