TP2 en Programmation système UNIX

Les signaux

Exercice1 :

a. Ecrire une application qui génère un processus fils. Ce dernier :

- Attend une seconde,
- Affiche son numéro de processus ainsi que celui de son père,
- Effectue un travail quelconque,
- Attend 20 secondes.
- Affiche un message indiquant sa fin puis,
- Exécute un exit(x). x étant un entier < 256 à communiquer au père.

Le processus père à son tour, affiche son numéro de processus puis se met en attente de la fin de son fils.

La syntaxe de la primitive wait est la suivante : ***int wait(int *Y)***

wait retourne le numéro du processus fils ayant émis le signal au père Le paramètre passé sert pour récupérer le résultat de la fin du fils. Y étant d'une taille de 16 bits structurés de la façon suivante :

si l'octet de poids faible (b7-b0) vaut zéro

alors Le fils a eu une fin normale et

l'Octet de poids fort (b15-b8) contient de résultat retourné par le fils. (*x*).

sinon Le fils s'est terminé d'une façon anormale (suite à un signal). Les bits b0 à b6 contiennent un entier (numéro du signal qui a tué le fils). Cependant si le bit 7 vaut 1, un fichier *core* est créé.

b. Tester votre application des différentes façons suivantes :

- Fin normale du fils,
- Le fils effectue une division par zéro,
- Vous tuer le fils avant sa fin en utilisant la commande **kill**.

Exercice2 :

Ecrire un programme en C qui :

- Demande à l'utilisateur de saisir deux lignes de commandes SHELL

N.B. Afin de ne pas attendre indéfiniment, le signal **SIGALRM** est utilisé pour que le programme ne soit pas bloqué plus de **5 secondes** avant la saisie

- Le père crée deux fils et leurs envoie **SIGUSR1** pour fils1 et **SIGUSR2** pour fils2. Lorsque le fils1 (resp. le fils2) reçoit un signal, il passe à l'exécution de la 1ère ligne de commandes (resp. la 2ème ligne de commande).

Exercice3 :

On vous demande d'écrire un premier programme dans lequel un processus père crée 3 processus fils. Le processus père utilise les signaux **SIGSTOP** et **SIGCONT** pour suspendre (bloquer) et reprendre (débloquer) l'exécution de ses processus fils.

Chaque processus fils entre dans une boucle infinie et affiche son numéro et son PID. Le processus père répète continuellement le traitement suivant en commençant par le premier fils : il envoie le signal SIGCONT à un fils puis s'endort pendant **5 secondes**. À son réveil, il envoie SIGSTOP au même fils et SIGCONT au fils suivant (le fils suivant du dernier est le premier).

Lorsqu'un processus fils reçoit le signal SIGCONT, il affiche le message indiquant qu'il a capturé le signal SIGCONT avant de poursuive son exécution.