Laborator 6

Semnale

Sisteme de Operare

24 - 30 Martie 2011

Semnale

- 'Întreruperi software'
- Specifice UNIX, diverse forme de echivalență pe Windows
- Generate sincron
  - Acces invalid la memorie – `SIGSEGV` ('Segmentation fault'), `SIGBUS` ('Bus error')
  - Împărțire la 0 – `SIGFPE`
  - `abort()` - `SIGABRT`
  - Eroare la scrierea în pipe – `SIGPIPE` („Broken pipe")
- Generate asincron
  - Tastatură: `SIGINT` (`CTRL+C`), `SIGQUIT` (`CTRL+\`), `SIGSTOP` (`CTRL+Z`)
  - Sistem sau utilizator: `SIGTERM`, `SIGKILL`, `SIGUSR1`, `SIGUSR2`

Operații cu semnale

- Generare și transmitere
  - `CTRL+C`, `CTRL+\`
  - comanda `kill`, funcțiile `kill(2)`, `raise(3)`, `sigqueue(3)`)
  - direct de SO
- Blocarea unui semnal
  - `sigprocmask(2)`
- Așteptarea unui semnal
  - `pause(2)`, `sigsuspend(2)`
- Tratarea unui semnal
  - mascare, ignorare, acțiune implicită
  - asociere *signal handler*
  - `SIGKILL` si `SIGSTOP` termină procesul întotdeauna

Măști de semnale. Blocarea semnalelor

- mască pe biți reprezentând semnalele
- per proces
- `kill -l` (32 de procese obișnuite + 32 real-time)
- sigprocmask

Tratarea unui semnal

- `signal`
- `int sigaction(int signum, const struct sigaction *act, struct sigaction *oldact)`
  - sa_handler
  - sa_sigaction

Întrebări

- Ce semnale nu pot fi prinse sau ignorate?
- De ce nu este recomandat să apelăm malloc într-un handler de semnal?
- Ce se întâmplă dacă handler-ul implicit de tratare al SIGSEGV este înlocuit cu o funcție goală?
- În mod normal, un apel read blocat în așteptare de date, dacă este întrerupt de un semnal va seta errno la EINTR. Descrieți o situație în care totuși la primirea unui semnal apelul read se deblochează cu errno setat pe SUCCESS.