Université de N'Djaména
Faculté des Sciences Exactes et Appliquées **Master 1**
Département d'Informatique

Fiche de TP4

L'objectif de ce TP est de travailler sur les processus sous UNIX/Linux (créer, gérer et détruire les processus d'une machine)

Exercice 1 : La commande grep

Le répertoire `/proc/` contient des informations sur le système d'exploitation en cours d'exécution, les processus, etc. Dans ce répertoire, le fichier `cpuinfo` contient des informations sur le type de cpu sur lequel fonctionne le système.

A l'aide de la commande `grep` et de ses options déterminez le modèle de processeur, ainsi que le nombre de processeur dans le cpu (c'est à la commande `grep` de compter). Vous pourrez dans un premier temps utiliser la commande `less` ou `more` afin de savoir quel critère chercher.

Exercice 2 : La commande du

La commande `du` permet d'afficher une estimation de l'espace disque utilisé par ou plusieurs fichiers.

1. Créez le fichier vide `/tmp/fichier_vide` à l'aide de la commande `touch`.
2. Quelle est la taille de ce fichier telle que donnée par la commande `ls` ?
3. Quelle est la taille de ce fichier telle que donnée par la commande `du` ?
4. Écrivez le texte « quelques mots » dans le fichier précédent. Quelles sont maintenant les tailles données par les commandes `ls` et `du` ?

Exercice 3 : La commande `sort`

La commande `sort` permet d'effectuer un tri sur les données qu'elle prend en entrée, selon des critères que l'on peut lui préciser.
Soit les données suivantes :

E 1
A 5
C 3
D 2
B 4

1. Triez ces données selon l'ordre lexicographique inverse des éléments de la première colonne.
2. Triez maintenant ces données selon l'ordre numérique inverse des éléments de la seconde colonne.

Exercice 4 : Les commandes `head` et `tail`

La commande `head` permet d'afficher que les premières lignes d'un fichier ou de données. Le nombre de lignes à afficher dépend d'un paramètre de cette commande.
La commande `tail` fait de même que `head` mais en retenant cette fois que les dernières lignes.

1. Affichez les 2 premières lignes des données précédentes (commande `sort`) à l'aide de la commande `head`.
2. Affichez la dernière ligne des donnée précédentes à l'aide de la commande `tail`.
3. Affichez tout sauf les 3 dernières lignes du fichier `/etc/group` à l'aide de la commande `head`.

Exercice 5 : Création des processus

Dans un terminal, il est possible de créer un nouveau processus indépendant dit « enfant » à partir d'un processus « parent ». Pour cela il suffit d'ajouter le caractère spécial "`&`" après le nom d'une commande (on dit aussi *lancement en mode détaché*). Avant de revenir au *prompt* (invite de commande shell), apparaît un numéro de tâche et le numéro du processus lancé (Pid). Ce numéro identifie de manière unique le processus lancé.

Expérimentez la création de nouveaux processus avec cette commande en lançant plusieurs processus différents ou identiques à partir d'un terminal (un éditeur (`gedit`) ou une calculatrice (`gnome-calculator`) ou une horloge (`xclock`)).

Exercice 6 : La commande `ps`

Cette commande permet d'obtenir une liste de processus respectant un critère et en afficher certaines caractéristiques.

1. Étudiez les différentes options disponibles, en particulier `-l`, `-e`, `-f`.
2. Quelle est l'option permettant d'afficher tous les processus du système ?
3. Quelle est l'option permettant d'afficher uniquement la liste de tous les processus qui vous appartiennent ?
4. Repérez les états des processus.
5. Repérez la priorité du processus de votre terminal.
6. Repérez la valeur "`nice`" du processus de votre terminal.
7. Pourquoi dans la liste de vos processus apparaît-il la commande `ps` ?
8. Comment peut-on repérer des processus enfants d'autres processus ?
9. Quel est le processus parent de tous les autres ?
10. Créez de nouveaux processus avec le caractère spécial "`&`" à partir du terminal et vérifiez la relation parent - enfant.

Exercice 7 : La commande `pstree`

Cette commande permet de visualiser sous forme graphique les relations parent-enfant dans la liste des processus du système.

NB: Elle n'est pas installée par défaut dans tous les systèmes.

1. Étudiez cette commande.
2. Quelles sont les options pour afficher le PID des processus et pour afficher uniquement vos processus ?

Exercice 8 : Manipulation des processus entre l'avant plan (premier plan) et l'arrière-plan

Les commandes "bg", "fg", "jobs" et Ctrl-Z permettent de manipuler la notion d'avant plan et d'arrière-plan pour un processus. Elles sont intimement liées au *SHELL*.

Un processus en avant plan s'exécute en ayant la main sur le terminal qui l'a lancé (le bloque). Un processus en arrière-plan s'exécute de façon détachée par rapport au terminal qui l'a lancé (pas de blocage).

La commande "jobs" permet d'afficher les processus activés ou suspendus (*STOPPED*) et leur numéro. Attention, cette commande n'affiche que les processus lancés à partir du terminal où elle est exécutée.

Pour suspendre l'exécution d'un processus en avant plan et revenir au terminal, il existe la séquence de touches Ctrl-Z. Ensuite, à partir de ce même terminal, il est possible :

- soit de remettre en avant plan une tâche : commande "fg %n° de job" ou "fg n° de job" ;
- soit de lancer en mode détaché une tâche : commande "bg %n° de job"

1. À partir d'un terminal, ouvrez une fenêtre horloge avec la commande "xclock -update2". Que constatez-vous ?
2. Suspendez-la avec la séquence de touches Ctrl-Z. Que constatez-vous ?
3. En utilisant la commande "jobs", déterminez quel est l'état de ce processus.
4. Relancez-le en arrière-plan avec la commande "bg". Que se passe-t-il ?
5. Vérifiez avec la commande "jobs"

Exercice 8 : Envoi de messages aux processus

La commande "`kill`" permet d'envoyer à un ou plusieurs processus un signal prédéfini dans le système (par ex. `SIGINT`, …) en le(s) référençant par son *PID*. L'utilisation de la commande "`kill`" est limitée aux processus appartenant à l'utilisateur sauf pour l'utilisateur `root` qui peut envoyer des signaux à tous les processus.

1. Expérimentez la commande "`kill -l`". Qu'affiche-t-elle ? Pour comprendre cet affichage, ouvrez avec `gedit` le fichier "/usr/include/x86_64-linux-gnu/bits/signum.h".
2. Quel est le signal pour suspendre l'exécution d'un processus ? Faites un test avec ce signal sur un nouveau processus que vous avez créé.
3. Quel est le signal (ou les signaux) pour arrêter définitivement un processus ?
4. On peut, à la place du *PID*, utiliser le numéro de job affiché avec la commande `jobs`. Créez un processus détaché (`xclock` par exemple) et envoyez-lui un signal `SIGKILL` avec son numéro de job (`%n°`).
5. D'autres commandes basées sur `kill` permettent une utilisation parfois plus adaptée de l'envoie de signal, par exemple `killall`, `pkill` ou `xkill`. Lisez le manuel afin de comprendre les nuances entre ces commandes et expérimentez-les.

Exercice 9 : Mesure du temps d'exécution des processus

La commande "`time`" permet de mesurer le temps d'exécution d'un processus.

1. Expérimentez la commande "`time`". À quoi correspondent les valeurs de temps mesurées ?
2. Exécutez la commande "`time gedit&`". Quand la commande time rend-elle son résulat ?
3. Pour la commande "`time ls -lR /usr/lib > /dev/null`", calculez le ratio temps CPU / temps réel.

Exercice 10 : La commande top

La commande `top` permet d'afficher en dynamique certaines valeurs du système (processus, temps, priorité, utilisation mémoire. Vous pouvez quitter cette commande avec la touche '`q`'. Par défaut la mise à jour des informations a lieu toutes les 3 secondes. Vous pouvez régler un grand nombre de paramètres avant le lancement de l'exécution (surveiller les processus d'un utilisateur particulier par exemple) ou les changer en cours d'exécution (touche '`h`'). Lire les premières pages du `man` pour découvrir la commande.

1. Lancez la commande `top` et visualisez les différents paramètres (mémoire physique, mémoire swap, *PID*, Priorité, *%CPU* et *MEM*).
2. Notez le nombre de processus `running`, `sleeping`, `stopped`, `zombie`.
3. Trouvez la touche permettant, en mode interactif, d'afficher le réglage des données à visualiser et affichez les données `DATA` et `CODE`.
4. Trouvez la touche permettant, en mode interactif, de sélectionner uniquement vos processus.

5. Lancez dans un autre terminal un explorateur internet. Observer le comportement des paramètres du processus avec la commande `top`. Quelle est la charge CPU et mémoire de ce processus ?
6. Arrêtez l' explorateur internet et relancez en regardant précisément la valeur de la colonne `PR` de ce processus dans les informations de `top`. Que constatez-vous ?

NB: il existe également la commande `htop` qui est assez similaire mais qui n'est pas forcément installée par défaut dans les distributions.

Exercice 11 : La commande at

La commande "`at`" permet de planifier le lancement de un ou plusieurs processus de façon automatique à une date donnée.

1. Expérimentez la commande "`at`".
2. Donnez la ligne de commande utilisant `at` permettant de rediriger la sortie de la commande "`ps -ef`" dans un fichier `ficsortie` dans 3 minutes ?
3. Quelle est la commande permettant de lister la liste des tâches en attente ?
4. Quelle est la commande permettant de supprimer une tâche en attente ?

NB: la commande "`at`" peut prendre ses commandes à partir d'un fichier ("`at -f …`").

Exercice 12 : La commande nice et renice

La commande Nice dans Linux aide à l'exécution d'un programme/processus avec priorité de planification modifiée. Il lance un processus avec une priorité de planification définie par l'utilisateur. Ainsi, si nous donnons à un processus une priorité plus élevée, alors le Noyau lui allouera plus de temps de processeur. La commande renice permet de modifier la priorité de planification d'un processus déjà en cours d'exécution.

1. Pour vérifier la valeur de nice d'un processus, exécuter ps -el | terminal grep
2. Définir la priorité d'un processus : nice -10 gnome-terminal
3. Définir une priorité négative d'un processus et examiner le résultat : nice --10 gnome-terminal
4. Modifier la priorité du processus d'exécution : sudo renice -n 15 -p pid
5. Modifier la priorité de tous les programmes d'un groupe spécifique : renice -n 10 -g 4
6. Pour modifier la priorité de tous les programmes d'un utilisateur spécifique : sudo renice -n 10 -u uid