Informatique Générale: UNIX

ESILV S1 ⇢ 2009-2010

TD6 - Gestion de processus, informations systèmes

Exercice 1 : Informations systèmes

Tout système d'exploitation permet aux utilisateurs de s'informer des propriétés et de l'état tant de l'ordinateur que du système. Linux fournit pour cela de nombreux outils. Pour débuter cette partie, affichez le fichier `/proc/cpuinfo` (via la commande cat). Quelle ligne affiche le nom du processeur ? Via la commande grep et une redirection | (le pipe), comment ne récupérer que cette information ? Comment récupérer le nombre de processeurs, en une seule commande ?

Exercice 2

Testez le code suivant dans votre terminal. Notez bien la back quote qui entoure date. Ensuite, ajoutez l'affichage du nombre et du type de processeurs à votre script monenv.sh.

```
vartemp=‘date‘
echo $vartemp
```

Exercice 3

Que fait la commande `free` ? En utilisant le fichier `/proc/meminfo`, comment obtenir la quantité de mémoire libre et la quantité de mémoire totale ?

Exercice 4 : Gestion de processus

Que fait la commande `top` ? Quel peut-être son usage ?

Exercice 5

La commande ps permet de lister les processus (ou tâches) exécutés. `ps -aux` permet par exemple de lister l'ensemble des processus, quelque soit l'utilisateur propriétaire de ce processus. le PID est l'identifiant unique de chaque processus. Quelle colonne indique le PID ? Après avoir lancé firefox, trouvez le PID de celui-ci. On pourra utiliser la commande grep.

Exercice 6

On va maintenant envoyer un signal à ce processus. La commande est `kill` envoie un signal

(souvent pour terminer le programme) à un processus. `kill -9 xxx` tue le processus avec le PID `xxx`. Lancez gimp puis tuez-le en utilisant la commande `kill`.

Exercice 7

Quelle est la différence, au niveau de l'état du terminal, entre ces deux lancements de l'éditeur emacs :

```
emacs
```

et

```
emacs &
```

L'usage du caractère & permet donc de lancer un processus en "tâche de fond" (ou arrière-plan, background) et de "garder la main" sur le terminal. Sans le &, on dit que le processus est lancé en avant-plan (foreground)

Exercice 8

Il est possible de basculer l'état d'un processus. Si un processus est lancé en foreground, il est possible de récupérer la main (le processus est alors "stoppé") puis de mettre le processus en background. Pour cela il faut taper `CTRL + Z` (touches Control et Z appuyées en même temps) puis `bg` suivi de enter. Testez la séquence suivante :

```
emacs
CTRL + Z (touches Control et Z appuyées en même temps)
bg
```

Exercice 9

La commande `fg` permet de mettre en foreground un processus en background. Testez la séquence suivante :

```
emacs
CTRL + Z
bg
fg
```

Exercice 10

Pour terminer un processus en foreground, il est possible d'utiliser la commande `CTRL + C`. Testez la séquence suivante :

```
emacs
CTRL + C
```

puis

```
emacs &
CTRL + C
fg
CTRL + C
```

Exercice 11 : En dernier recours

Surtout ne tapez pas `kill -9 -1`, mais essayez quand même en fin de séance, après avoir TOUT sauvegardé et fermé firefox.

Exercice 12 : Retour sur les scripts

Proposez un script `unscript.sh` dont le code est :

```
echo $1
echo $2
```

Puis exécutez le script ainsi : `unscript.sh aaa bbb`. Pensez à définir les droits en exécution.

Exercice 13

Proposez un script `terminer.sh` qui kill le programme dont le nom est passé en argument. Par exemple `terminer.sh firefox` doit tuer le processus firefox.