Laborator 6

Semnale

Sisteme de Operare

31 martie 2010

Outline

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 comunicarea 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
  - Sistem
  - Tastatură (`CTRL+C`, `CTRL+\`)
  - Utilizator (comanda `kill`, funcțiile `kill(2)`, `raise(3)`, `sigqueue(3)`)
- Blocarea unui semnal
  - `sigprocmask(2)`
- Așteptarea unui semnal
  - `pause(2)`, `sigsuspend(2)`
- Tratarea unui semnal
  - Implicit: ignorare, terminarea procesului
  - Asocierea unei rutine de tratare a semnalului (*handler*)
  - Semnalele `SIGKILL` si `SIGSTOP` termină procesul întotdeauna

Măști de semnale. Blocarea semnalelor

- `sigset_t`
  - Mască de biți reprezentând semnalele
  - `kill -l` (32 de procese obișnuite + 32 real-time)
  - Fiecare proces deține o mască de semnale
- `sigemptyset`, `sigdelset`, `sigaddset`, `sigismember` (`man sigsetops`)
- `int sigprocmask(int how, const sigset_t *set, sigset_t *oldset)`
  - Configurarea măștii de semnale a procesului
  - how = {`SIG_BLOCK`, `SIG_UNBLOCK`, `SIG_SETMASK`}

Așteptarea și tratarea unui semnal

- `int sigsuspend(const sigset_t *mask)`
  - Configurarea temporară a măștii procesului și așteptarea de semnale
- `int sigaction(int signum, const struct sigaction *act, struct sigaction *oldact)`
  - `signum` - numărul semnalului
  - `struct sigaction`
    - `sa_handler` este handler-ul
    - Dacă `sa_flags` conține `SA_SIGINFO`, se folosește `sa_sigaction`

- `unsigned int alarm(unsigned int seconds)`
  - Granularitate la nivel de secundă
  - Trimite `SIGALRM` la expirare
- `int setitimer(int which, const struct itimerval *value, struct itimerval *ovalue)`
  - Granularitate la nivel de nanosecundă
  - Trimite `SIGALRM`, `SIGVTALRM` sau `SIGPROF` la expirare
    - În funcție de parametrul `which` (`ITIMER_REAL`, `ITIMER_VIRTUAL`, `ITIMER_PROF`)

Outline

Evenimente de consolă

- `CTRL+C`, `CTRL+Break`, închiderea consolei
  - `CTRL_C_EVENT`, `CTRL_BREAK_EVENT`, `CTRL_CLOSE_EVENT`
- `SetConsoleCtrlHandler(...)`
  - permite configurarea unui handler pentru **toate** evenimentele de consolă
  - tipul handler-ului este `PHANDLER_ROUTINE`
  - Se pot înregistra mai multe handlere care acționează succesiv
    - Un handler întoarce `TRUE` dacă tratează sau `FALSE` dacă trimite mai departe

Timer-e

- Creare
  - `CreateWaitableTimer`
- Pornire
  - `SetWaitableTimer`
    - Argumentul 2 (`DueTime`) este de tip `LARGE_INTEGER`; folosiți `QuadPart` pentru inițializare
    - Timeout-ul este o valoare negativă; când ajunge la 0, timer-ul expiră
    - Se poate specifica un APC sau se poate aștepta semnalul
- Așteptare
  - `WaitForSingleObject`, dacă nu a fost specificat un APC
  - `SleepEx(INFINITE, TRUE)`, pentru așteptarea expirării timer-ului și rularea APC-ului

Outline

Întrebări (1)

- Care din următoarele semnale NU poate fi ignorat în Linux?
  - `SIGTERM`
  - `SIGSTOP`
  - `SIGINT`
  - `SIGQUIT`
- Care este apelul echivalent lui `kill(2)`?
  - `sigaction(2)`
  - `sigqueue(2)`
  - `sigsuspend(2)`
  - `sigprocmask(2)`

Întrebări (2)

- Care din următoarele semnale NU poate fi transmis folosind combinații de taste?
  - `SIGHUP`
  - `SIGINT`
  - `SIGQUIT`
  - `SIGSTOP`
- Considerând `setitimer` echivalentul `SetWaitableTimer`, care este echivalentul `WaitForSingleObject`?
  - `wait`
  - `waitpid`
  - `sleep`
  - `sigsuspend`

Întrebări (3)

- Care din următoarele reprezintă o caracteristică a semnalelor de timp real (real-time)?
  - pot fi tratate doar cu `sigaction(2)`, dar nu si `signal(2)`
  - mai multe instanțe de semnal recepționate pot fi reținute într-o coadă de semnale
  - fiecare semnal are un rol predefinit
  - nu pot fi folosite în aplicații multithreaded