Enseignante : **Amani Braham**
Mail : **amani.braham@issatso.u-sousse.tn**

TP03 : Gestion des processus

Ce TP a pour objectif de :

- Maîtriser les commandes de gestion des processus.
- Savoir observer les processus s'exécutant sur une machine.
- Manipuler un processus en cours d'exécution.
- Savoir tuer un ou plusieurs processus.

Pour ce TP vous devez rédiger un compte-rendu (nom_prénom_groupe_CTP03.docx) qui sera envoyé par mail à la fin de la séance. (Précisez dans l'objet : TP03-Prepa-A2-Groupe)

Rappel et Révision

1. Introduction

On appelle **processus** un objet dynamique correspondant à l'exécution d'une suite d'instructions : un programme qui s'exécute, ses données, ainsi que d'autres informations sur son contexte d'exécution.

Un processus possède les caractéristiques qui permettent au système de l'identifier. Parmi ces caractéristiques :

- Etat : exécution, suspendu, etc.
- Identifiant du processus.
- Identifiant du processus qui lui a donnée naissance : processus parent.
- Identifiant de l'utilisateur qui l'a lancé.
- Compteur ordinal : indique la prochaine instruction à exécuter.
- Pile d'exécution : mémorise l'empilement des appels de fonction.
- Données en mémoire.
- Etc.

Un processus s'exécute soit en avant-plan (**foreground**), soit en arrière-plan (**background**).

2. Commandes Shell de gestion de processus :

- *La commande ps :* ***$ ps [options]***

Permet d'obtenir la liste des processus actifs sur le Shell.

```
amani@debian:~$ ps
  PID TTY          TIME CMD
 1591 pts/0    00:00:00 bash
 1606 pts/0    00:00:00 ps
```

L'option **–e** donne des informations sur tous les processus en cours et l'option **–f** permet d'avoir des informations détaillées sur les processus.

```
amani@debian:~$ ps -ef
UID          PID    PPID  C STIME TTY          TIME CMD
root           1       0  0 12:57 ?        00:00:02 /sbin/init
root           2       0  0 12:57 ?        00:00:00 [kthreadd]
root           3       2  0 12:57 ?        00:00:00 [ksoftirqd/0]
root           5       2  0 12:57 ?        00:00:00 [kworker/0:0H]
```

COLONNE	***DEFINITION***
UID	Identifiant de l'utilisateur (User ID).
PID	Identifiant du processus (Process ID).
PPID	Identifiant du processus parent (Parent Process ID).
C	Facteur de priorité (plus la valeur est grande plus la priorité est élevée).
STIME	Heure de lancement du processus.
TTY	Nom du terminal depuis lequel le processus a été lancé.
TIME	Durée de traitement du processus.
CMD	Commande exécutée.

L'option **–u** permet d'obtenir la liste des processus lancés par un utilisateur particulier.

```
amani@debian:~$ ps -u amani
   PID TTY          TIME CMD
  1173 ?        00:00:00 systemd
  1174 ?        00:00:00 (sd-pam)
  1180 ?        00:00:00 gnome-keyring-d
  1183 tty2     00:00:00 gdm-x-session
  1185 tty2     00:00:04 Xorg
```

- *La commande pstree :* ***$ pstree***

Permet de visualiser l'arborescence des processus.

```
amani@debian:~$ pstree
systemd─┬─ModemManager─┬─{gdbus}
        │              └─{gmain}
        ├─NetworkManager─┬─dhclient
        │                ├─{gdbus}
        │                └─{gmain}
        ├─accounts-daemon─┬─{gdbus}
        │                 └─{gmain}
        ├─avahi-daemon───avahi-daemon
        ├─bluetoothd
        ├─colord─┬─{gdbus}
        │        └─{gmain}
        ├─cron
        ├─cups-browsed─┬─{gdbus}
        │              └─{gmain}
        ├─cupsd
        ├─dbus-daemon
```

- *La commande top :* ***$ top [options]***

Permet de gérer les processus en temps réel (visualisation dynamique des processus).

```
top - 14:25:34 up  1:27,  1 user,  load average: 0.00, 0.00, 0.00
Tasks: 165 total,   1 running, 164 sleeping,   0 stopped,   0 zombie
%Cpu(s):  0.4 us,  0.3 sy,  0.0 ni, 99.2 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
KiB Mem :  2033712 total,   919620 free,   602360 used,   511732 buff/cache
KiB Swap:  2094076 total,  2094076 free,        0 used.  1274344 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU %MEM     TIME+ COMMAND
 1284 amani     20   0 2427768 224064  82960 S   2.7 11.0   0:50.19 gnome-she+
 1185 amani     20   0  364224  45160  29396 S   1.0  2.2   0:06.34 Xorg
 2233 amani     20   0  599976  32504  24364 S   0.3  1.6   0:00.44 gnome-ter+
 2248 amani     20   0   44920   3656   3052 R   0.3  0.2   0:00.03 top
    1 root      20   0  204552   6988   5368 S   0.0  0.3   0:02.22 systemd
```

Dès que top est lancée, il est possible d'exécuter des commandes interactives :

- N : Classer les processus par PID.
- A : Classer les processus dans l'ordre chronologique.
- P : Classer les processus par rapport à leur utilisation CPU.
- M : Classer les processus par rapport à leur utilisation de la mémoire.
- k : Tuer un processus (PID sera demandé).
- q : Quitter l'utilitaire top.

- *La commande kill:* ***$ kill [-l] -Num_signal PID [PID2...]***

Permet d'envoyer des signaux à un processus dont on connaît son identifiant. Cette commande ne sert pas seulement à « tuer » un processus ; on peut lister les signaux disponible avec la commande ***kill –l***.

```
amani@debian:~$ kill -l
 1) SIGHUP       2) SIGINT       3) SIGQUIT      4) SIGILL       5) SIGTRAP
 6) SIGABRT      7) SIGBUS       8) SIGFPE       9) SIGKILL     10) SIGUSR1
11) SIGSEGV     12) SIGUSR2     13) SIGPIPE     14) SIGALRM     15) SIGTERM
16) SIGSTKFLT   17) SIGCHLD     18) SIGCONT     19) SIGSTOP     20) SIGTSTP
21) SIGTTIN     22) SIGTTOU     23) SIGURG      24) SIGXCPU     25) SIGXFSZ
26) SIGVTALRM   27) SIGPROF     28) SIGWINCH    29) SIGIO       30) SIGPWR
31) SIGSYS      34) SIGRTMIN    35) SIGRTMIN+1  36) SIGRTMIN+2  37) SIGRTMIN+3
38) SIGRTMIN+4  39) SIGRTMIN+5  40) SIGRTMIN+6  41) SIGRTMIN+7  42) SIGRTMIN+8
43) SIGRTMIN+9  44) SIGRTMIN+10 45) SIGRTMIN+11 46) SIGRTMIN+12 47) SIGRTMIN+13
48) SIGRTMIN+14 49) SIGRTMIN+15 50) SIGRTMAX-14 51) SIGRTMAX-13 52) SIGRTMAX-12
53) SIGRTMAX-11 54) SIGRTMAX-10 55) SIGRTMAX-9  56) SIGRTMAX-8  57) SIGRTMAX-7
58) SIGRTMAX-6  59) SIGRTMAX-5  60) SIGRTMAX-4  61) SIGRTMAX-3  62) SIGRTMAX-2
63) SIGRTMAX-1  64) SIGRTMAX
```

SIGNAL	***DESCRIPTION***
SIGSTOP (19)	Arrêter un processus.
SIGCONT (18)	Continuer un processus arrêté.
SIGTERM (15)	Signifier au processus qu'il doit se terminer.
SIGKILL (9)	Tuer un processus.

- *Commandes pour manipuler les jobs :*

Il est possible de démarrer plusieurs processus appelés aussi « **jobs** ».

***&* :** Démarrer un processus en arrière-plan.

```
amani@debian:~$ xeyes &
[1] 2447
```

jobs : Afficher la liste des tâches du Shell courant.

```
amani@debian:~$ jobs
[1]+  Running                 xeyes &
```

fg %n : Relancer l'exécution d'un processus en arrière-plan en un processus en avant-plan.

```
amani@debian:~$ fg %1
xeyes
```

Ctrl* + *Z : Suspendre un job

```
^Z
[1]+  Stopped                 xeyes
amani@debian:~$
```

bg %n : Relancer l'exécution d'un processus suspendu en processus en arrière-plan.

```
amani@debian:~$ bg %1
[1]+ xeyes &
amani@debian:~$
```

kill %n : Tuer un job.

```
amani@debian:~$ kill %1
amani@debian:~$
```

Pour lancer un processus en arrière-plan, on utilise le symbole **&.** Lorsqu'on ferme la console, le processus est arrêté. Pour éviter qu'un processus ne s'arrête lorsque son parent se termine, on utilise la commande **nohup**.

- *Commande time:* ***$ time***

Permet de mesurer les durées d'exécution d'une commande processus et retourner trois valeurs (real, user, system).

```
amani@debian:~$ time ls -l tp3
total 0

real    0m0.002s
user    0m0.000s
sys     0m0.000s
```

Travail Demandé

A. Exercices de prise en main

1) Lister tous les processus lancés sur le système. Affichez la liste des processus dont vous êtes propriétaire.
2) a- Lister de nouveau les processus lancés sur le système de façon détaillée.
   b- A quoi correspond l'information **STIME** ?
   c- A quelle heure votre machine a-t-elle démarré ?
3) Affichez la hiérarchie des processus s'exécutant sur le système.
4) Quel signal est lancé par défaut à la commande **kill**?

5) Afficher en temps réel les informations sur l'ensemble de processus exécutés sur votre machine. Quelles informations sont affichées par défaut ?
6) a- Consulter le man de **nice** et **renice**. A quoi sert les commandes **nice** et **renice**?
   b- La commande **top** consomme des ressources. Faites en sorte que sa priorité soit de « 19 ».
7) a- Consulter le man de **ps**. A quoi sert cette commande ? Placer le processus en arrière-plan sans le terminer.
   b- Consulter le man de **kill**. A quoi sert cette commande ? Placer le processus en arrière-plan sans le terminer.
   c- Lister les tâches lancées par le Shell.
   d- Réafficher le man de **ps**.
   e- Ouvrez un deuxième terminal et envoyez le signal **SIGKILL** au processus affichant le man de **ps**.
   f- Retourner dans le premier terminal, lister les tâches lancées par le Shell, et réactiver le man de **kill**.

B. Exercices avancés

Exercice 1

1) Dans un système Linux, quel est le processus qui n'a pas un père ? Préciser son PID.
2) Utiliser la commande **ps** et trouver le PID du processus **acpid.**
3) Utiliser la commande **pstree** pour trouver le PID du processus **acpid** et le nom de son processus père.
4) Consulter la documentation de la commande **yes**. A quoi sert cette commande ?
5) Lancer la commande: **$ yes "Good morning, my name is yourname !".**
6) Pendant l'exécution de la commande **yes**, lancer la commande **top** dans un autre terminal. Quels sont les processus qui consomment le plus le CPU ?

Exercice 2

1) a- Lancer les commandes **sleep 4444**, **sleep 3333** et **sleep 2222** en arrière-plan.
   b- Que signifie ce qui est retourné à l'écran ?
2) Afficher la liste des tâches en cours.
3) Que signifient les caractères **+** et **-** dans la liste précédente ?
4) Lancer la commande **sleep 1111** en avant-plan.
5) Interrompre le processus en avant plan en utilisant **Ctrl+Z**. Quel est le numéro de travail du processus interrompu ?

6) Afficher à nouveau la liste des tâches en cours.
7) Relancer en arrière-plan la dernière commande **sleep**.
8) Arrêter l'exécution de la première commande **sleep** en utilisant son numéro de travail.
9) Afficher à nouveau la liste des tâches en cours.
10) Arrêter l'exécution de la deuxième commande **sleep** en utilisant son PID.
11) Afficher à nouveau la liste des tâches en cours.
12) Passer en avant-plan les dernières commandes **sleep** et arrêter l'exécution de ces commandes.
13) Afficher à nouveau la liste des tâches en cours.

Exercice 3

1) Lancer la commande **xeyes**.
2) Pouvez-vous exécuter une autre commande dans le même Shell ? Justifiez.
3) Suspendre l'application **xeyes**.
4) Ramener l'application en avant-plan.
5) Suspendre à nouveau l'application **xeyes**.
6) Continuer l'exécution de l'application **xeyes** tournant en arrière-plan.
7) Pouvez-vous exécuter une autre commande dans le même Shell ? Justifiez.
8) Lancer la commande **gedit** en avant-plan. Fermer la fenêtre de l'interpréteur de commandes à partir de laquelle l'application est lancée. Que se passe-t-il ?
9) Lancer la commande **gedit** en arrière-plan. Fermer la fenêtre de l'interpréteur de commandes à partir de laquelle l'application est lancée. Que se passe-t-il ?
10) Comment faire pour éviter qu'un processus ne s'arrête lorsque son parent se termine ? Tester avec la commande **gedit**.