Université Paris 7
IUP 2
Réseaux
Année 2003-2004

Résumé de Cours pour le TP no 3
Multiplexage I/O et la Fonction *select*.

- **Le problème :**

  Quand on a besoin de lire un descripteur et d'écrire dans un autre, on utilise de l'entrée/sortie (I/O) bloquant dans une boucle comme ceci :

```
while ((n = read(STDIN_FILENO, buf, BUFSIZ)) > 0)
   if (write(STDOUT_FILENO, buf, n) != n)
       err_sys("write error");
```

  Que faire si on veut lire deux descripteurs ? On ne peut pas faire de *read* bloquant sur l'un des deux car des données peuvent arriver sur l'autre pendant que nous serons bloqués.

- **Les Solutions**

  - *fork* : créer deux processus et leur faire faire un *read* bloquant sur chacun des processus. Mais problème quand l'opération termine : on doit utiliser des signaux en cas de mort du fils ou du père et le programme gagne en complexité.
  - I/O asynchrone : utiliser des *read* et *write* non bloquants, dans des boucles et alternativement sur les deux descripteurs. Mais le problème est que cela utilise beaucoup le processeur.
  - Multiplexage I/O et la Fonction *select* : on crée une liste de descripteurs qui non intéressent et on appelle une fonction qui ne retourne pas avant que l'un des descripteurs ne soit prêt pour lire ou écrire.

- **Select**

```
#include <sys/time.h>
#include <sys/types.h>
#include <unistd.h>

int select(int n, fd_set *readfds, fd_set *writefds, fd_set *exceptfds,
       struct timeval *timeout);
```

- **Valeur de Retour :**

  -1 pour une erreur

  0 pour un timeout

  > 0 Nombre de descripteurs prêts. Dans ce cas, seuls les descripteurs prêts sont encore présents dans les ensembles readfds, writefds et exceptfds.

- **Que veut dire qu'un descripteur est prêt ?**

  1 Un descripteur de l'ensemble lire (readfds) est considéré comme prêt si un *read* sur ce descripteur ne va pas bloquer.

  2 Un descripteur de l'ensemble écrire (writefds) est considéré comme prêt si un *write* sur ce descripteur ne va pas bloquer.

- **Le premier argument de *select***, n, est le plus grand descripteur de fichier qui nous intéresse auquel on a ajouté 1.

- **Manipuler la structure** `fd_set` :

  Il faut utiliser les macros :

```
FD_ZERO(fd_set *set);           Efface tous les descripteurs de fdset.
FD_SET(int fd, fd_set *set);    Rentre le descripteur fd dans fdset.
FD_CLR(int fd, fd_set *set);    Efface le descripteur fd de fdset.
FD_ISSET(int fd, fd_set *set);  Teste si le descripteur fd est dans fdse
```

- **L'exemple du *man select***

```
#include <stdio.h>
#include <sys/time.h>
#include <sys/types.h>
#include <unistd.h>

int main(void) {
    fd_set rfds;
    struct timeval tv;
    int retval;

    /* Watch stdin (fd 0) to see when it has input. */
    FD_ZERO(&rfds);
    FD_SET(0, &rfds);
    /* Wait up to five seconds. */
    tv.tv_sec = 5;
    tv.tv_usec = 0;

    retval = select(1, &rfds, NULL, NULL, &tv);

    if (retval)
        printf("Data is available now.\n");
        /* FD_ISSET(0, &rfds) will be true. */
    else
        printf("No data within five seconds.\n");

    return 0;
}
```