TP Tubes nommés FIFO

Systèmes d'Exploitation

2019-2020

Introduction

Un tube nommé (ou FIFO) offre un canal de communication unidirectionnel entre deux processus: un des processus va jouer le rôle de *l'écrivain* et l'autre celui du *lecteur*.

Nous pouvons noter les caractéristiques suivantes pour un tube nommé:

- Un tube nommé va exister comme un fichier spécial dans le système de fichier.
- Des processus indépendants peuvent communiquer en utilisant le tube (les processus n'ont pas besoin d'avoir une relation père-fils avec un `fork`)
- Quand les processus terminent leurs traitements, le tube nommé continuera d'exister dans le système de fichier

Création d'une FIFO

Nous pouvons créér une FIFO directement depuis le shell. Pour cela, nous avons deux commandes à notre disposition: `mknod` et `mkfifo`

A faire

```
man mknod
man mkfifo
```

Création avec mkfifo

Pour créer notre premier tube nommé, nous allons nous déplacer dans le répertoire temporaire et demander la création de la FIFO avec des droits associés

A faire

```
cd /tmp
mkfifo -m=rw MAFIFO
ls -al MAFIFO
```

Comment identifier une FIFO dans le file system ?

Observez le résultat de la commande `ls`; vous allez remarquer la présence d'un `p` qui indique qu'il s'agit d'un *pipe*.

Les opérations sur FIFO

Les opérations d'E/S sur une FIFO sont similaires à celles des fichiers standards. Par exemple, nous pouvons utiliser `fopen()` et `fclose()` pour ouvrir et fermer la FIFO et `fread()` et `fwrite()` pour lire et écrire.

A faire

Voici les codes sources de deux programmes client/serveur qui vont utiliser la FIFO créée précédemment.

```
//code du lecteur
#include <stdio.h>
#include <stdlib.h>
#include <sys/stat.h>
#include <unistd.h>
#include <linux/stat.h>

#define FIFO_PATH       "/tmp/MAFIFO"

int main(void)
{
        FILE *fp;
        char readbuf[80];
        while(1)
        {
                fp = fopen(FIFO_PATH, "r"); // je suis lecteur
                fgets(readbuf, 80, fp);
                printf("Recu: %s\n", readbuf);
                fclose(fp);
        }
```

```
        exit(0);
}
```

Voici maintenant le code de l'écrivain:

```
//code de ecrivain
#include <stdio.h>
#include <stdlib.h>

#define FIFO_PATH       "/tmp/MAFIFO"

int main(int argc, char *argv[])
{
        FILE *fp;
        if ( argc != 2 ) {
                printf("USAGE: %s [string]\n",argv[0]);
                exit(1);
        }
        if((fp = fopen(FIFO_PATH, "w")) == NULL) {
                perror("fopen");
                exit(1);
        }
        fputs(argv[1], fp);
        fclose(fp);
        return(0);
}
```

Vous pouvez:

1. Compiler les deux prgrammes
2. Lancer le programme serveur (le lecteur)
3. Lancer le programme client (l'écrivain) avec un message en paramètre
4. Le serveur devra recvoir votre message.

Application

Nous souhaitons réaliser une communication inter processus (par tubes nommés) dans laquelle deux processus A et B s'échangent une chaîne de caractères, plus précisément: - le processus client envoie la chaîne "hello, je suis le processus client"; - le processus serveur répond par la chaîne "hello, je suis le processus serveur".

Nous allons créer deux tubes nommés avec 'mkfifo':

```
mkfifo tube1
mkfifo tube2
```

Voici le code du client:

```
/************* PROCESSUS CLIENT ***************************/
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sys/types.h>
#include <sys/stat.h>
main()
{ int t1 ,t2;
char message[80];
t1 = open("tube1",O_WRONLY);
t2 = open("tube2",O_RDONLY);
strcpy(message, "hello, je suis le processus client");
write(t1,message,strlen(message)) ;
read(t2,message,80) ;
printf("le message recu est :%s\n", message);
exit(0);
}
```

Complétez le code du serveur:

```
/************* PROCESSUS SERVEUR ***************************/
main()
{
// partie a écrire
}
```