ELF1 7 Examples - 2 Object File rel.o - ELF Study 1999

Young W. Lim

2020-03-19 Thr

Outline

Based on

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

Compling 32-bit program on 64-bit gcc

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

TOC: Summary of the results

- Reloc summary for `rel.o` object file
- Symbols and sections for `rel.o` object file

TOC: Reloc summary for `rel.o` object file

- Relocation table sections for `rel.o` object file
- Relocation listing sections for `rel.o` object file
- a) data section relocs of an object file `rel.o`
- b) text section relocs of an object file `rel.o`
- c) `.rel.data(.rel)` reloc listing of an object file `rel.o`
- d) `.rel.text` reloc listing of an object file `rel.o`

Relocation table sections for rel.o object files

- for rel.o

	-fno-pic	default	-fPIC
.plt			
.plt.got			
.got			
.got.plt			

```
readelf -t run-fno-pic | grep -e .plt -e .got -e .rel
```

Relocation listing sections for rel.o object files

- for rel.o

	-fno-pic	default	-fPIC
.rel.data	✓		
.rel.data.rel		✓	✓
.rel.text	✓	✓	✓
.rel.dyn			
.rel.plt			
.rel.got			

```
readelf -t run-fno-pic | grep -e .plt -e .got -e .rel
```

a) data section relocs of an object file rel.o

- data section relocs of `rel.o` file
  - local data symbol reference (`cLocal`)
    no reloc (offset relative to `.bss`)
    `R_386_32` for the `.bss` reference in `.data`
  - local function symbol reference (`fLocal`)
    no reloc (offset relative to `.text`)
    `R_386_32` for the `.text` reference in `.data`
  - global data symbol reference (`cPub`)
    `R_386_32` in `.data` (absolute)
  - global function symbol reference (`fPub`)
    `R_386_32` in `.data` (absolute)

  for all cases (-fno-pic, default, -fPIC)

b) text section relocs of an object file rel.o

- text section relocs of rel.o file
  - local data symbol reference (cLocal)
    no reloc (offset relative to .bss)
    R_386_GOTOFF for .bss : when GOT is used (default, -fPIC)
    R_386_32 for .bss : otherwise (-fno-pic)
  - local function symbol reference (fLocal)
    no reloc (offset relative to .text)
    fully resolved for the .text symbol reference in .text
  - global data symbol reference (cPub)
    R_386_GOT32 in .text : when GOT is used (default, -fPIC)
    R_386_32 in .text : otherwise (-fno-pic)
  - global function symbol reference (fPub)
    R_386_PLT32 in .text : when PLT is used (-fPIC)
    R_386_PC32 in .text : otherwise (-fno-pic, default)

c) .rel.data{.rel} reloc listing of an object file rel.o

- .rel.data reloc listing of rel.o file (-fno-pic)
  .rel.data.rel reloc listing of rel.o file (default, -fPIC)

symbol	-fno-pic	default	-fPIC
cPub	R_386_32	R_386_32	R_386_32
	in .data	in .data.rel	in .data.rel
fPub	R_386_32	R_386_32	R_386_32
	in .data	in .data.rel	in .data.rel
.bss	R_386_32	R_386_32	R_386_32
(cLocal)	in .data	in .data.rel	in .data.rel
.text	R_386_32	R_386_32	R_386_32
(fLocal)	in .data	in .data.rel	in .data.rel

d) .rel.text reloc listing of an object file rel.o

- .rel.text reloc listing of rel.o file

symbol	-fno-pic	default	-fPIC
cPub	R_386_32	R_386_GOT32x	R_386_GOT32x
	in .text	in .text	in .text
fPub	R_386_PC32	R_386_PC32	R_386_PLT32
	in .text	in .text	in .text
.bss	R_386_32	R_386_GOTOFF	=R_386_GOTOFF
(cLocal)	in .text	in .text	in .text
.text	resolved	resolved	resolved
(fLocal)	in .text	in .text	in .text

- fPub is defined in the same module (rel.o)

TOC: Symbols and sections for `rel.o` object file

- Symbol table in `rel.o` (-fno-pic, default, -fPIC)
- Section header in `rel.o` (-fno-pic, default, -fPIC)
- Relocs in `rel.o` (-fno-pic, default, -fPIC)

Symbol table in rel.o (-fno-pic)

```
young@USys2:~$ readelf -s rel-fno-pic.o

Symbol table '.symtab' contains 14 entries:
   Num:    Valor  Tam  Tipo    Unión  Vis      Nombre Ind
     0: 00000000    0 NOTYPE  LOCAL  DEFAULT  UND
     1: 00000000    0 FILE    LOCAL  DEFAULT  ABS rel.c
     5: 00000008    8 FUNC    LOCAL  DEFAULT    1 fLocal
     6: 00000000    1 OBJECT  LOCAL  DEFAULT    5 cLocal
    10: 00000000    8 FUNC    GLOBAL DEFAULT    1 fPub
    11: 00000001    1 OBJECT  GLOBAL DEFAULT  COM cPub
    12: 00000000   16 OBJECT  GLOBAL DEFAULT    3 a
    13: 00000010   77 FUNC    GLOBAL DEFAULT    1 foo
```

https://stackoverflow.com/questions/1685483/how-can-i-examine-contents-of-a-data-s

Symbol table in rel.o (default)

```
young@USys2:~$ readelf -s rel-default.o

Symbol table '.symtab' contains 22 entries:
   Num:    Valor  Tam  Tipo    Unión  Vis      Nombre Ind
     0: 00000000    0 NOTYPE  LOCAL  DEFAULT  UND
     1: 00000000    0 FILE    LOCAL  DEFAULT  ABS rel.c
     5: 00000012   18 FUNC    LOCAL  DEFAULT    3 fLocal
     6: 00000000    1 OBJECT  LOCAL  DEFAULT    6 cLocal
    15: 00000000   18 FUNC    GLOBAL DEFAULT    3 fPub
    17: 00000000    0 NOTYPE  GLOBAL DEFAULT  UND _GLOBAL_OFFSET_TABLE_
    18: 00000001    1 OBJECT  GLOBAL DEFAULT  COM cPub
    19: 00000000   16 OBJECT  GLOBAL DEFAULT    7 a
    20: 00000024   96 FUNC    GLOBAL DEFAULT    3 foo
```

https://stackoverflow.com/questions/1685483/how-can-i-examine-contents-of-a-data-s

Symbol table in rel.o (-fPIC)

```
young@USys2:~$ readelf -s rel-fPIC.o

Symbol table '.symtab' contains 22 entries:
   Num:    Valor  Tam  Tipo    Unión  Vis      Nombre Ind
     0: 00000000    0 NOTYPE  LOCAL  DEFAULT  UND
     1: 00000000    0 FILE    LOCAL  DEFAULT  ABS rel.c
     5: 00000012   18 FUNC    LOCAL  DEFAULT    3 fLocal
     6: 00000000    1 OBJECT  LOCAL  DEFAULT    6 cLocal
    15: 00000000   18 FUNC    GLOBAL DEFAULT    3 fPub
    17: 00000000    0 NOTYPE  GLOBAL DEFAULT  UND _GLOBAL_OFFSET_TABLE_
    18: 00000001    1 OBJECT  GLOBAL DEFAULT  COM cPub
    19: 00000000   16 OBJECT  GLOBAL DEFAULT    7 a
    20: 00000024  102 FUNC    GLOBAL DEFAULT    3 foo
```

https://stackoverflow.com/questions/1685483/how-can-i-examine-contents-of-a-data-s

Section header in rel.o (-fno-pic)

```
young@USys2:~$ readelf -S rel-fno-pic.o
```

[Nr]	Nombre	Tipo	Direc	Desp	Tam	ES	Opt	En	Inf	Al
[1]	.text	PROGBITS	00000000	000034	00005d	00	AX	0	0	1
[2]	.rel.text	REL	00000000	000254	000028	08	I	10	1	4
[3]	.data	PROGBITS	00000000	000094	000010	00	WA	0	0	4
[4]	.rel.data	REL	00000000	00027c	000020	08	I	10	3	4
[5]	.bss	NOBITS	00000000	0000a4	000001	00	WA	0	0	1
[10]	.symtab	SYMTAB	00000000	00014c	0000e0	10		11	10	4

Section header in rel.o (default)

```
young@USys2:~$ readelf -S rel-default.o
```

[Nr]	Nombre	Tipo	Direc	Desp	Tam	ES	Opt	En	Inf	Al
[3]	.text	PROGBITS	00000000	000044	000084	00	AX	0	0	1
[4]	.rel.text	REL	00000000	00037c	000058	08	I	15	3	4
[5]	.data	PROGBITS	00000000	0000c8	000000	00	WA	0	0	1
[6]	.bss	NOBITS	00000000	0000c8	000001	00	WA	0	0	1
[7]	.data.rel	PROGBITS	00000000	0000c8	000010	00	WA	0	0	4
[8]	.rel.data.rel	REL	00000000	0003d4	000020	08	I	15	7	4
[15]	.symtab	SYMTAB	00000000	0001b4	000160	10		16	15	4

Section header in rel.o (-fPIC)

```
young@USys2:~$ readelf -S rel-fPIC.o
```

[Nr]	Nombre	Tipo	Direc	Desp	Tam	ES	Opt	En	Inf	Al
[3]	.text	PROGBITS	00000000	000044	00008a	00	AX	0	0	1
[4]	.rel.text	REL	00000000	000384	000058	08	I	15	3	4
[5]	.data	PROGBITS	00000000	0000ce	000000	00	WA	0	0	1
[6]	.bss	NOBITS	00000000	0000ce	000001	00	WA	0	0	1
[7]	.data.rel	PROGBITS	00000000	0000d0	000010	00	WA	0	0	4
[8]	.rel.data.rel	REL	00000000	0003dc	000020	08	I	15	7	4
[15]	.symtab	SYMTAB	00000000	0001bc	000160	10		16	15	4

Relocs of rel.o (-fno-pic)

```
young@USys2:~$ readelf -r rel-fno-pic.o

La sección de reubicación '.rel.text' at offset 0x254 contains 5 entries:
 Desplaz    Info    Tipo            Val.Símbolo Nom. Símbolo
00000018  00000a02 R_386_PC32        00000000   fPub
00000033  00000b01 R_386_32          00000001   cPub
0000003c  00000b01 R_386_32          00000001   cPub
00000046  00000401 R_386_32          00000000   .bss
0000004f  00000401 R_386_32          00000000   .bss

La sección de reubicación '.rel.data' at offset 0x27c contains 4 entries:
 Desplaz    Info    Tipo            Val.Símbolo Nom. Símbolo
00000000  00000401 R_386_32          00000000   .bss
00000004  00000201 R_386_32          00000000   .text
00000008  00000b01 R_386_32          00000001   cPub
0000000c  00000a01 R_386_32          00000000   fPub
```

Relocs of rel.o (default)

```
La sección de reubicación '.rel.text' at offset 0x37c contains 11 entries:
 Desplaz    Info    Tipo            Val.Símbolo Nom. Símbolo
00000009  0000110a R_386_GOTPC       00000000   _GLOBAL_OFFSET_TABLE_
0000001b  0000110a R_386_GOTPC       00000000   _GLOBAL_OFFSET_TABLE_
00000030  0000110a R_386_GOTPC       00000000   _GLOBAL_OFFSET_TABLE_
00000038  00000f02 R_386_PC32        00000000   fPub
00000055  0000122b R_386_GOT32X      00000001   cPub
0000005d  0000122b R_386_GOT32X      00000001   cPub
0000006b  00000409 R_386_GOTOFF      00000000   .bss
00000074  00000409 R_386_GOTOFF      00000000   .bss

La sección de reubicación '.rel.data.rel' at offset 0x3d4 contains 4 entries:
 Desplaz    Info    Tipo            Val.Símbolo Nom. Símbolo
00000000  00000401 R_386_32          00000000   .bss
00000004  00000201 R_386_32          00000000   .text
00000008  00001201 R_386_32          00000001   cPub
0000000c  00000f01 R_386_32          00000000   fPub
```

Relocs of rel.o (-fPIC)

```
young@USys2:~$ readelf -r rel-fPIC.o

La sección de reubicación '.rel.text' at offset 0x384 contains 11 entries:
 Desplaz    Info    Tipo            Val.Símbolo Nom. Símbolo
00000009  0000110a R_386_GOTPC       00000000   _GLOBAL_OFFSET_TABLE_
0000001b  0000110a R_386_GOTPC       00000000   _GLOBAL_OFFSET_TABLE_
00000030  0000110a R_386_GOTPC       00000000   _GLOBAL_OFFSET_TABLE_
0000003b  00000f04 R_386_PLT32       00000000   fPub
0000005b  0000122b R_386_GOT32X      00000001   cPub
00000063  0000122b R_386_GOT32X      00000001   cPub
00000071  00000409 R_386_GOTOFF      00000000   .bss
0000007a  00000409 R_386_GOTOFF      00000000   .bss

La sección de reubicación '.rel.data.rel' at offset 0x3dc contains 4 entries:
 Desplaz    Info    Tipo            Val.Símbolo Nom. Símbolo
00000000  00000401 R_386_32          00000000   .bss
00000004  00000201 R_386_32          00000000   .text
00000008  00001201 R_386_32          00000001   cPub
0000000c  00000f01 R_386_32          00000000   fPub
```

TOC: Symbol references in the definition of a[] (.data)

- (1) local symbol reference in .data
- (2) local symbol's section-plut-offset in .data
- (3) .bss and .text symbol reference in .data
- (4) .bss and .text symbol relocation in .data
- (5) global symbol reference in .data
- Relocs in the data section summary

(1) local symbol reference in .data

- &cLocal and fLocal *would have*
  a R_386_32 reloc

- but these symbols are local symbols
  thus, their addresses are determined locally (*resolved*)
  by an offset to the .bss or .text address, respectively

```
static char fLocal(int b) { return b; }
static char cLocal;
```

http://netwinder.osuosl.org/users/p/patb/public_html/elf_relocs.html

(2) local symbol's section-plus-offset in .data

- `&cLocal` and `fLocal` symbols are accessed by a section-plus-offset, using a section index for the `.bss` or `.text` section
- the *symbol name* is not used in this case but the *section number* is used instead

`http://netwinder.osuosl.org/users/p/patb/public_html/elf_relocs.html`

(3) .bss and .text symbol reference in .data

- .bss and .text symbol references in the .data section have a R_386_32

- .rel.data reloc listing (-fno-pic)
  .rel.data.rel reloc listing (default and -fPIC)
  - .bss has the R_386_32 reloc (for the uninitialized &cLocal)
  - .text has the R_386_32 reloc (for the function fLocal)

http://netwinder.osuosl.org/users/p/patb/public_html/elf_relocs.html

(4) .bss and .text symbol relocation in .data

- the load time address of .bss or .text is not known at compile time
- the R_386_32 reloc of the .bss or .text symbol will be changed into a R_386_RELATIVE reloc after dynamic linking
- the module load address will be added to the offset at run time

http://netwinder.osuosl.org/users/p/patb/public_html/elf_relocs.html

(5) global symbol reference in .data

- .rel.data reloc listing (-fno-pic)
  .rel.data.rel reloc listing (default and -fPIC)
  - cPub has the R_386_32 reloc (uninitialized)
  - fPub has the R_386_32 reloc (function)
- again, &cPub and fPub addresses could also be determined locally using an offset to the .bss or .text address of the module
- but these are global symbols, they all have the R_386_32 reloc

```
char fPub(int a) { return a; }
char cPub;
```

my opinon

Relocs in the data section summary

- Relocs in the data section
  - `rel.data` reloc listing (-fno-pic)
  - `rel.data.rel` reloc listing (default, -fPIC)

symbol	type	-fno-pic	default	-fPIC
cPub	global data	R_386_32	R_386_32	R_386_32
fPub	glocal func	R_386_32	R_386_32	R_386_32
cLocal	local data	relative to	relative to	relative to
.bss		R_386_32	R_386_32	R_386_32
fLocal	local func	relative to	relative to	relative to
.text		R_386_32	R_386_32	R_386_32

O

```
_st a[] = { { &cLocal,  // 1
              fLocal }, // 2
            { &cPub,    // 3
              fPub } }; // 4
```

```
typedef struct {
     char* p;
     char (*f)(int);
   } _st;
```

TOC: Symbol references in the definition of `foo` (`.text`)

(1) Function symbols

- defined function codes are located in the `.text` section
- a function symbol is
  - a global symbol
  - a local symbol, if `static` keyword is used
- a function symbol reference
  - external function call
    when the defintion of the called function in other module
  - local function call
    function call within the same module

(2) PIC symbol references in a function definition

- position independent code utilizes GOT and PLT
- each module has its own GOT and PLT

	functon symbol reference	data symbol reference
global	PLT / GOT	GOT
local	N/A	N/A

(3) PIC prolog overhead in a function definition

- position independent `foo` requires `&GOT[0]`
- `&GOT[0]` has its own reloc `R_386_GOTPC`
  (the distance from *here* to the GOT)
- the addtional overhead of each public function
  - in the begining of the prolog in `foo()`
    a local reference to the GOT is generated
    `mov &GOT[0], %ebx`
    the rest part of the procedure can refer to `&GOT[0]`

`http://netwinder.osuosl.org/users/p/patb/public_html/elf_relocs.html`

(4) PIC symbol references in `foo` (`.text`)

- `int foo(int a)` publishes the function as a public symbol
  - is called from outside the module
  - position independent enforced by `-fPIC`

  - global function symbol reference `fPub` : PLT / GOT
  - local function symbol reference `fLocal` : no reloc
  - global data symbol reference `&cPub`, `cPub` : GOT
  - local data symbol reference `&cLocal`, `cLocal` : no reloc

`http://netwinder.osuosl.org/users/p/patb/public_html/elf_relocs.html`

Relocs in the text section summary

- relocation information for the referenced symbols in the definition of `foo` function : `.rel.text`:

	default	-fPIC	-fno-pic
fPub	R_386_PC32	R_386_PLT32	R_386_PC32
fLocal	—	—	—
&cPub	R_386_GOT32x	R_386_GOT32x	R_386_32
&cLocal	—	—	—

```
int foo(int a) {          // 5          + cPub            // 9
  return fPub(a)          // 6          + (int) &cLocal   // 10
       + fLocal(a)        // 7          + cLocal;         // 11
       + (int) &cPub      // 8      }
```

TOC: Relocs in the definition of a[] (.data)

- Initialized structure array `a[]` definition
- Initializing 1st member : `char *p`
- Initializing 2nd member : `char (*f) (int)`
- (1) local symbol refences in `.data`
- (2) local symbol refences in `.data`
- (3) global symbol refences in `.data`
- (4) global symbol refences in `.data`

Initialized structure array a[] definition

- the array a[] is a global array in .data
  which are initialized by global variables and functions

  1. address of a static variable cLocal in .bss
  2. address of a static function fLocal in .text
  3. address of a global variable cPub in .bss (actually, COMMON)
  4. address of a global function fPub in .text

```
_st a[] = { { &cLocal,  // 1
              fLocal }, // 2
            { &cPub,    // 3
              fPub } }; // 4
```

```
typedef struct {
     char* p;
     char (*f)(int);
   } _st;
```

Initializing 1st member : char *p

- the 1st member type : character pointer `char *p`

cLocal	cPub
local data symbol - global variable with `static`	global data symbol - global veriable

&cLocal	&cPub
local symbol reference - address of the data	global symbol reference - address of the data

Initializing 2nd member : `char (*f) (int)`

- the 2nd member type : function pointer (address) `char (*f) (int)`
  - no function call / jump is performed (no PLT is used)

`fLocal`	`fPub`
local function symbol - function with `static`	global function symbol - function

`fLocal`	`fPub`
local function symbol reference (function call) - address of the function	global function symbol reference (function call) - address of the function

(1) local symbol references in .data

- when the .o file is built,
  - &cLocal and fLocal are marked as needing a full 32-bit address
  - R_386_32 relocs are generated
- but cLocal and fLocal are local symbols and are referenced only within the same module
- the address can be determined locally and the reloc is dropped and offset is used instead
- &fLocal may be inlined into the caller function of the module

(2) local symbol reqferences in .data

- &cLocal and fLocal addresses can be determined locally using an offset to the .bss or .text address of the module
  - cLocal is in the .bss section (uninitialized)
  - fLocal is in the .text section (function)
- these symbols are referenced by the section number
- .rel.data.rel and .rel.data reloc listings say
  - .bss has the R_386_32 reloc
  - .text has the R_386_32 reloc

```
La sección de reubicación '.rel.data.rel' at offset 0x3d4 contains 4 entries:
 Desplaz     Info     Tipo            Val.Símbolo Nom. Símbolo
00000000  00000401 R_386_32          00000000   .bss
00000004  00000201 R_386_32          00000000   .text
00000008  00001201 R_386_32          00000001   cPub
0000000c  00000f01 R_386_32          00000000   fPub
```

(3) global symbol references in .data

- &cPub and fPub are marked as needing a full 32-bit address
  - R_386_32 relocs are generated
    full absolute addresses
  - these symbols are referenced by their names

```
_st a[] = { { &cLocal,  // 1
              fLocal }, // 2
            { &cPub,    // 3
              fPub } }; // 4
```

http://netwinder.osuosl.org/users/p/patb/public_html/elf_relocs.html

(4) global symbol references in .data

- &cPub and fPub addresses could also be determined locally using an offset to the .bss or .text address of the module
- but these are global symbols, they all have the R_386_32 reloc
- .rel.data.rel and .rel.data reloc listings say
  - cPub has the R_386_32 reloc
  - fPub has the R_386_32 reloc

```
La sección de reubicación '.rel.data.rel' at offset 0x3d4 contains 4 entries:
 Desplaz     Info     Tipo            Val.Símbolo Nom. Símbolo
00000000  00000401 R_386_32          00000000   .bss
00000004  00000201 R_386_32          00000000   .text
00000008  00001201 R_386_32          00000001   cPub
0000000c  00000f01 R_386_32          00000000   fPub
```

my opinon

TOC: Relocs in the definition of `foo` (`.text`)

- Public function `foo` definition
- Function calls : `fPub(a)`, `fLocal(a)`
- Addresses and contents of `cPub`, `cLocal`
- (1) function symbol refernces in `.text`
- (2) global data symbol references in `.text`
- (3) local data symbol references in `.text`
- Unresolve references

Public function foo defintiion

- the function foo is in .text

1. a global function call fPub(a) in .text
2. a local function call fLocal(a) in .text
3. the address of a global variable cPub in .bss (COMMON)
4. the content of aglobal variable cPub in .bss (COMMON)
5. the address of a local variable cLocal in .bss
6. the content of a local variable cLocal in .bss

```
int foo(int a) {          // 5          + cPub             // 9
  return fPub(a)          // 6          + (int) &cLocal    // 10
       + fLocal(a)        // 7          + cLocal;          // 11
       + (int) &cPub      // 8      }
```

http://netwinder.osuosl.org/users/p/patb/public_html/elf_relocs.html

Function calls : fPub(a), fLocal(a)

- function calls (address) fPub(a) , fLocal(a)

fPub	fLocal
function	function with static
global function symbol	local function symbol

fPub(a)	fLocal(a)
a function call	a function call
global function symbol reference	local function symbol reference

```
int foo(int a) {          // 5            + cPub             // 9
  return fPub(a)          // 6*           + (int) &cLocal    // 10
       + fLocal(a)        // 7*           + cLocal;          // 11
       + (int) &cPub      // 8       }
```

http://netwinder.osuosl.org/users/p/patb/public_html/elf_relocs.html

Addresses and contents of cPub, cLocal

- character pointer `char *p`

cPub	cLocal
global veriable	global variable with `static`
global symbol	local symbol
`&cPub` : address `cPub` : content	`&cLocal` : address `cLocal` : content
global symbol reference	local symbol reference

```
int foo(int a) {          // 5            + cPub            // 9*
  return fPub(a)          // 6            + (int) &cLocal   // 10*
       + fLocal(a)        // 7            + cLocal;         // 11*
       + (int) &cPub      // 8*     }
```

(1) function symbol references in .text

- global function symbol reference `fPub(a)`
  has a `R_386_PLT32` reloc (when `-fPIC` used)
- local function symbol reference `fLocal(a)` also
  has a `R_386_PLT32` reloc (when `-fPIC` used)
  - this reloc will disappear
    at the final link because `fLocal` is local

`http://netwinder.osuosl.org/users/p/patb/public_html/elf_relocs.html`

(2) global data symbol references in .text

- &cPub requires the address of a public object
  - this object location must be flexible at run time
  - R_386_GOT32 reloc is used
  - later at link time, this will create
    an address slot in the GOT[] (R_386_GLOB_DAT)
- cPub requires the content of a public object
  - the object contents are fetched by using
    the address stored in the GOT[] entry
- actually use a single reloc to &cPub and cPub

http://netwinder.osuosl.org/users/p/patb/public_html/elf_relocs.html

(3) local data symbol references in .text

- &cLocal requires the address of a local object
- cLocal requires the data of a local object

- we don't know the exact location
  where this object is going to be located in run time
- but because it is a local object, we can represent
  its position relative to the &GOT[0]
- a pair of R_386_GOTOFF relocs are generated
- the compiler may merge these two into a single line
  but it does not

http://netwinder.osuosl.org/users/p/patb/public_html/elf_relocs.html

Unresolved references

- because of the structure of the linker
  full name resolution is not checked
  until a link is made with an executable

- if your library has unresolved references,
  you won't find out it until you try
  to make an app using your library

`http://netwinder.osuosl.org/users/p/patb/public_html/elf_relocs.html`

TOC: Locating relocs of `rel.o` object file

- Locating `.data` section relocs of `rel.o` object file
- Locating `.text` section relocs of `rel.o` object file

TOC: Locating .data section relocs of rel.o shared object

- (a) referencing symbols in the .data section of rel.o
- (b) disassemble .data.rel section in rel.o
- (c) hexadump data.rel section in rel.o
- Relocs to be converted in the .data section of rel.o

(a) referencing symbols in the .data section of rel.o

- .data section of rel.o with -fno-pic

```
0x00000000 00000000    .bss    R_386_32   cLocal
0x00000004 00000012    .text   R_386_32   fLocal
0x00000008 00000000    cPub    R_386_32
0x0000000c 00000000    fPub    R_386_32
```

- .data.rel section of rel.o with default

```
0x00000000 00000000    .bss    R_386_32   cLocal
0x00000004 00000012    .text   R_386_32   fLocal
0x00000008 00000000    cPub    R_386_32
0x0000000c 00000000    fPub    R_386_32
```

- .data.rel section of rel.o with -fPIC

```
0x00000000 00000000    .bss    R_386_32   cLocal
0x00000004 00000012    .text   R_386_32   fLocal
0x00000008 00000000    cPub    R_386_32
0x0000000c 00000000    fPub    R_386_32
```

(b) disassemble .data.rel section in rel.o (-fPIC)

```
00000000 <a>:
   0:   00 00                   add    %al,(%eax)
                        0: R_386_32     .bss          cLocal
   2:   00 00                   add    %al,(%eax)
   4:   12 00                   adc    (%eax),%al
                        4: R_386_32     .text         fLocal
        ...
                        8: R_386_32     cPub
                        c: R_386_32     fPub
```

(c) hexadump .data.rel section in rel.o (-fPIC)

```
objdump -s -j .data.rel rel-fPIC.o

rel-fPIC.o:     file format elf32-i386

Contents of section .data.rel:
 0000 00000000 12000000 00000000 00000000  ................

readelf -x .data.rel rel-fPIC.o

Hex dump of section '.data.rel':
 NOTE: This section has relocations against it,
 but these have NOT been applied to this dump.
  0x00000000 00000000 12000000 00000000 00000000 ................

  https://stackoverflow.com/questions/1685483/how-can-i-examine-contents-of-a-data-s
```

Relocs to be converted in the .data section of rel.o

- .data section relocs of rel.o with -fno-pic

```
.bss  sym in .data (R_386_32)      >>>>  .bss  sym in .data (R_386_RELATIVE)
.text sym in .data (R_386_32)      >>>>  .text sym in .data (R_386_RELATIVE)
```

- .data.rel section relocs of rel.o with default

```
.bss  sym in .data.rel (R_386_32) >>>>  .bss  sym in .data (R_386_RELATIVE)
.text sym in .data.rel (R_386_32) >>>>  .text sym in .data (R_386_RELATIVE)
```

- .data.rel section relocs of rel.o with -fPIC

```
.bss  sym in .data.rel (R_386_32) >>>>  .bss  sym in .data (R_386_RELATIVE)
.text sym in .data.rel (R_386_32) >>>>  .text sym in .data (R_386_RELATIVE)
```

TOC: Locating .text section relocs of rel.o shared object

- (a) call fPub in the .text section of rel.o
- (b) call fPub in the .text section of librel.so
- (c) referencing cPub in the .text section of rel.o
- (c) referencing cPub in the .text section of librel.so
- Relocs to be converted in the .text section of rel.o

(a) calling fPub in the .text section of rel.o

- rel.o with -fno-pic

```
17:   e8 fc ff ff ff          call   18 <foo+0x8> ; call function at 18
                      18: R_386_PC32  fPub
                                ;    18 = 10 + 8 ; fPub func ref location
                                ;    -4 = fffffffc ; offset (pc adjust)
                                ;    00000010 <foo>: ...
```

- rel.o with default

```
37:   e8 fc ff ff ff          call   38 <foo+0x14> ; call function at 38
                      38: R_386_PC32  fPub
                                ;    38 = 24 + 14 ; fPub func ref location
                                ;    -4 = fffffffc ; offset (pc adjust)
                                ;    00000024 <foo>: ...
```

- rel.o with -fPIC (fPub : PLT)

```
3a:   e8 fc ff ff ff          call   3b <foo+0x17> ; call function at 3b
                      3b: R_386_PLT32 fPub
                                ;    3b = 24 + 17 ; fPub func ref location
                                ;    -4 = fffffffc ; offset (pc adjust)
                                ;    00000024 <foo>: ...
```

(b) calling fPub in the .text section of librel.so

- librel.so with -fno-pic

```
4c4:  e8 fc ff ff ff          call   4c5 <foo+0x8> ; call func at 4c5
                              ;   4c5 = 4bd + 8; fPub func ref location
                              ;  -4 = fffffffc; offset (pc adjust)
                              ;   000004ad <fPub>:
                              ;   000004bd <foo>: ...
                                  4c5+4
```

- librel.so with default

```
4d4:  e8 fc ff ff ff          call   4d5 <foo+0x14> ; call func at 4d5
                              ;   4d5 = 4c1 + 14; fPub func ref location
                              ;  -4 = fffffffc; offset (pc adjust)
                              ;   0000049d <fPub>:
                              ;   000004c1 <foo>: ...
```

- librel.so with -fPIC (fPub : PLT)

```
4e7:  e8 a4 fe ff ff          call   390 <fPub@plt> ; call func at 390
                              ;   4e8 = fPub func ref location
                              ;  -15c = fffffea4; offset (4e8+4-15c=390)
                              ;   00000390 <fPub@plt>:
                              ;   000004ad <fPub>:
                              ;   000004d1 <foo>: ...
```

(c) referencing cPub in the .text section of rel.o

- rel.o with -fno-pic

```
32:   ba 00 00 00 00          mov    $0x0,%edx
                      33: R_386_32    cPub
39:   0f b6 05 00 00 00 00    movzbl 0x0,%eax
                      3c: R_386_32    cPub
                                ;   33, 3c = cPub symbol ref location
                                ;    0,  0 = offset (no pc adjust)
```

- rel.o with default (cPub : GOT)

```
53:   8b 83 00 00 00 00       mov    0x0(%ebx),%eax
                      55: R_386_GOT32X        cPub
5b:   8b 83 00 00 00 00       mov    0x0(%ebx),%eax
                      5d: R_386_GOT32X        cPub
                                ;   55, 5d = cPub symbol ref location
                                ;    0,  0 = offset (no pc adjust)
```

- rel.o with -fPIC (cPub : GOT)

```
59:   8b 83 00 00 00 00       mov    0x0(%ebx),%eax
                      5b: R_386_GOT32X        cPub
61:   8b 83 00 00 00 00       mov    0x0(%ebx),%eax
                      63: R_386_GOT32X        cPub
                                ;   5b, 63 = cPub symbol ref location
                                ;    0,  0 = offset (no pc adjust)
```

(d) referencing cPub in the .text section of librel.so

- librel.so with -fno-pic

```
4e6:  0f b6 05 00 00 00 00    movzbl 0x0,%eax
                                    ; 4e9 = cPub symbol ref location
                                    ;   0 = offset (no pc adjust)
```

- librel.so with default (cPub : GOT)

```
4f8:  8b 83 f4 ff ff ff       mov    -0xc(%ebx),%eax
                                   ; 4fa = cPub symbol ref location
                                   ;  -c = offset (1fec-4+c=1ff4)
                                   ;  00001fec <.got>: ...
                                   ; same module reference (pc adjust)
```

- librel.so with -fPIC (cPub : GOT)

```
50e:  8b 83 f4 ff ff ff       mov    -0xc(%ebx),%eax
                                   ; 510 = cPub symbol ref location
                                   ;  -c = offset (1fec-4+c=1ff4)
                                   ;  00001fec <.got>: ...
                                   ; same module reference (pc adjust)
```

Relocs to be converted in the .text section of rel.o

- .text section relocs of rel.o with -fno-pic
- .text section relocs of rel.o with default

```
cPUb in .text (R_386_GOT32)  >>>>  entry in .got     (R_386_GLOB_DAT)
```

- .text section relocs of rel.o with -fPIC (fPub : PLT)

```
fPub in .text (R_386_PLT32)  >>>>  slot  in .got.plt (R_386_JUMP_SLOT)
cPUb in .text (R_386_GOT32)  >>>>  entry in .got     (R_386_GLOB_DAT)
```