Unix : concepts et outils avancés
TD no 9 : tubes nommés, mémoire partagée

Olivier Aumage
LIP, bureau 302
email : `Olivier.Aumage@ens-lyon.fr`

29 novembre 2001

1 Tubes nommés

La commande `mkfifo(1)` permet de créer un *tube nommé* (ou FIFO), c'est-à-dire un tube persistant et accessible par l'intermédiaire de l'espace de noms des fichiers.

Questions

- Créez un tube nommé `fifo` à l'aide de la commande `mkfifo`. En utilisant deux terminaux, testez le fonctionnement du tube (utilisez deux instances de `cat`).
- Les tubes nommés s'utilisent généralement dans les applications de type client-serveur. Dans cet exercice, on veut pouvoir obtenir la date actuelle en effectuant une lecture sur le tube `fifo`. Ecrivez un programme serveur en **C** qui propose ce service. Utilisez `time(2)` et `ctime(3C)` pour récupérer la date. Testez votre serveur en utilisant `cat`.

2 Mémoire partagée et fichiers mappés

L'appel système `mmap(2)` permet d'associer (dans son intégralité ou partiellement) un fichier désigné par son descripteur à une zone de mémoire d'un processus. Il est alors possible d'accéder au contenu du fichier directement à l'aide d'opérations de lecture et/ou d'écriture en mémoire : un fichier ainsi *mappé* est vu comme un tableau d'octets. L'effet des opérations d'écriture sur le fichier initial est contrôlé par un drapeau dont la signification est la suivante :

- `MAP_SHARED` signifie que les opérations d'écriture dans la zone de mémoire correspondant au fichier mappé affectent directement le contenu du fichier initial.
- `MAP_PRIVATE` signifie que les opérations d'écriture dans la zone de mémoire correspondant au fichier mappé s'effectuent sur une copie privée du contenu du fichier. Le fichier lui-même n'est dans ce cas pas altéré.

L'appel système `munmap(2)` supprime le *mapping* du fichier en mémoire.

Questions

– Écrivez une version simplifiée de la commande `tac(1)` qui prend un fichier en argument et affiche les lignes du fichier dans l'ordre inverse. Le fichier passé en argument sera évidemment mappé en mémoire à l'aide de `mmap(2)`.
– Un processus père `p1` dispose d'un tableau de 8 caractères. De même, un processus fils `p2` dispose d'un tableau de 8 caractères. Le but est d'échanger les chaînes respectives de `p1` et `p2` au moyen d'un unique segment de mémoire partagée de taille 9 caractères, sous la forme d'un fichier mappé (avec la fonction `mmap`). L'un des 9 caractères du segment partagé sera utilisé pour ordonner les accès du père et les accès du fils.

— ✲ —