Système partie 1 - cours 3

Pastille 4 : fonctions de manipulation des fichiers I

N. de Rugy-Altherre - Vincent Colotte

Primitives de base d'opérations sur les fichiers

Les fonctions de manipulation des fichiers : POSIX

1. open
2. close
3. read
4. write
5. lseek
6. fcntl, fstat
7. dup, dup2

. . . d'un point de vue «système».

Open

int open (const char * *ref*, int *mode_ouv*, . . . /* les droits*/) ;

Open

ouvre un fichier.

Open

int open (const char * *ref*, int *mode_ouv*, . . . /* les droits*/) ;

Open

ouvre un fichier.

Open

int open (const char * *ref*, int *mode_ouv*, . . . /* les droits*/) ;

Open

Déclare au système le type des opérations réalisables sur le fichier en cas de réussite de l'ouverture :

- Charge dans la table des i-noeuds en mémoire (i-noeud désigné par *ref*).
- Allocation d'une nouvelle entrée dans la table des fichiers ouverts du système.
- allocation d'un descripteur dans la table lié au processus.

Il renvoie le **descripteur** (-1 sinon).

Les constantes d'ouvertures

int open (const char * *ref*, int *mode_ouv*, . . . /* les droits*/) ;

Les constantes du mode d'ouverture sont ajoutées par disjonction et sont de 3 types :

- Obligatoire : 0_RDONLY, O_WRONLY ou O_RDWR (inscrit dans la table de fichiers ouverts mais non modifiable)
- Optionnelle : utile à l'ouverture seulement (non inscrit dans la table des fichiers ouverts)
  (O_TRUNC le fichier est mis à zéro, O_CREAT création avec les droits)
- Optionnelle mémorisable : inscrit dans la table et modifiable par la suite
  (O_APPEND écriture à la fin, O_NONBLOCK open non bloquant (pour tube ou terminaux), O_SYNC bloque le processus jusqu'à l'écriture effective sur le disque)

Exercice

Exercice

- Donnez la commande pour ouvrir un fichier nommé `truc.c` en lecture seule avec une mise à zéro.
- Demandez la création du fichier `machin.truc` en écriture seule avec les droits rwxr-xr-x

Fermeture d'un fichier

close

```
#include <unistd.h>
int close (int descr) ;
```

- enlève les verrous mis par le processus sur le fichier,
- décrémentation de l'entrée dans la table des fichiers ouverts,
- si nul, l'entrée est libérée et le i-noeuds est décrémenté.
- ...

Exercice

Écrivez un programme C qui ouvre un fichier et le ferme

.

Read

```
#include <unistd.h>
ssize_t read(int descr, void *buffer, size_t nb_octets)
```

Read

Demande la lecture d'au plus *nb_octets* caractères dans le fichier ayant pour descripteur *descr*. Les caractères lus sont placés dans *buffer*. La fonction renvoie le nombre de caractères effectivement lus.

Read

```
#include <unistd.h>
ssize_t read(int descr, void *buffer, size_t nb_octets)
```

Read

Demande la lecture d'au plus *nb_octets* caractères dans le fichier ayant pour descripteur *descr*. Les caractères lus sont placés dans *buffer*. La fonction renvoie le nombre de caractères effectivement lus.

Remarques :

- lit à partir de la position courant
- l'**offset** est incrémenté du nombre de caractères effectivement lus.
- S'il y a un verrou exclusif impératif sur le fichier dans la portée de lecture : le processus est bloqué (sauf si O_NONBLOCK ou O_NDELAY).

Exercice

Exercice

- Créez un fichier `truc.txt` contenant le text `Bonjour.`.
- Écrivez un programme en C qui ouvre ce fichier et le ferme. Affichez son descripteur et le retour de `close` pour vérifier que tout va bien.
- Lisez avec la fonction `read` le premier caractère de votre fichier. Pensez à afficher le nombre de caractère effectivement lus.

Write

```
#include <unistd.h>
ssize_t write (int descr, void * buffer, size_t nb_octets) ;
```

Write

Demande d'écriture des nb_octets caractères, lus dans *buffer*, dans le fichier dont le descripteur est *descr*. La fonction renvoie le nombre de caractères effectivement écrits dans le cache du noyau (ou sur le disque si O_SYNC).

Write

```
#include <unistd.h>
ssize_t write (int descr, void * buffer, size_t nb_octets) ;
```

Write

Demande d'écriture des nb_octets caractères, lus dans *buffer*, dans le fichier dont le descripteur est *descr*. La fonction renvoie le nombre de caractères effectivement écrits dans le cache du noyau (ou sur le disque si O_SYNC).

Remarque :

- S'il y a un verrou exclusif impératif sur le fichier dans la portée d'écriture : le processus est bloqué (sauf si O_NONBLOCK ou O_NDELAY).

lseek

```
#include <unistd.h>
off_t lseek (int descr, off_t position, int origine);
```

lseek

- Modifier la position courante (ou offset) de l'entrée de la table des fichiers ouverts
- Position courante = *origine* + *position*
- *origine* : SEEK_SET, SEEK_CUR, SEEK_END
- Valeur renvoyée : nouvelle position courante (entier relatif). (off_t) -1 : erreur (dépassement de la fin du fichier ou avant 0)

À noter

- Certains fichiers n'autorisent pas son utilisation (tube, terminaux...)
- Cette fonction est utilisée pour accéder à des données non séquentiellement.

fcntl

```
#include <fcntl.h>
int fcntl (int descr, int commande, ...) ;
```

fcntl

permet de retrouver ou de modifier les attributs des différentes entrées liées à *descr*.

- La fonction fcntl permet de pratiquement tout faire sur les fichiers.
- Le verrouillage des fichiers est réalisé avec cette fonction.

Exemple : F_GETFL, F_SETFL : Récupération/modifications des attributs liés au mode d'ouverture.

fstat et stat

```
#include <sys/types.h>
#include <sys/stat.h>
#include <unistd.h>
int fstat (int descr, struct stat *pStat) ;
int stat (const char *ref, struct stat *pStat) ;
```

fstat

permet d'obtenir la structure *stat* à partir de *descr*.

- La structure *stat* : contient les informations propriétaire, droits, date d'accès, numéro d'i-node...
- La fonction *stat* : permet d'accéder au fichier par son chemin/nom.
- *fstat* et *stat* renvoient 0 si la récupération est réussie (-1 sinon)

(voir TP)