Link Example 3.A Position Independent Code - Examples

Young W. Lim

2019-01-21 Mon

Outline

Based on

1. http://bottomupcs.sourceforge.net/csbu/x3824.htm
2. https://www.technovelty.org/linux/plt-and-got-the-key-to-code-sharing-and-dynamic-libraries.h

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

GOT example source codes

```
//======= got.c ====================
extern int i;

void test(void)
{
    i = 100;
}
```

```
$ gcc -nostdlib  -shared -fPIC -m32 -o got.so ./got.c
$ objdump --disassemble -m i386 ./got.so
$ readelf --sections ./got.so
$ readelf --relocs ./got.so
```

http://bottomupcs.sourceforge.net/csbu/x3824.htm

objdump - disassembly of .text (32-bit)

```
./got.so:     formato del fichero elf32-i386

Desensamblado de la sección .text:

000001f0 <test>:
 1f0: 55                          push   %ebp
 1f1: 89 e5                       mov    %esp,%ebp
 1f3: e8 14 00 00 00              call   20c <__x86.get_pc_thunk.ax>
 1f8: 05 08 1e 00 00              add    $0x1e08,%eax
 1fd: 8b 80 fc ff ff ff           mov    -0x4(%eax),%eax
 203: c7 00 64 00 00 00           movl   $0x64,(%eax)
 209: 90                          nop
 20a: 5d                          pop    %ebp
 20b: c3                          ret

0000020c <__x86.get_pc_thunk.ax>:
 20c: 8b 04 24                    mov    (%esp),%eax
 20f: c3                          ret
```

readelf - sections (32-bit)

```
There are 16 section headers, starting at offset 0x12b4:

Encabezados de Sección:
  [Nr] Nombre              Tipo            Direc    Desp   Tam    ES Opt En Inf Al
  [ 0]                     NULL            00000000 000000 000000 00      0   0  0
  [ 1] .note.gnu.build-i NOTE            00000114 000114 000024 00   A  0   0  4
  [ 2] .gnu.hash           GNU_HASH        00000138 000138 000030 04   A  3   0  4
  [ 3] .dynsym             DYNSYM          00000168 000168 000060 10   A  4   1  4
  [ 4] .dynstr             STRTAB          000001c8 0001c8 000020 00   A  0   0  1
  [ 5] .rel.dyn            REL             000001e8 0001e8 000008 08   A  3   0  4
  [ 6] .text               PROGBITS        000001f0 0001f0 000020 00  AX  0   0  1
  [ 7] .eh_frame_hdr       PROGBITS        00000210 000210 00001c 00   A  0   0  4
  [ 8] .eh_frame           PROGBITS        0000022c 00022c 00004c 00   A  0   0  4
  [ 9] .dynamic            DYNAMIC         00001f8c 000f8c 000070 08  WA  4   0  4
  [10] .got                PROGBITS        00001ffc 000ffc 000004 04  WA  0   0  4
  [11] .got.plt            PROGBITS        00002000 001000 00000c 04  WA  0   0  4
  [12] .comment            PROGBITS        00000000 00100c 00002a 01  MS  0   0  1
  [13] .symtab             SYMTAB          00000000 001038 000180 10     14  19  4
  [14] .strtab             STRTAB          00000000 0011b8 00006c 00      0   0  1
  [15] .shstrtab           STRTAB          00000000 001224 00008f 00      0   0  1
Key to Flags:
  W (write), A (alloc), X (execute), M (merge), S (strings), I (info),
  L (link order), O (extra OS processing required), G (group), T (TLS),
  C (compressed), x (unknown), o (OS specific), E (exclude),
```

readelf - .got section address (32-bit)

```
[10] .got              PROGBITS        00001ffc 000ffc 000004 04   WA  0   0  4
[11] .got.plt          PROGBITS        00002000 001000 00000c 04   WA  0   0  4
```

- memory address 00001ffc
- file offset 000ffc
- size 4 (32-bit address)
- entry size 4 (32-bit address)
- %eax points to the .got section (0x1ffc)

Section Header Table Fields

- Addr (`sh_addr`)
  if the section will appear in the memory image of a process,
  this number give the address at which the section's first byte
  should reside. Otherwise the member contains 0.
- Off (`sh_offset`)
  the byte offset from the beginning of the file
  to the section's first byte
- Size (`sh_size`)
  this member gives the section's size in bytes
- ES (`sh_entsize`)
  for a table of fixed-size entries (eg. a symbol table)
  this member give the size in bytes of each fixed-size entry

`http://www.cs.cmu.edu/afs/cs/academic/class/15213-s00/doc/elf.pdf`

accessing the external symbol i

- the shared library got.so refers to an external symbol i
- *at compile time*, the symbol address is not known,
- *at runtime*, the dynamic linker will fix up the address
- a *sharable* code can be accessed simultaneously by many other processes.
- %eax points to the memory address where the .got section is loaded symbol entry address
- at this address, the absolute address of the symbol i will be stored symbol entry value

http://bottomupcs.sourceforge.net/csbu/x3824.htm

.got section address

- the `readelf` output shows that
  the `.got` section starts `0x1ffc` bytes past from
  where the `got.so` library is loaded into memory.

- if the shared object `got.so` were to be loaded into memory
  at `0x40000000`
- the `.got` (global object table section) would start
  at `0x40001ffc`,
- register %eax points to this address.
- `[10] .got PROGBITS 00001ffc 000ffc 000004 04 WA 0 0 4`

`http://bottomupcs.sourceforge.net/csbu/x3824.htm`

calculating the address of the symbol i

```
[10] .got              PROGBITS        00001ffc 000ffc 000004 04  WA  0   0  4
```

```
0x1ffc
```

```
1f3:  e8 14 00 00 00          call   20c <__x86.get_pc_thunk.ax>
1f8:  05 08 1e 00 00          add    $0x1e08,%eax
1fd:  8b 80 fc ff ff ff       mov    -0x4(%eax),%eax
203:  c7 00 64 00 00 00       movl   $0x64,(%eax)
```

```
%eax = 1f4 (pc)
%eax = 1f4 + 1e08 = 1ffc
```

assigning i=100

- store the value 100 (0x64) into the memory address held in register %eax
- ```
  mov    -0x4(%eax),%eax
  movl   $0x64,(%eax)
  ```

```
%eax = M[1ffc] - 4    ..... the absolute symbol address
M[ M[1ffc]-4 ] = 100 ..... the symbol value 100
```

http://bottomupcs.sourceforge.net/csbu/x3824.htm
```

got entry

- The got is simply a list of entries,
  one entry for each external (global) variable
- the got entry for the external variable `i`
  stores a 4 byte address (32 bit addresses)
- in this example,
  - the `.got` starts at offset `0x1ffc`
  - only one symbol `i` exists

`http://bottomupcs.sourceforge.net/csbu/x3824.htm`

relocation

```
La sección de reubicación '.rel.dyn' at offset 0x1e8 contains 1 entry:
 Desplaz    Info    Tipo            Val.Símbolo Nom. Símbolo
00001ffc  00000106 R_386_GLOB_DAT    00000000   i
```

- the relocation process replaces the value at offset `1ffc` with the address where the symbol `i` is stored at
- before the program begins, the dynamic linker will have *fixed up the relocation* to ensure that the value of the memory at offset `0x1ffc` is the address of the global variable `i`

Relocation Table

- The relocation table specifies
  where relocations are needed
- normally symbol relocations
- the dynamic linker has to resolve
  the needed symbol by its name,
  and then write the symbol address
  to the place specified in the relocation entry.

`https://greek0.net/elf.html`

Relocation Table Fields

- Offset (a section offest / a virtual address)
- Info (a symbol table index)
- Type (a relocation type)
- Sym. Value
- Sym. Name
- Addened

`https://greek0.net/elf.html`

Relocation Offset

- the location where to apply the relocation action.
  - For a relocatable file,
    the value indicates a section offset
  - For an executable or shared object,
    the value indicates the virtual address of the storage unit
    affected by the relocation.

`https://docs.oracle.com/cd/E23824_01/html/819-0690/chapter6-54839.html`

Relocation Info

- gives both the symbol table index, with respect to which the relocation must be made, and the type of relocation to apply.
- a call instruction's relocation entry holds the symbol table index of the function being called.
- if the index is STN_UNDEF, the undefined symbol index, the relocation uses zero as the symbol value.

https://docs.oracle.com/cd/E23824_01/html/819-0690/chapter6-54839.html

Relocation Types

- Relocation types are architecture specific
- On i386 the most important ones are
  - the R_386_COPY type
    just copy the address of the symbol to that address
  - the R_386_JUMP_SLOT
    used for the normal PLT/GOT function call relocation mechanism.

`https://greek0.net/elf.html`

GOT example - disassembly of .text (bottomupcs)

```
$ objdump --disassemble ./got.so

./got.so:     file format elf64-ia64-little

Disassembly of section .text:

0000000000000410 <test>:
 410:   0d 10 00 18 00 21       [MFI]       mov r2=r12
 416:   00 00 00 02 00 c0                   nop.f 0x0
 41c:   81 09 00 90                         addl r14=24,r1;;
 420:   0d 78 00 1c 18 10       [MFI]       ld8 r15=[r14]
 426:   00 00 00 02 00 c0                   nop.f 0x0
 42c:   41 06 00 90                         mov r14=100;;
 430:   11 00 38 1e 90 11       [MIB]       st4 [r15]=r14
 436:   c0 00 08 00 42 80                   mov r12=r2
 43c:   08 00 84 00                         br.ret.sptk.many b0;;
```

http://bottomupcs.sourceforge.net/csbu/x3824.htm

GOT example - sections (1) (bottomupcs)

```
$ readelf --sections ./got.so
There are 17 section headers, starting at offset 0x640:

  Section Headers:
  [Nr] Name              Type             Address           Offset
       Size              EntSize          Flags  Link  Info  Align
  [ 0]                   NULL             0000000000000000  00000000
       0000000000000000  0000000000000000           0     0     0
  [ 1] .hash             HASH             0000000000000120  00000120
       00000000000000a0  0000000000000004   A       2     0     8
  [ 2] .dynsym           DYNSYM           00000000000001c0  000001c0
       00000000000001f8  0000000000000018   A       3     e     8
  [ 3] .dynstr           STRTAB           00000000000003b8  000003b8
       000000000000003f  0000000000000000   A       0     0     1
  [ 4] .rela.dyn         RELA             00000000000003f8  000003f8
       0000000000000018  0000000000000018   A       2     0     8
  [ 5] .text             PROGBITS         0000000000000410  00000410
       0000000000000030  0000000000000000  AX       0     0     16
  [ 6] .IA_64.unwind_inf PROGBITS         0000000000000440  00000440
       0000000000000018  0000000000000000   A       0     0     8
  [ 7] .IA_64.unwind     IA_64_UNWIND     0000000000000458  00000458
       0000000000000018  0000000000000000  AL       5     5     8
```

http://bottomupcs.sourceforge.net/csbu/x3824.htm

GOT example - sections (2) (bottomupcs)

```
  [ 8] .data             PROGBITS         0000000000010470  00000470
       0000000000000000  0000000000000000  WA       0     0     1
  [ 9] .dynamic          DYNAMIC          0000000000010470  00000470
       0000000000000100  0000000000000010  WA       3     0     8
  [10] .got              PROGBITS         0000000000010570  00000570
       0000000000000020  0000000000000000 WAp       0     0     8
  [11] .sbss             NOBITS           0000000000010590  00000590
       0000000000000000  0000000000000000  W        0     0     1
  [12] .bss              NOBITS           0000000000010590  00000590
       0000000000000000  0000000000000000  WA       0     0     1
  [13] .comment          PROGBITS         0000000000000000  00000590
       0000000000000026  0000000000000000                   0     0     1
  [14] .shstrtab         STRTAB           0000000000000000  000005b6
       000000000000008a  0000000000000000                   0     0     1
  [15] .symtab           SYMTAB           0000000000000000  00000a80
       0000000000000258  0000000000000018          16    12     8
  [16] .strtab           STRTAB           0000000000000000  00000cd8
       0000000000000045  0000000000000000                   0     0     1
Key to Flags:
  W (write), A (alloc), X (execute), M (merge), S (strings)
  I (info), L (link order), G (group), x (unknown)
  O (extra OS processing required) o (OS specific), p (processor specific);
```

http://bottomupcs.sourceforge.net/csbu/x3824.htm

GOT example - disassembly of .text (64-bit)

```
objdump --disassemble got.so

got.so:     file format elf64-x86-64


Disassembly of section .text:

00000000000002f0 <test>:
 2f0: 55                          push   %rbp
 2f1: 48 89 e5                    mov    %rsp,%rbp
 2f4: 48 8b 05 fd 0c 20 00        mov    0x200cfd(%rip),%rax  # 200ff8 <i>
 2fb: c7 00 64 00 00 00           movl   $0x64,(%rax)         # i=100
 301: 90                          nop
 302: 5d                          pop    %rbp
 303: c3                          retq
```

GOT example - sections (64-bit) (1)

```
There are 16 section headers, starting at offset 0x1358:

Section Headers:
  [Nr] Name              Type             Address           Offset
       Size              EntSize          Flags  Link  Info  Align
  [ 0]                   NULL             0000000000000000  00000000
       0000000000000000  0000000000000000           0     0     0
  [ 1] .note.gnu.build-i NOTE             00000000000001c8  000001c8
       0000000000000024  0000000000000000   A       0     0     4
  [ 2] .gnu.hash         GNU_HASH         00000000000001f0  000001f0
       0000000000000034  0000000000000000   A       3     0     8
  [ 3] .dynsym           DYNSYM           0000000000000228  00000228
       0000000000000090  0000000000000018   A       4     1     8
  [ 4] .dynstr           STRTAB           00000000000002b8  000002b8
       0000000000000020  0000000000000000   A       0     0     1
  [ 5] .rela.dyn         RELA             00000000000002d8  000002d8
       0000000000000018  0000000000000018   A       3     0     8
  [ 6] .text             PROGBITS         00000000000002f0  000002f0
       0000000000000014  0000000000000000  AX       0     0     1
  [ 7] .eh_frame_hdr     PROGBITS         0000000000000304  00000304
       0000000000000014  0000000000000000   A       0     0     4
```

GOT example - sections (64-bit) (2)

```
  [ 8] .eh_frame         PROGBITS         0000000000000318  00000318
       0000000000000038  0000000000000000   A       0     0     8
  [ 9] .dynamic          DYNAMIC          0000000000200f18  00000f18
       00000000000000e0  0000000000000010  WA       4     0     8
  [10] .got              PROGBITS         0000000000200ff8  00000ff8
       0000000000000008  0000000000000008  WA       0     0     8
  [11] .got.plt          PROGBITS         0000000000201000  00001000
       0000000000000018  0000000000000008  WA       0     0     8
  [12] .comment          PROGBITS         0000000000000000  00001018
       000000000000002a  0000000000000001  MS       0     0     1
  [13] .symtab           SYMTAB           0000000000000000  00001048
       0000000000000228  0000000000000018          14    18     8
  [14] .strtab           STRTAB           0000000000000000  00001270
       0000000000000058  0000000000000000           0     0     1
  [15] .shstrtab         STRTAB           0000000000000000  000012c8
       0000000000000090  0000000000000000           0     0     1
Key to Flags:
  W (write), A (alloc), X (execute), M (merge), S (strings), I (info),
  L (link order), O (extra OS processing required), G (group), T (TLS),
  C (compressed), x (unknown), o (OS specific), E (exclude),
  l (large), p (processor specific)
```

GOT example - sections (64-bit) (3)

```
There are 16 section headers, starting at offset 0x1358:

Section Headers:
  [Nr] Name              Type             Address           Offset
       Size              EntSize          Flags  Link  Info  Align
  [ ]                    NULL

  [ 1] .note.gnu.build-i NOTE                              1c8       1c8
                    24                       A                      4
  [ 2] .gnu.hash         GNU_HASH                          1f0       1f0
                    34                       A       3              8
  [ 3] .dynsym           DYNSYM                            228       228
                    90                  18   A       4     1        8
  [ 4] .dynstr           STRTAB                            2b8       2b8
                    20                       A                      1
  [ 5] .rela.dyn         RELA                              2d8       2d8
                    18                  18   A       3              8
  [ 6] .text             PROGBITS                          2f0       2f0
                    14                       AX                     1
  [ 7] .eh_frame_hdr     PROGBITS                          304       304
                    14                       A                      4
```

GOT example - sections (64-bit) (4)

```
  [ 8] .eh_frame         PROGBITS                 318       318
                   38                   A                   8
  [ 9] .dynamic          DYNAMIC                 200f18      f18
                   e0                10  WA       4           8
  [10] .got              PROGBITS                200ff8      ff8
                    8                 8  WA                   8
  [11] .got.plt          PROGBITS                201000     1000
                   18                 8  WA                   8
  [12] .comment          PROGBITS                    0     1018
                   2a                 1  MS                   1
  [13] .symtab           SYMTAB                      0     1048
                  228                18          14  18       8
  [14] .strtab           STRTAB                      0     1270
                   58                                        1
  [15] .shstrtab         STRTAB                      0     12c8
                   90                                        1
Key to Flags:
  W (write), A (alloc), X (execute), M (merge), S (strings), I (info),
  L (link order), O (extra OS processing required), G (group), T (TLS),
  C (compressed), x (unknown), o (OS specific), E (exclude),
  l (large), p (processor specific)
```

GOT example - relocation (64-bit)

```
Relocation section '.rela.dyn' at offset 0x2d8 contains 1 entry:
  Offset          Info           Type           Sym. Value    Sym. Name + Addend
000000200ff8  000100000006 R_X86_64_GLOB_DAT 0000000000000000 i + 0

      200ff8     100000006 R_X86_64_GLOB_DAT                0 i + 0
```

readelf -headers /bin/ls (1)

```
Encabezado ELF:
  Mágico:   7f 45 4c 46 02 01 01 00 00 00 00 00 00 00 00 00
  Clase:                             ELF64
  Datos:                             complemento a 2, little endian
  Versión:                           1 (current)
  OS/ABI:                            UNIX - System V
  Versión ABI:                       0
  Tipo:                              DYN (Fichero objeto compartido)
  Máquina:                           Advanced Micro Devices X86-64
  Versión:                           0x1
  Dirección del punto de entrada:               0x5850
  Inicio de encabezados de programa:          64 (bytes en el fichero)
  Inicio de encabezados de sección:         132000 (bytes en el fichero)
  Opciones:                          0x0
  Tamaño de este encabezado:         64 (bytes)
  Tamaño de encabezados de programa: 56 (bytes)
  Número de encabezados de programa: 9
  Tamaño de encabezados de sección:  64 (bytes)
  Número de encabezados de sección:  28
  Índice de tabla de cadenas de sección de encabezado: 27
```

https://www.technovelty.org/linux/plt-and-got-the-key-to-code-sharing-and-dynamic-l

readelf -headers /bin/ls (2)

```
Encabezados de Sección:
  [Nr] Nombre              Tipo            Dirección          Despl
       Tamaño              TamEnt          Opts   Enl    Info  Alin
  [ 0]                     NULL            0000000000000000  00000000
       0000000000000000    0000000000000000           0     0     0
  [ 1] .interp             PROGBITS        0000000000000238  00000238
       000000000000001c    0000000000000000  A        0     0     1
  [ 2] .note.ABI-tag       NOTE            0000000000000254  00000254
       0000000000000020    0000000000000000  A        0     0     4
  [ 3] .note.gnu.build-i   NOTE            0000000000000274  00000274
       0000000000000024    0000000000000000  A        0     0     4
  [ 4] .gnu.hash           GNU_HASH        0000000000000298  00000298
       00000000000000ec    0000000000000000  A        5     0     8
  [ 5] .dynsym             DYNSYM          0000000000000388  00000388
       0000000000000df8    0000000000000018  A        6     1     8
  [ 6] .dynstr             STRTAB          0000000000001180  00001180
       0000000000000682    0000000000000000  A        0     0     1
  [ 7] .gnu.version        VERSYM          0000000000001802  00001802
       000000000000012a    0000000000000002  A        5     0     2
  [ 8] .gnu.version_r      VERNEED         0000000000001930  00001930
       0000000000000070    0000000000000000  A        6     1     8
```

https://www.technovelty.org/linux/plt-and-got-the-key-to-code-sharing-and-dynamic-l

readelf -headers /bin/ls (3)

```
[ 9] .rela.dyn          RELA             00000000000019a0  000019a0
     0000000000001350   0000000000000018   A       5     0     8
[10] .rela.plt          RELA             0000000000002cf0  00002cf0
     0000000000000a68   0000000000000018  AI       5    23     8
[11] .init              PROGBITS         0000000000003758  00003758
     0000000000000017   0000000000000000  AX       0     0     4
[12] .plt               PROGBITS         0000000000003770  00003770
     0000000000000700   0000000000000010  AX       0     0     16
[13] .plt.got           PROGBITS         0000000000003e70  00003e70
     0000000000000018   0000000000000008  AX       0     0     8
[14] .text              PROGBITS         0000000000003e90  00003e90
     00000000000124d9   0000000000000000  AX       0     0     16
[15] .fini              PROGBITS         000000000001636c  0001636c
     0000000000000009   0000000000000000  AX       0     0     4
[16] .rodata            PROGBITS         0000000000016380  00016380
     0000000000004e1d   0000000000000000   A       0     0     32
[17] .eh_frame_hdr      PROGBITS         000000000001b1a0  0001b1a0
     0000000000000884   0000000000000000   A       0     0     4
[18] .eh_frame          PROGBITS         000000000001ba28  0001ba28
     0000000000002cc0   0000000000000000   A       0     0     8
[19] .init_array        INIT_ARRAY       000000000021eff0  0001eff0
     0000000000000008   0000000000000008  WA       0     0     8
```

https://www.technovelty.org/linux/plt-and-got-the-key-to-code-sharing-and-dynamic-l

readelf -headers /bin/ls (4)

```
  [20] .fini_array       FINI_ARRAY       0000000000021eff8  0001eff8
       0000000000000008  0000000000000008  WA       0     0     8
  [21] .data.rel.ro      PROGBITS         0000000000021f000  0001f000
       0000000000000a38  0000000000000000  WA       0     0     32
  [22] .dynamic          DYNAMIC          0000000000021fa38  0001fa38
       0000000000000200  0000000000000010  WA       6     0     8
  [23] .got              PROGBITS         0000000000021fc38  0001fc38
       00000000000003c8  0000000000000008  WA       0     0     8
  [24] .data             PROGBITS         0000000000220000  00020000
       0000000000000268  0000000000000000  WA       0     0     32
  [25] .bss              NOBITS           0000000000220280  00020268
       00000000000012e0  0000000000000000  WA       0     0     32
  [26] .gnu_debuglink    PROGBITS         0000000000000000  00020268
       0000000000000034  0000000000000000           0     0     4
  [27] .shstrtab         STRTAB           0000000000000000  0002029c
       0000000000000101  0000000000000000           0     0     1
Key to Flags:
  W (write), A (alloc), X (execute), M (merge), S (strings), I (info),
  L (link order), O (extra OS processing required), G (group), T (TLS),
  C (compressed), x (unknown), o (OS specific), E (exclude),
  l (large), p (processor specific)
```

https://www.technovelty.org/linux/plt-and-got-the-key-to-code-sharing-and-dynamic-l

readelf -headers /bin/ls (5)

```
Encabezados de Programa:
  Tipo           Desplazamiento     DirVirtual          DirFísica
                 TamFichero         TamMemoria           Opts    Alineación
  PHDR           0x0000000000000040 0x0000000000000040 0x0000000000000040
                 0x00000000000001f8 0x00000000000001f8  R E    0x8
  INTERP         0x0000000000000238 0x0000000000000238 0x0000000000000238
                 0x000000000000001c 0x000000000000001c  R      0x1
      [Requesting program interpreter: /lib64/ld-linux-x86-64.so.2]
  LOAD           0x0000000000000000 0x0000000000000000 0x0000000000000000
                 0x000000000001e6e8 0x000000000001e6e8  R E    0x200000
  LOAD           0x000000000001eff0 0x000000000021eff0 0x000000000021eff0
                 0x0000000000001278 0x0000000000002570  RW     0x200000
  DYNAMIC        0x000000000001fa38 0x000000000021fa38 0x000000000021fa38
                 0x0000000000000200 0x0000000000000200  RW     0x8
  NOTE           0x0000000000000254 0x0000000000000254 0x0000000000000254
                 0x0000000000000044 0x0000000000000044  R      0x4
  GNU_EH_FRAME   0x000000000001b1a0 0x000000000001b1a0 0x000000000001b1a0
                 0x0000000000000884 0x0000000000000884  R      0x4
  GNU_STACK      0x0000000000000000 0x0000000000000000 0x0000000000000000
                 0x0000000000000000 0x0000000000000000  RW     0x10
  GNU_RELRO      0x000000000001eff0 0x000000000021eff0 0x000000000021eff0
                 0x0000000000001010 0x0000000000001010  R      0x1
```

https://www.technovelty.org/linux/plt-and-got-the-key-to-code-sharing-and-dynamic-l

readelf -headers /bin/ls (6)

```
mapeo de Sección a Segmento:
 Segmento Secciones...
  00
  01     .interp
  02     .interp .note.ABI-tag .note.gnu.build-id .gnu.hash .dynsym .dynstr .gnu.v
  03     .init_array .fini_array .data.rel.ro .dynamic .got .data .bss
  04     .dynamic
  05     .note.ABI-tag .note.gnu.build-id
  06     .eh_frame_hdr
  07
  08     .init_array .fini_array .data.rel.ro .dynamic .got
```

https://www.technovelty.org/linux/plt-and-got-the-key-to-code-sharing-and-dynamic-l

readelf -headers /bin/ls vs /lib/libc.so.6

```
young@USys2:~$ readelf --headers /bin/ls
  LOAD           0x0000000000000000 0x0000000000000000 0x0000000000000000
                 0x00000000001e6e8 0x00000000001e6e8  R E    0x200000
  LOAD           0x00000000001eff0 0x00000000021eff0 0x00000000021eff0
                 0x0000000000001278 0x0000000000002570  RW     0x200000


locate libc.so
/lib/x86_64-linux-gnu/libc.so.6
/usr/lib/x86_64-linux-gnu/libc.so

young@USys2:/lib$ readelf -a /lib/x86_64-linux-gnu/libc.so.6
  LOAD           0x0000000000000000 0x0000000000000000 0x0000000000000000
                 0x00000000001e6aa0 0x00000000001e6aa0  R E    0x200000
  LOAD           0x00000000001e7620 0x00000000003e7620 0x00000000003e7620
                 0x0000000000005240 0x00000000000094c0  RW     0x200000



https://www.technovelty.org/linux/plt-and-got-the-key-to-code-sharing-and-dynamic-l
```

simple example

```
$ cat a.c
extern int foo;

int function(void) {
    return foo;
}
$ gcc -c -m32 a.c
$ readelf --relocs ./a.o

La sección de reubicación '.rel.text' at offset 0x1f0 contains 3 entries:
 Desplaz     Info    Tipo            Val.Símbolo Nom. Símbolo
0000000f  00000d2b R_386_GOT32X      00000000   foo


https://www.technovelty.org/linux/plt-and-got-the-key-to-code-sharing-and-dynamic-l
```

readef -relocs

```
$ gcc -c -m32 a.c
$ readelf --relocs ./a.o

La sección de reubicación '.rel.text' at offset 0x1f0 contains 3 entries:
 Desplaz    Info    Tipo            Val.Símbolo Nom. Símbolo
00000004  00000b02 R_386_PC32        00000000   __x86.get_pc_thunk.ax
00000009  00000c0a R_386_GOTPC       00000000   _GLOBAL_OFFSET_TABLE_
0000000f  00000d2b R_386_GOT32X      00000000   foo

La sección de reubicación '.rel.eh_frame' at offset 0x208 contains 2 entries:
 Desplaz    Info    Tipo            Val.Símbolo Nom. Símbolo
00000020  00000202 R_386_PC32        00000000   .text
00000040  00000502 R_386_PC32        00000000   .text.__x86.get_pc_thu

https://www.technovelty.org/linux/plt-and-got-the-key-to-code-sharing-and-dynamic-l
```

objdump -disassemble

```
./a.o:     formato del fichero elf32-i386


Desensamblado de la sección .text:

00000000 <function>:
   0:   55                      push   %ebp
   1:   89 e5                   mov    %esp,%ebp
   3:   e8 fc ff ff ff          call   4 <function+0x4>
   8:   05 01 00 00 00          add    $0x1,%eax
   d:   8b 80 00 00 00 00       mov    0x0(%eax),%eax
  13:   8b 00                   mov    (%eax),%eax
  15:   5d                      pop    %ebp
  16:   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

https://www.technovelty.org/linux/plt-and-got-the-key-to-code-sharing-and-dynamic-l
```

test.c returns a global variable

```
$ cat test.c
static int foo = 100;

int function(void) {
    return foo;
}

$ gcc -nostdlib -fPIC -shared -m32 -o libtest.so test.c
$ objdump --disassemble -m i386 libtest.so
```

https://www.technovelty.org/linux/plt-and-got-the-key-to-code-sharing-and-dynamic-l

test.c : objdump -disassemble

```
libtest.so:     formato del fichero elf32-i386


Desensamblado de la sección .text:

000001da <function>:
 1da:   55                      push   %ebp
 1db:   89 e5                   mov    %esp,%ebp
 1dd:   e8 0d 00 00 00          call   1ef <__x86.get_pc_thunk.ax>
 1e2:   05 1e 1e 00 00          add    $0x1e1e,%eax
 1e7:   8b 80 0c 00 00 00       mov    0xc(%eax),%eax
 1ed:   5d                      pop    %ebp
 1ee:   c3                      ret

000001ef <__x86.get_pc_thunk.ax>:
 1ef:   8b 04 24                mov    (%esp),%eax
 1f2:   c3                      ret


https://www.technovelty.org/linux/plt-and-got-the-key-to-code-sharing-and-dynamic-l
```

test2.c returns a extern variable

```
$ cat test2.c
extern int foo;

int function(void) {
    return foo;
}

$ gcc -nostdlib -fPIC -shared -m32 -o libtest2.so test2.c
$ objdump --disassemble -m i386 libtest2.so
```

https://www.technovelty.org/linux/plt-and-got-the-key-to-code-sharing-and-dynamic-l

test2.c : objdump -disassemble

```
libtest2.so:     formato del fichero elf32-i386


Desensamblado de la sección .text:

000001f8 <function>:
 1f8:   55                      push   %ebp
 1f9:   89 e5                   mov    %esp,%ebp
 1fb:   e8 0f 00 00 00          call   20f <__x86.get_pc_thunk.ax>
 200:   05 00 1e 00 00          add    $0x1e00,%eax
 205:   8b 80 fc ff ff ff       mov    -0x4(%eax),%eax
 20b:   8b 00                   mov    (%eax),%eax
 20d:   5d                      pop    %ebp
 20e:   c3                      ret

0000020f <__x86.get_pc_thunk.ax>:
 20f:   8b 04 24                mov    (%esp),%eax
 212:   c3                      ret


https://www.technovelty.org/linux/plt-and-got-the-key-to-code-sharing-and-dynamic-l
```

test2.c : readelf -sections

```
There are 16 section headers, starting at offset 0x12c0:

Encabezados de Sección:
  [Nr] Nombre              Tipo            Direc    Desp   Tam    ES Opt En Inf Al
  [ 0]                     NULL            00000000 000000 000000 00      0   0  0
  [ 1] .note.gnu.build-i NOTE            00000114 000114 000024 00   A  0   0  4
  [ 2] .gnu.hash           GNU_HASH        00000138 000138 000030 04   A  3   0  4
  [ 3] .dynsym             DYNSYM          00000168 000168 000060 10   A  4   1  4
  [ 4] .dynstr             STRTAB          000001c8 0001c8 000026 00   A  0   0  1
  [ 5] .rel.dyn            REL             000001f0 0001f0 000008 08   A  3   0  4
  [ 6] .text               PROGBITS        000001f8 0001f8 00001b 00  AX  0   0  1
  [ 7] .eh_frame_hdr       PROGBITS        00000214 000214 00001c 00   A  0   0  4
  [ 8] .eh_frame           PROGBITS        00000230 000230 00004c 00   A  0   0  4
  [ 9] .dynamic            DYNAMIC         00001f8c 000f8c 000070 08  WA  4   0  4
  [10] .got                PROGBITS        00001ffc 000ffc 000004 04  WA  0   0  4
  [11] .got.plt            PROGBITS        00002000 001000 00000c 04  WA  0   0  4
  [12] .comment            PROGBITS        00000000 00100c 00002a 01  MS  0   0  1
  [13] .symtab             SYMTAB          00000000 001038 000180 10     14  19  4
  [14] .strtab             STRTAB          00000000 0011b8 000076 00      0   0  1
  [15] .shstrtab           STRTAB          00000000 00122e 00008f 00      0   0  1
Key to Flags:
  W (write), A (alloc), X (execute), M (merge), S (strings), I (info),
  L (link order), O (extra OS processing required), G (group), T (TLS),
  C (compressed), x (unknown), o (OS specific), E (exclude),
```

test2.c : readelf -relocs

```
La sección de reubicación '.rel.dyn' at offset 0x1f0 contains 1 entry:
 Desplaz     Info    Tipo            Val.Símbolo Nom. Símbolo
00001ffc  00000106 R_386_GLOB_DAT    00000000   foo
```

https://www.technovelty.org/linux/plt-and-got-the-key-to-code-sharing-and-dynamic-l

test3.c returns extern function

```
$ cat test3.c
extern int foo(void);

int function(void) {
    return foo();
}

$ gcc -nostdlib -fPIC -shared -m32 -o libtest3.so test3.c
$ objdump --disassemble -m i386 libtest3.so
```

https://www.technovelty.org/linux/plt-and-got-the-key-to-code-sharing-and-dynamic-l

test3.c : objdump -disassemble (1)

```
libtest3.so:     formato del fichero elf32-i386


Desensamblado de la sección .plt:

00000200 <.plt>:
 200:   ff b3 04 00 00 00       pushl  0x4(%ebx)
 206:   ff a3 08 00 00 00       jmp    *0x8(%ebx)
 20c:   00 00                   add    %al,(%eax)
        ...

00000210 <foo@plt>:
 210:   ff a3 0c 00 00 00       jmp    *0xc(%ebx)
 216:   68 00 00 00 00          push   $0x0
 21b:   e9 e0 ff ff ff          jmp    200 <.plt>
```

https://www.technovelty.org/linux/plt-and-got-the-key-to-code-sharing-and-dynamic-l

test3.c : objdump -disassemble (2)

```
Desensamblado de la sección .text:

00000220 <function>:
 220:   55                      push   %ebp
 221:   89 e5                   mov    %esp,%ebp
 223:   53                      push   %ebx
 224:   83 ec 04                sub    $0x4,%esp
 227:   e8 12 00 00 00          call   23e <__x86.get_pc_thunk.ax>
 22c:   05 d4 1d 00 00          add    $0x1dd4,%eax
 231:   89 c3                   mov    %eax,%ebx
 233:   e8 d8 ff ff ff          call   210 <foo@plt>
 238:   83 c4 04                add    $0x4,%esp
 23b:   5b                      pop    %ebx
 23c:   5d                      pop    %ebp
 23d:   c3                      ret

0000023e <__x86.get_pc_thunk.ax>:
 23e:   8b 04 24                mov    (%esp),%eax
 241:   c3                      ret
```

https://www.technovelty.org/linux/plt-and-got-the-key-to-code-sharing-and-dynamic-l

test3.c : readelf -sections

```
There are 16 section headers, starting at offset 0x12bc:

Encabezados de Sección:
  [Nr] Nombre              Tipo            Direc    Desp   Tam    ES Opt En Inf Al
  [ 0]                     NULL            00000000 000000 000000 00      0   0  0
  [ 1] .note.gnu.build-i NOTE            00000114 000114 000024 00   A  0   0  4
  [ 2] .gnu.hash           GNU_HASH        00000138 000138 000030 04   A  3   0  4
  [ 3] .dynsym             DYNSYM          00000168 000168 000060 10   A  4   1  4
  [ 4] .dynstr             STRTAB          000001c8 0001c8 000026 00   A  0   0  1
  [ 5] .rel.plt            REL             000001f0 0001f0 000008 08  AI  3  11  4
  [ 6] .plt                PROGBITS        00000200 000200 000020 04  AX  0   0 16
  [ 7] .text               PROGBITS        00000220 000220 000022 00  AX  0   0  1
  [ 8] .eh_frame_hdr       PROGBITS        00000244 000244 000024 00   A  0   0  4
  [ 9] .eh_frame           PROGBITS        00000268 000268 000074 00   A  0   0  4
  [10] .dynamic            DYNAMIC         00001f88 000f88 000078 08  WA  4   0  4
  [11] .got.plt            PROGBITS        00002000 001000 000010 04  WA  0   0  4
  [12] .comment            PROGBITS        00000000 001010 00002a 01  MS  0   0  1
  [13] .symtab             SYMTAB          00000000 00103c 000180 10     14  19  4
  [14] .strtab             STRTAB          00000000 0011bc 000076 00      0   0  1
  [15] .shstrtab           STRTAB          00000000 001232 00008a 00      0   0  1
Key to Flags:
  W (write), A (alloc), X (execute), M (merge), S (strings), I (info),
  L (link order), O (extra OS processing required), G (group), T (TLS),
  C (compressed), x (unknown), o (OS specific), E (exclude),
```

test3.c : readelf -relocs

```
La sección de reubicación '.rel.plt' at offset 0x1f0 contains 1 entry:
 Desplaz    Info    Tipo            Val.Símbolo Nom. Símbolo
0000200c  00000107 R_386_JUMP_SLOT   00000000   foo
```

https://www.technovelty.org/linux/plt-and-got-the-key-to-code-sharing-and-dynamic-l

test3.c : objdump -disassemble-all (1)

```
Desensamblado de la sección .note.gnu.build-id:
Desensamblado de la sección .gnu.hash:
Desensamblado de la sección .dynsym:
Desensamblado de la sección .dynstr:
Desensamblado de la sección .rel.plt:
Desensamblado de la sección .plt:
Desensamblado de la sección .text:
Desensamblado de la sección .eh_frame_hdr:
Desensamblado de la sección .eh_frame:
Desensamblado de la sección .dynamic:
Desensamblado de la sección .got.plt:
Desensamblado de la sección .comment:
```

https://www.technovelty.org/linux/plt-and-got-the-key-to-code-sharing-and-dynamic-l

test3.c : objdump -disassemble-all (2)

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

000001f0 <.rel.plt>:
 1f0:   0c 20                   or     $0x20,%al
 1f2:   00 00                   add    %al,(%eax)
 1f4:   07                      pop    %es
 1f5:   01 00                   add    %eax,(%eax)
        ...

Desensamblado de la sección .plt:

00000200 <.plt>:
 200:   ff b3 04 00 00 00       pushl  0x4(%ebx)
 206:   ff a3 08 00 00 00       jmp    *0x8(%ebx)
 20c:   00 00                   add    %al,(%eax)
        ...

00000210 <foo@plt>:
 210:   ff a3 0c 00 00 00       jmp    *0xc(%ebx)
 216:   68 00 00 00 00          push   $0x0
 21b:   e9 e0 ff ff ff          jmp    200 <.plt>
```

test3.c : objdump -disassemble-all (3)

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

00002000 <_GLOBAL_OFFSET_TABLE_>:
    2000:       88 1f                       mov    %bl,(%edi)
        ...
    200a:       00 00                       add    %al,(%eax)
    200c:       16                          push   %ss
    200d:       02 00                       add    (%eax),%al
        ...
```

https://www.technovelty.org/linux/plt-and-got-the-key-to-code-sharing-and-dynamic-l