Signal

Giảng Viên: Lưu An Phú

Agenda

- Tổng quan về signal
- Xử lý khi nhận signal
- Signal table
- Một số signal cơ bản
- Send signal
- Thực hành

Tổng quan về signal

- Là software interrupt
- Là phương pháp giao tiếp phổ biến nhất giữa các app
- Không thể lường trước được lúc nào thì nhận được signal
- Có 32 giá trị khác nhau, mỗi giá trị được gán 1 tên riêng biệt

Tổng quan về signal

- Các trường hợp xảy ra signal:
  - User sử dụng kill command
  - Process gửi signal
  - Chương trình gặp lỗi
  - User ấn 1 số tổ hợp phím đặc biệt

Xử lý khi nhận signal

- Phân loại:
  - Signal có thể ignore
  - Signal không thể ignore
  - Signal có thể chủ động điều khiển được
  - Signal không thể chủ động điều khiển được

Xử lý khi nhận signal

- Function tương ứng:
  - signal.h
  - sighandler_t signal(int signum, sighandler_t handler)
  - void sig_handler(int signo)
  - Thực hành:
    - Viết 1 chương trình C để in chữ hello world mỗi khi user nhấn phím Ctrl + C
    - Viết 1 chương trình C sleep vô thời hạn, sử dụng 1 chương trình C khác gửi signal để wakeup chương trình C đầu tiên lên.

Signal table

- Mỗi một process sẽ có 1 bảng signal table
- Hệ thống tự tạo ra bảng signal table khi khởi tạo process
- Mỗi ô trong bảng là 1 struct 2 phần tử, 1 phần tử là signal number, 1 phần tử là địa chỉ của signal handler
- Khi mới được khởi tạo, tất cả các signal hander sẽ là default handler

Signal table

- Một số signal cơ bản
  - SIGCHLD
  - SIGILL
  - SIGINT
  - SIGKILL
  - SIGSEGV

Những function an toàn khi gọi trong signal handler

- Signal handler được gọi tại bất cứ thời điểm nào
- Signal handler break luồng thực thi hiện tại của chương trình chính
- Có thể gây xung đột dữ liệu với chương trình chính
- Nếu xảy ra bug xung đột dữ liệu trong signal handler, đó sẽ là random bug
- Một số function không nên sử dụng trong signal hanler

aio_suspend	fcntl	mkdirat	setgid	tcflow
alarm	fdatasync	mkfifo	setpgid	tcflush
bind	fexecve	mkfifoat	setsid	tcgetattr
cfgetispeed	fork	mknod	setsockopt	tcgetpgrp
cfgetospeed	fstat	mknodat	setuid	tcsendbreak
cfsetispeed	fstatat	open	shutdown	tcsetattr
cfsetospeed	fsync	openat	sigaction	tcsetpgrp
chdir	ftruncate	pause	sigaddset	time
chmod	futimens	pipe	sigdelset	timer_getoverrun
chown	getegid	poll	sigemptyset	timer_gettime
clock_gettime	geteuid	posix_trace_event	sigfillset	timer_settime
close	getgid	pselect	sigismember	times
connect	getgroups	raise	signal	umask
creat	getpeername	read	sigpause	uname
dup	getpgrp	readlink	sigpending	unlink
dup2	getpid	readlinkat	sigprocmask	unlinkat
execl	getppid	recv	sigqueue	utime
execle	getsockname	recvfrom	sigset	utimensat
execv	getsockopt	recvmsg	sigsuspend	utimes
execve	getuid	rename	sleep	wait

Send signal

- int kill(pid_t pid, int signo)
  - pid > 0, pid < 0, pid == 0, pid == -1
  - int raise(int signo)
  - kill command
  - Permission to send a signal

Mask, unmask, block, unblock a signal

- Tổng quan:
  - Process can mask signal nào nó muốn nhận, chỉ có signal có cùng mask mới được kernel gửi đến cho process
  - Đôi khi trong 1 thời điểm, process không muốn xử lý 1 số signal nào đó, nó có thể block signal đó. Signal vẫn được gửi đến cho process nhưng sẽ nằm trong hàng chờ, ko được xử lý ngay.

Mask, unmask, block, unblock a signal

- Initialize a sigset:
  - int sigfillset(sigset_t *set)
  - int sigaddset(sigset_t *set, int signo)
  - int sigdelset(sigset_t *set, int signo)
  - int sigismember(const sigset_t *set, int signo)

Mask, unmask, block, unblock a signal

- Chặn và bỏ chặn 1 signal:
- int sigprocmask(int how, const sigset_t *set, sigset_t *oldset)
  - SIG_BLOCK
  - SIG_UNBLOCK
  - SIG_SETMASK

Mask, unmask, block, unblock a signal

- Thực hành:
  - Viết 1 chương trình C để unset hoặc block Ctrl + C signal từ user
  - Viết 1 chương trình C để in ra các signal mask của process hiện tại

Mask, unmask, block, unblock a signal

- Kiểm tra các signal đang bị pending
  - int sigpending(sigset_t *set)
  - Một signal đang bị block thì kernel vẫn gửi nó đến cho process nhưng signal đó sẽ bị pending trong hàng đợi mà không được xử lý.
  - Thực hành:
    - Viết 1 chương trình C block SIGINT, sau đó check trong số signal pending có SIGINT không, nếu có thì thoát chương trình

Thực hành

- Viết chương trình A fork ra chương trình B. A và B cùng ghi vào 1 file. Mỗi chương trình sẽ đọc lần lượt dòng counter cuối cùng của file và tăng nó lên 1, sau đó ghi tiếp 1 dòng mới là process ID của chính nó kèm biến counter mới. Phải sử dụng signal để đồng bộ giữa 2 process

Thank you