TP n°5-6 : Processus et parallélisme

Objectif : Visualiser et manipuler des processus UNIX

Exercice 1 – Visualisation de processus

Pour voir quels processus tournent sur une machine a un moment donné, il faut utiliser la commande `ps`.

1) Ouvrir deux terminaux. Dans le premier terminal, lancer 2 applications, par exemple *firefox* et *gedit* à l'aide des commandes `firefox &` et `xemacs &`. Dans le deuxième terminal, tapez la commande `ps`.

Que se passe-t-il ? Pourquoi `firefox` et `gedit` n'apparaissent-ils pas dans la liste ?
Quelle option utiliser avec `ps` pour les voir ?

2) Utilisez la commande `ps` pour déterminer le PID (*Process ID*) du `firefox` que vous avez lancé. Tapez `kill -9 lepiddefirefox`.

Que se passe-t-il ? Déterminez le PID d'une des commandes bash et arrêtez-la à l'aide de la commande `kill -9`. Pourquoi la fenêtre du terminal disparaît-elle ?

3) Tapez `firefox` dans le premier terminal.

Pouvez-vous exécuter d'autres commandes dans ce terminal ? Pourquoi ? Faites un Ctrl-C. Quel processus a été tué ?

Exercice 2 – La fonction fork()

Programmer l'exercice 1 du TD n°5 « Processus ».

1) En exécutant la commande `ps -aux` depuis un autre terminal, observez l'état des processus. Pourquoi ne visualise-t-on pas l'état *Running* ?

2) Modifiez le code de manière à pouvoir observer en alternance les états *Sleeping* et *Running*.

3) Envoyez aux processus le signal `STOP` avec la commande `kill -STOP n°PID`. Quel état observe-t-on pour les processus ?

4) Envoyez-leur de la même manière le signal `CONT`. Quel est l'effet de cette commande sur les processus ?

Exercice 3 – La fonction execl()

1) Donnez le code source C d'un programme `affichez.c` qui affiche à l'écran la chaîne de caractères qui lui est passée en paramètre en ligne de commande.

Exemple d'utilisation : `affichez coucou`

Compiler et tester ce programme.

2) Ecrire un programme `prog1` qui crée un processus fils qui exécute `affichez` avec l'argument *salut*. On utilisera la fonction `execl`.

Exercice 4 – La fonction kill()

Ecrire un programme qui crée un processus fils qui affiche à chaque seconde le nombre de secondes écoulées. Le processus père arrête le processus fils au bout de 10 secondes.

Exercice 5 – La fonction wait()

Ecrire un programme qui crée 2 processus, l'un faisant la commande `ls -l`, l'autre `ps -l`. Le père devra attendre la fin de ses deux fils et afficher quel a été le premier processus à terminer.