

# PB173 – Binární programování Linux

## IX. Ladění

Jiri Slaby

Fakulta informatiky  
Masarykova univerzita

26. 11. 2015

### 1 Staticky

- Spouštím analyzátor a ten hledá nesrovnalosti
- Analýza ukazatelů, hodnot, velikosti zásobníku, ...
- Není cílem cvičení, ale *formela*

### 2 Dynamicky

- Spouštím na CPU a očekávám výsledek
- Testování
- Vyzkoušíme si dnes a příště

### 1 Ladění funkčnosti (dnes)

- Tracery
- ptrace
- valgrind

### Příště

- Pokračování s gdb
- Ladění výkonnosti/velikosti
  - Optimalizace kódu
  - Odstranění nepoužitého kódu

# Sekce 1

## Ladění funkčnosti (dnes)

- Výpisy
  - `printf` a příbuzné
- Monitorování chování
  - `gdb`, `ltrace`, `strace`, `valgrind`, ...
- Post-mortem
  - `core` soubor (obraz paměti)

- Sledují události a vypisují je
- `ltrace`: sleduje knihovní volání
- `strace`: sleduje systémová volání
- Používají systémové volání `ptrace`
  - `ltrace` si nastaví breakpointy na všechny externí symboly z ELFu
  - `strace` jednoduše použije `PTRACE_SYSCALL`
- Spuštění: `{s,l}trace binarka --volby`
- Další parametry
  - `-e`: filtr (`-e write`)
  - `-f`: sleduje i potomky
  - `-o`: výstup do souboru

## Práce s tracery

- 1 Projděte si `pb173-bin/09/tracer.c`
- 2 Spusťte nejdříve s `ltrace`
  - Kolik je volání `fwrite` do `libc`?
- 3 Spusťte s `strace`
  - Kolik je volání `write` do jádra?
  - $\Rightarrow$  `libc` bufferuje
- 4 Pomocí `ltrace` zjistěte, jak velký buffer `libc` používá
  - V cyklu vypisujte znak pomocí `fwrite`
  - Použijte volbu `-S`
- 5 Pomocí `strace` ověřte totéž
  - Pozorujte třetí parametr `write`

- `sys/ptrace.h`
- Systémové volání ke sledování procesů
  - Naváže se na proces
  - Ovládá ho
- `ptrace(enum __ptrace_request r, pid_t p, void *addr, void *data)`
  - `r` – co chceme
  - `p` – na kom to chceme provést
  - `addr` a `data` – význam závisí na hodnotě `request`



### Dvě možnosti

- 1 Svého potomka
  - Potomek: `ptrace(PTRACE_TRACEME)` a `raise(SIGSTOP)`
  - Rodič: `waitpid` a `ptrace(neco)`, např. `PTRACE_CONT`
  - `waitpid` počká na `raise` potomka, který už je p-trasovatelný
- 2 Cizí proces
  - Trasovač: `ptrace(PTRACE_ATTACH, pid)` a `waitpid`
  - Nastaví `pid` p-trasovatelný, pošle `SIGSTOP`, `waitpid` na signál počká

### Odpojení *zastaveného* trasovaného

`ptrace(PTRACE_DETACH, pid)`

## Trasování potomka

- 1 Proved'te fork
- 2 Potomek
  - Nastavte jej p-trasovatelného
  - Něco vypište
  - Pošlete mu signál
  - Něco vypište
- 3 Rodič
  - Počkejte na zastaveného potomka
  - Čekejte na standardní vstup
  - Něco vypište
  - Odpojte potomka
  - Něco vypište
- 4 Vyzkoušejte

- `ptrace(PTRACE_CONT)` – pokračuj ve vykonávání
- Přerušení klasickým `kill`
- `ptrace(PTRACE_SYSCALL)` – do příštího systémového volání
- `ptrace(PTRACE_SINGLESTEP)` – jedna instrukce
- Vždy je třeba počkat na signál od potomka

## Krokování potomka

- 1 V rodiči v cyklu:
  - Čekejte na standardní vstup
  - Krokujte potomka po instrukci
- 2 Nezapomeňte ošetřit stav, kdy potomek skončil
- 3 Vyzkoušejte

- `ptrace(PTRACE_PEEKTEXT)` – přečti kód
- `ptrace(PTRACE_PEEKDATA)` – přečti data
- `ptrace(PTRACE_PEEKUSER)` – přečti registr, ...
- `ptrace(PTRACE_GETREGS)` – přečti registry
  - `struct user_regs_struct`
- `ptrace(PTRACE_GETFPREGS)` – přečti plovoucí desetinné registry
  - `struct user_fpregs_struct`
- SET\* a POKE\* varianty

## Manipulace s procesem

- 1 V rodiči v cyklu navíc:
  - Vypište adresu aktuálně zastavené instrukce
  - První byte instrukce
- 2 Vyzkoušejte

- Sleduje (zejména) operace s pamětí (*Memcheck*)
  - Úniky paměti
  - Neinicializované a nezarovnané přístupy
  - Použití paměti po `free`
  - ...

- Několik dalších modulů
  - Callgrind: profiluje cache CPU
  - Massif: profiluje použití haldy
  - Další pro problémy se zámkami atd.

- Spuštění:

```
valgrind --volby_valgrindu binarka --volby_binarky
```

- Další parametry
  - `-tool`: výběr nástroje shora
  - `-leak-check`: sledovat úniky paměti

## Práce s nástrojem `valgrind`

- 1 Projděte si `pb173-bin/09/memcheck.c`
- 2 Spusťte v nástroji `valgrind`
- 3 Řiďte se pokyny na konci výpisu
  - Vypište si všechny podrobnosti
  - Přidáváním navržených parametrů („Rerun with“)