PB173 – Binární programování Linux

III. Binární objektové soubory

Jiri Slaby

Fakulta informatiky
Masarykova univerzita

4. 10. 2016

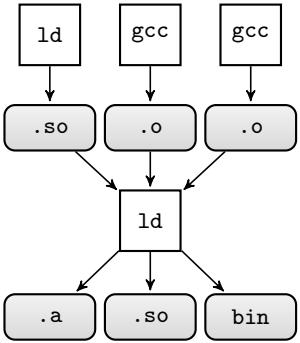
Obsah cvičení

Sekce 1

Objektové soubory

Objektové soubory

- Binární soubory s kódem (a daty)
- Mohou mít různý formát
- Výstup překladače
  - `gcc -c` produkuje `.o`
  - Obvykle nespustitelný objekt
- Výstup linkeru
  - Statická/dynamická knihovna
  - Spustitelná binárka



Obsah objektových souborů

- Obvykle
  - Identifikace souboru
    - ELF, PE, ...
  - Informace o souboru
    - Cílová architektura, struktura, ...
  - Kód
  - Data
  - Ostatní informace potřebné k běhu/linkování
    - Relokace, informace o symbolech atd.
- Ale také např.
  - Ladicí informace
  - Informace o překladači
- Tyto informace bývají v oddílech – tzv. sekcích

Úkol

Práce s objektovým souborem

1. Vyberte si nějaký objektový soubor
   - Např. `/bin/true`
2. Vypište si informace o souboru
   - `objdump -f`
3. Vypište si seznam sekcí
   - `objdump -h`
4. Vypište si dump sekce `.rodata`
   - `objdump -s -j .rodata`
5. Vypište si debug info
   - `objdump -Wi`
   - Pravděpodobně se nevypíše nic – proč?

Formáty objektových souborů

- Několik forem/standardů
  - a.out (UNIX/starý Linux)
  - *ELF* (Linux)
  - COM (DOS, čistě kód+data, nic víc)
  - COFF (UNIX, ELF ho převálcoval)
  - MZ (DOS .exe)
  - PE (Windows .exe, vychází z COFF)

Demo: DosBox a reboot přes COM (`ea f0 ff 00 f0`)

Úkol

Objdump neumí jen ELF

1. Stáhněte si nějakou Windows aplikaci
   - Cokoliv, co je .exe (PE), např. putty
2. Vypište si informace o souboru
   - `objdump -f`
3. Vypište si informace závislé na formátu
   - `objdump -p`
   - Zjistěte DLL, na kterých váš program závisí
4. Porovnejte sekce s předchozím úkolem

Sekce 2

libbfd

libbfd

- Knihovna pro práci s *různými* binárními formáty (`bfd.h`, v binutils)
  - a.out, ELF, PE, binární, ...
  - Seznam: `const char **bfd_target_list()`
- A různými architekturami
  - Seznam: `const char **bfd_arch_list()`
- Dokumentace
  - `https://sourceware.org/binutils/docs/bfd/`
  - *Některá užitečná makra bez dokumentace* (jen v `bfd.h`)
- Knihovny při překladu: `gcc ... -lbfd -liberty -ldl -lz`
- Nutno inicializovat pomocí `void bfd_init()`

Chyby

Jak zjistit, co se stalo?

- Poslední chyba: `bfd_error_type bfd_get_error()`
- Převod na text: `const char *bfd_errmsg(bfd_error_type error_tag)`

Typicky

```
if (!bfd_function (...) )
  errx(1, "bfd_function: %s", bfd_errmsg(bfd_get_error()));
```

Úkol

Výpis podporovaných formátů a architektur vaší `libbfd`

1. Doplňujte `pb173-bin/03/bfd.c`
2. Zavolejte `bfd_target_list`
3. Vypište návratové hodnoty (v cyklu)
   - Dokud nenarazíte na `NULL`
4. Zavolejte `bfd_arch_list`
5. Vypište návratové hodnoty (v cyklu)
   - Dokud nenarazíte na `NULL`
6. Přeložte a spusťte

Soubory

- Otevření
  - `bfd *bfd_openr(const char *file, const char *target)`
  - `bfd *bfd_openw(const char *file, const char *target)`
  - Jako `target` používejte `NULL`
  - Návratová hodnota se používá jako držátko
- Ověření
  - *Nutné před prací se souborem*
  - `bfd_boolean bfd_check_format(bfd *b, bfd_format fmt)`
  - Jako `fmt` používejte `bfd_object`
- Zavření
  - `bfd_boolean bfd_close(bfd *b)`

Příklad

```
b = bfd_openr("file", NULL);
if (!bfd_check_format(b, bfd_object))
  warnx("Not an object!");
bfd_close(b);
```

Úkol

Doplňte otevírání souboru zadaného jako parametr programu

1. Zavolejte `bfd_openr`
2. Zavolejte `bfd_check_format`
3. Zavolejte `bfd_close`
4. Ověřujte návratové hodnoty podle manuálu
   - Vypisujte chyby při neúspěchu
5. Přeložte a spusťte

Sekce

- Pro každou existuje struktura `asection`
  - Jméno: `bfd_get_section_name`
  - Velikost: `bfd_get/set_section_size`
  - Obsah: `bfd_get/set_section_contents`
  - A další: `flags`, `vma`, `lma`, ...
- Nalezení: `bfd_get_section_by_name`
- Iterace: `bfd_map_over_sections`
  - Háček, který se zavolá pro (skoro) každou sekci
- Vytvoření: `bfd_make_section_with_flags`
  - Flags: `SEC_HAS_CONTENTS`, pokud má sekce i obsah

Úkol

Hexdump sekcí (součást domácího)

1. Iterujte přes sekce (`bfd_map_over_sections`)
2. Vypište údaje o každé sekci
   - Název
   - Velikost
   - Flags
   - Hexdump prvních 16 bytů obsahu
3. Přeložte
4. Spusťte
   - Pro ELF
   - Pro PE