

Programmation Système






Travaux Pratiques (5), Licence 2 Informatique

La communication Inter-Processus sous BSD : Les sockets

Les exercices suivants ont pour but de vous familiariser à la manipulation des sockets, points de communication inter-processus sous BSD.

Il vous est recommandé de consulter les pages man pour de plus amples informations sur leur syntaxe, leur sémantique et les éventuelles options qu'elles offrent.






Les instructions des exercices se repèrent par des icônes, qui sont les suivantes :

- | | | |
|---|-------------|--|
|  | Information | Information concernant l'usage ou le rôle d'une commande, par exemple. Dans certains cas, il s'agit d'une information sur ce que vous êtes en train de faire ou sur ce qui se passe. |
|  | Exemple | Exemple d'utilisation. |
|  | Contrôle | Vérifier le résultat d'une (ou plusieurs) action(s). |
|  | Action | Effectuer la ou les action(s) décrite(s). |
|  | Question | Questions auxquelles vous devez répondre. |

De plus, un texte en police courier correspond soit à une sortie écran soit à des noms spécifiques (menus, fenêtre, icône, processus, commandes...).

Un **texte en police times gras** correspond à ce que l'utilisateur doit introduire comme valeur de paramètre, ou encore, est utilisé pour attirer l'attention de l'utilisateur.

La communication par socket dans le domaine Unix

- | | |
|---|--|
|  | La primitive <code>int socket(int domain, int type, int protocol)</code> ; permet de créer une socket : point de communication. Elle retourne un descripteur. A une socket est associé un domaine (<code>AF_UNIX</code> , <code>AF_INET</code> , ...) ainsi qu'un type (<code>SOCK_DGRAM</code> , <code>SOCK_STREAM</code> , ...). |
|  | Écrire deux programmes, un client et un serveur, qui créent chacun une socket dans le domaine <code>AF_UNIX</code> dans le mode non connecté (socket de type <code>SOCK_DGRAM</code>). |
|  | S'assurer que les sockets ont bien été créées (vérifier le code de retour). |
|  | La primitive <code>int bind(int sockfd, struct sockaddr *my_addr, int addrlen)</code> ; permet d'associer un nom à une socket. |
|  | Compléter le programme serveur de façon que sa socket ait le nom " <code>serv_sock</code> ". |



Vérifier que la socket précédente est bien visible dans le système de fichiers.



Les primitives

```
int sendto(int s, const void *msg, int len, unsigned int flags,  
           const struct sockaddr *to, int tolen); et  
  
int recvfrom(int s, void *buf, int len, unsigned int flags,  
             struct sockaddr *from, int *fromlen);
```

permettent l'échange de données entre processus utilisant des sockets de type SOCK_DGRAM.



Compléter les précédents programmes de façon qu'ils communiquent.



Modifier les programmes précédents de façon qu'ils communiquent en mode connecté (sockets de type SOCK_STREAM).

La communication par socket dans le domaine INTERNET



Écrire deux programmes, un client et un serveur, qui communiquent à l'aide des sockets dans le domaine AF_INET dans les deux modes: non connecté (sockets de type SOCK_DGRAM) et connecté (sockets de type SOCK_STREAM).



Tester vos programmes en exécutant chacun d'eux sur un site différent.



Exécuter votre programme client avec le programme serveur d'un autre groupe.

Implémentation d'un serveur multiple



Transformer le serveur de l'exercice précédent de manière qu'il puisse prendre en compte les connexions de plusieurs clients (chaque nouvelle communication est prise en charge par un processus fils). Application au mode connecté seulement.



Tester vos programmes en lançant plusieurs clients.

Multiplexage des entrées/sorties



La primitive `int select(int n, fd_set *readfds, fd_set *writefds, fd_set *exceptfds, struct timeval *timeout);` permet de multiplexer les entrées/sorties en indiquant lequel des descripteurs est prêt pour une lecture, une écriture ou présente une condition d'erreur pendante.



Transformer le serveur de l'exercice précédent de manière qu'il puisse se mettre en attente de réception de messages sur plusieurs sockets à la fois (un seul processus avec utilisation de la primitive `select`).