PB173 – Binární programování Linux

I. GIT a binární programování

Jiri Slaby

ITI, Fakulta informatiky

17. 9. 2013

Část I

Úvodní informace

Úvodní informace

- Semestr = 13 týdnů (22. 10. konference)
- Cvičící
  - Vývoj jádra od r. 2005 (NetBSD, Linux)
  - Student FI
- Cíle cvičení
  - Nastínit trochu jiný model programování
  - Prohloubit znalosti vnitřností OS a vrstvy pod jazykem
- Ukončení: k
  - Splnění *všech* domácích úkolů
  - 10 bodů na úkol, alespoň $\frac{3}{5}$ z celkového počtu
- Vše potřebné ve studijních materiálech v ISu

Bodování úkolů

Na začátku: 10 bodů za každý příklad

- -3 body za každý týden prodlení (termín je vždy do dalšího cvičení)
- -2 body za každé vrácení v případě nějaké nefunkčnosti
- -2 body za závažný problém (ve slidech značené POZOR)
- -1 bod za kód neodpovídající stylu
- -1 bod za ostatní drobnosti

S čím budeme pracovat?

HW

- Stroje satyr01–10
  - CentOS 6.x
  - Login/heslo: vyvoj/vyvoj
  - Nemají viditelnou IP

S čím budeme pracovat?

SW

- GIT
  - Úvod do GITu dnes
  - Podrobněji: `http://book.git-scm.com/`
- Zdroje jádra
  - GIT: `http://git.kernel.org`
  - LXR: `http://lxr.linux.no/`
- Zdroje glibc
  - GIT: `http://repo.or.cz/w/glibc.git`
  - LXR: `http://koala.cs.pub.ro/lxr/glibc/`
- ANTLR, BINUTILS, COREUTILS, ELFUTILS, ...

Část II

GIT

Úkol

Práce s GITem I.

1. Vytvořte si účet (pokud nemáte) na `github.com`
2. Proveďte fork
   - `https://github.com/jirislaby/pb173-bin`
3. Stáhněte si fork
   - `git clone git://github.com/<jmeno>/pb173-bin.git`
4. Prozkoumejte strukturu
   - Příklady ze cvičení
   - Adresář pro domácí úkoly

Záplatování

Výstup GITu = patch (záplata)

Záplaty v linuxovém jádře

1. Poslat patch
2. Poslat „pull request” a vystavit celý GIT strom

Stejným způsobem odevzdávání domácích úkolů
Ale ne přes github!

Úkol

Práce s GITem II.

1. Změňte cokoliv v souboru `sandbox/hello`
2. Zkontrolujte změny (`git diff --color`)
3. `git commit -a` (správný log: shrnutí na řádek, volný řádek, zdůvodnění; vzor na `git.kernel.org`)
   - Git může chtít nastavit jméno a e-mail (instrukce jsou na stdout)
4. Smažte `sandbox/hello` (`git rm`)
5. `git commit -a`
6. Zkontrolujte log, zda obsahuje 2 změny (`git log --color`)
7. Podívejte se na poslední 2 změny (`git show --color HEAD`, resp. `HEAD~1`)
8. Vyzkoušejte `git format-patch -2`
9. Proveďte push (`git push`)
10. Vygenerujte pull request (`git request-pull`)

Část III

Binární programování

Jen jádro a my

- Žádné *libc* (`printf`, `strlen`, `malloc`, `...`), ani ostatní (pthread)
- Jen rozhraní jádra a překladač
  - Naučíme se rozhraní používat
  - Nastíníme tvorbu překladače
  - Uvedeme si souborové formáty a budeme s nimi pracovat
  - Probereme si malé základy assembleru
  - ...

GCC

- Pro překlad C i assembleru
- C budeme psát ve stylu jádra (*CodingStyle*)
  - Kontrola: `scripts/checkpatch.pl` (není 100%)
- Důležité volby
  - `-E` – jen preprocesor
  - `-S` – generování assembleru
  - `-c` – generování objektových souborů
  - `-x` – definuje jazyk souboru (`-x c soubor1 -x assembler soubor2`)
  - `-O` – optimalizace (`-O0`, `-O2`, ...)
- INFO/PINFO
- Dokumentace: *Using the GNU Compiler Collection*

Úkol

Volání GCC

1. Pomocí roury přeložte `main`, který něco výpíše, do objektu
2. `echo -e '#include <stdio.h>\n int main(){ puts("Hello"); return 0; }' | ...`
3. Spusťte všechny tři fáze (`-E`, `-S` a `-c`) jako samostatné příkazy propojené rourami
4. Pozorujte `-E` a `-S` výstupy
5. Zapněte optimalizace `-O2` a pozorujte znovu

OBJDUMP/OBJCOPY

- Slouží k práci s objekty
- OBJDUMP vypisuje informace
  - `-D` disassembler
  - `-h` výpis sekcí souboru
  - `-b` formát vstupu
  - `-m` stroj vstupu
- OBJCOPY transformuje
  - `-I` formát vstupu
  - `-O` formát výstupu
- Společný důležitý argument
  - `-j` specifikace sekce

Úkol

Práce s objekty

1. Vypište si informace o vašem objektu z předchozího příkladu pomocí `objdump`
   - Za použití `-h`
2. Extrahujte `.text` sekci pomocí `objcopy`
   - Z předchozího výpisu zjistěte, která má nenulovou velikost
   - Vložte ji do souboru v binární podobě
   - Tj. použijete `-j .sekce` a `-O binary`
3. Disassemblujte tento binární výstup pomocí `objdump`
   - Musíte specifikovat formát (`-b binary`) a stroj (`-m i386:x86-64`)

Úkol

Práce s objekty II.

1. Prozkoumejte adresář `01` z pb173 git repozitáře
2. Je tam soubor `x.bin` podobně vytvořený jako váš (ve formátu `i386:x86-64`)
3. Je v něm funkce
   - Akceptuje dva `int` parametry a vrací `int`
   - Je „const” – tj. kromě parametrů nebere v úvahu nic
4. Zavolejte ji
5. Zjistěte, co dělá