I IUT de Montpellier - Programmation Systeme
Les processus

6 septembre 2022

1 Manipulation des processus en Shell (rappels)

Connectez vous sous Linux et ouvrez un terminal.
ps permet de lister les processus en cours
— Lister les processus en cours dans votre terminal. Quel est le PID de votre bash ?
— A l'aide de la page man de ps, trouvez l'option qui permet de lister tous les processus de votre machine. Remarquez que la commande ps est elle mˆeme un processus.
— (*?*) `a l'aide d'un pipe et de la commande wc, comptez ce nombre total de processus.
— L'option -l (comme souvent avec les commandes shell) permet d'obtenir plus d'informations. Quel est le PPID de votre bash ?
— Quel est le processus parent du processus init (ou /sbin/launchd sur un Mac) ?
— (*?*) De quelle g´en´eration est votre processus bash (c.a.d. combien de parents lui et le processus init) ?
Vous pouvez vous aider d'un pipe et de la commande grep pour ne pas avoir `a lire tous les processus — lancer la commande sleep pour 10 secondes
— lancer la commande sleep pour 10 secondes en arri`ere plan. Quel est le p`ere du processus sleep. — Lorsque vous ex´ecutez la commande ps -Al vous devez voir apparaˆıtre une colonne UID qui est l'id de l'utilisateur qui a lanc´e le processus.
Remarquez qu'il y a de nombreux 'utilisateurs' mˆeme si vous ˆetes le seul connect´e `a la machine.
— (*? ? ?*) A l'aide des commandes ps, tr, cut , sort , cut et wc, comptez le nombre d'UID diff´erents.

2 Programmation C

Pour cette partie du TD reportez vous `a la section Le PID du support de cours fourni en Amphi.

2.1 Compiler un programme C (rappel)

— Cr´eez un fichier hello.c et ins´erer y le code suivant :

```
#i n cl u d e <s t d i o . h>
i n t main ( v oid ) {
p r i n t f ( " H ell o World ! " ) ;
}
```

— Compilez ce fichier pour obtenir un ex´ecutable nomm´e hello. Ex´ecutez le

2.2 Gestion des arguments

Compilez et testez ce petit programme simple de manipulation des arguments.

```
#i n cl u d e <s t d i o . h>
i n t main ( i n t argc , ch a r * argv [ ] ) {
i n t i =0;

p r i n t f ( "\n Le nombre d ' arguments e s t =%d" , a r g c ) ;

/* Le p remie r argument e s t l e nom du programme */
p r i n t f ( "\n Cet e x e c u t a bl e e s t =%s " , argv [ 0 ] ) ;

f o r ( i =1; i< a r g c ; i++) {
p r i n t f ( "\n Argument %d=%s " , i , argv [ i ] ) ;
```

```
}
printf("\n");
}
```

2.3 Affichage du pid

Ecrire un programme C afichePid qui

1. affiche son pid
2. s'endort pour 10 secondes grace `a l'appel sleep

```
#include <unistd.h>
unsigned int sleep(unsigned int seconds);
```

   Remarque Ici on donne la signature de la m´ethode. Dans votre code C, il faut juste ajouter l'instruction sleep(10)

3. fini proprement `a l'aide de l'appel exit(0).

```
#include <stdlib.h>
void exit(int status);
```

Rappel sur les codes de retour d'un programme

Il n'y a pas d'exception en C (comme en JAVA). Toute commande LINUX a un code de retour. Si celui si vaut 0, c'est que la commande s'est bien pass´e. Sinon elle renvoie un entier (diff´erente de 0). En shell, on r´ecup`ere le code de retour de la derni`ere commande ex´ecut´ee avec echo $?

bash-3.2$ ls public_html
index.html
bash-3.2$ echo $?
0
bash-3.2$ ls public_html/truc.php. // un fichier qui n'existe pas
ls: public_html/truc.php: No such file or directory
bash-3.2$ echo $?
1. //diff´erent de 0, donc il y a eu un probl`eme.

2.4 Affichage du PPID d'un processus

— Ecrire un programme C affichePPid qui prend un argument de de type entier, qui affiche le PPID du processus dont le PID est celui pass´e en param`etre du programme.
Pour convertir le convertir une chaine de caract`eres (ici argv[1] en un entier, vous utiliserez la fonction atoi, dont voici la signature.

```
#include <stdlib.h>
int atoi(const char *str);
```

— Gestion des erreurs. Faites en sorte que votre programme ai comme code de retour

   0 si le PID correspond `a un processus existant.

   1 si le programme est appel´e sans argument

   2 si l'argument pass´e n'est pas le PID d'un processus

— (?) Ecrire un programme C generationPid qui prend comme argument un PID, qui calcule le g´en´eration (profondeur dans l'arbre des processus) du processus dont le PID est pass´e en param`etre du programme et l'affiche. Vous pourrez utiliser au choix avec une fonction r´ecursive ou un boucle until. Impl´ementez les codes de retours du programme affichePPid de la question pr´ec´edente.