PB173 – Binární programování Linux

V. ELF II.

Jiri Slaby

Fakulta informatiky
Masarykova univerzita

18. 10. 2016

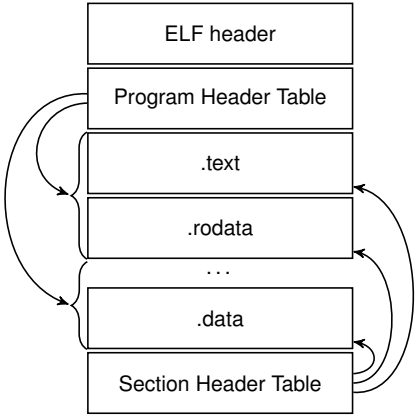
ELF

Executable and Linkable Format

- Hlavní souborový formát na Linuxu
- Přenositelný, relokovatelný, rozšiřitelný

Struktura ELFu

- ELF hlavička
- Hlavičky sekcí
  - Popis dat pro linkování, ladění apod.
  - Pro překladač, debugger, . . .
- Hlavičky programové
  - Pohled na data pro spuštění
  - Pro interpretr
- Data
  - Odkazovány z obou typů hlaviček



Sekce 1

Obsah sekcí

Obsah sekcí

- Závislý na typu sekce (`Shdr->sh_type`)
- Nic (`SHT_NULL`)
- Pro spuštění (`SHT_PROGBITS`)
- Tabulky (`SHT_*TAB`)
  - Počet položek: `Shdr->sh_size / Shdr->sh_entsize`
  - Symbolů
  - Dynamické

Tabulka řetězců

- Typ sekce: `Shdr->sh_type == SHT_STRTAB`
- Řetězce odkazované odjinud
- Uložené jako řetězce ukončené nulou
- Obsah dostaneme z `elf_getdata`

Příklad sekce

Index	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9
0	\0	i	n	i	t	.	c	\0	c	r
10	t	s	t	u	f	f	.	c	\0	_
20	_	J	C	R	_	L	I	S	T	_

Demo: `readelf -x .strtab -p .strtab`

Tabulka symbolů

- Typ sekce: `Shdr->sh_type == SHT_SYMTAB` nebo `SHT_DYNSYM`
- Dump: `readelf --syms`
- Tabulka s různými „symboly”
- Symbol – `GElf_Sym`
  - `st_name` – jméno symbolu (offset do `Shdr->sh_link`)
  - `st_value` – hodnota symbolu
  - `GELF_ST_TYPE(st_info)` – k čemu se symbol váže (`STT_OBJECT`, `STT_FUNC`, `STT_SECTION`, `STT_FILE`, ...)
  - `st_shndx` (pro `STT_SECTION`) – které sekce se symbol týká (může být i `SHN_UNDEF`, `SHN_ABS`, ...)
- `gelf_getsym(Elf_Data *data, int idx, GElf_Sym *dst)`
  - `data` – dostaneme z `elf_getdata`
  - `idx` – x-tý symbol
  - `dst` – návratová hodnota

Úkol

Výpis symbolů ze sekcí

1. Pro všechny sekce typu `SHT_SYMTAB` nebo `SHT_DYNSYM`
2. Iterujte přes všechny symboly
   - `0 .. Shdr->sh_size / Shdr->sh_entsize`
3. U všech vypište jméno symbolu (`elf_strptr`)
4. Pro typ `STT_FUNC` vypište navíc
   - Adresu (`st_value`) a index sekce, ve které je (`st_shndx`)
5. Pro typ `STT_OBJECT` vypište navíc
   - Hodnotu (`st_value`) a velikost (`st_size`)
6. Pro typ `STT_SECTION` vypište navíc
   - Hodnotu (`st_value`) a index sekce (`st_shndx`)

Dynamická tabulka

- Typ sekce: `Shdr->sh_type == SHT_DYNAMIC`
- Dump: `readelf --dynamic`
- Tabulka pro spuštění
- Dynamický symbol – `GElf_Dyn`
  - `d_tag` – druh symbolu (`DT_NEEDED`, `DT_INIT`, ...)
  - `d_u.d_ptr` – hodnota symbolu (ukazatel)
  - `d_u.d_val` – hodnota symbolu (ostatní)
    - Pro jména: offset do sekce `Shdr->sh_link`
- `gelf_getdyn(Elf_Data *data, int idx, GElf_Dyn *dst)`
  - `data` – dostaneme z `elf_getdata`
  - `idx` – x-tý dynamický symbol
  - `dst` – návratová hodnota

Úkol

Výpis dynamických symbolů

1. Pro všechny sekce typu `SHT_DYNAMIC`
2. Iterujte přes všechny dynamické symboly
3. Pro `DT_NEEDED` vypište jméno knihovny (`elf_strptr` a `d_val`)
4. Pro `DT_INIT` vypište, kde se má začít (`d_ptr`)
   - Podívejte se do kódu, kde to je (`objdump -d`)
5. Pro `DT_STRSZ` vypište velikost tabulky řetězců (`d_val`)

Tabulka relokací

- Typ sekce: `Shdr->sh_type == SHT_REL` a `SHT_RELA`
- Dump: `readelf -r`
- Tabulka pro linkování – kde se má co a čím nahradit
- Symbol relokace – `GElf_Rel` a `GElf_Rela`
  - `r_offset` – kde nahrazovat (odkaz do sekce č. `Shdr->sh_info`)
  - `GELF_R_TYPE(r_info)` – typ (`R_X86_64_PC32`, `R_X86_64_32`, ...)
  - `GELF_R_SYM(r_info)` – za co nahrazovat (v sekci č. `Shdr->sh_link`)
  - `r_addend` (jen `RELA`) – kolik přičíst k symbolu
- `gelf_getrel(Elf_Data *data, int idx, GElf_Rel *dst)` a `gelf_getrela(Elf_Data *data, int idx, GElf_Rela *dst)`
  - `data` – dostaneme z `elf_getdata`
  - `idx` – x-tý relokační symbol
  - `dst` – návratová hodnota

Úkol

Výpis relokací

1. Pro všechny sekce typu `SHT_RELA`
2. Iterujte přes všechny relokace
3. Pro každou vypište
   - Č. sekce, která se bude měnit (`Shdr->sh_info`)
   - Pozici, kde se bude měnit (`r_offset`)
   - Č. symbolu, kterého adresa se použije (`GELF_R_SYM(r_info)`)
4. Vyzkoušejte na `.o` souboru

Sekce 2

Programové hlavičky

Programové hlavičky

- `readelf -l`
- Vytvářené linkerem
- Čtené jádrem/interpretem
- Mapování na sekce
- Předdefinované programové hlavičky:
  - `PT_PHDR`: samotná hlavička
  - `PT_INTERP`: interpretr (`.interp`)
  - `PT_DYNAMIC`: dynamická tabulka (`.dynamic`)
  - `PT_LOAD`: data, kód (`.text`, `.data`, ...)
  - ...

Úkol

Výpis programových hlaviček

1. Vypište si programové hlavičky (`readelf -l`)
2. Prostudujte, jak se mapují na sekce
3. Podívejte se, které sekce jsou pro zápis

- Počet: `int elf_getphdrnum(Elf *elf, size_t *dst)`
- Hlavička: `GElf_Phdr` (`gelf.h`)
  - Získání hlavičky:
    `GElf_Phdr *gelf_getphdr(Elf *elf, int idx, GElf_Phdr *dst)`
  - `p_type`: `PT_LOAD`, `PT_DYNAMIC`, `PT_INTERP`, ...
  - `p_offset`: offset v souboru
  - `p_vaddr`: kde bude v paměti
  - `p_filesz`: velikost v souboru
  - `p_memsz`: velikost v paměti (po načtení)

Úkol

Výpis programových hlaviček

1. Otevřete ELF soubor pro čtení
2. Vypište interpretr
   - Může se hodit `elf_rawfile`
3. Vypište programové hlavičky
   - Typ
   - Offset v souboru
   - Kde bude v paměti
   - Velikost v souboru a v paměti
4. Přeložte a vyzkoušejte
5. Porovnejte s `readelf -l`