Synchronisation par sémaphores
[IPC System V]

Présentation

Linux propose un ensemble de primitives de gestion de sémaphores, il s'agit de :

- `semget`, (création)
- `semctl`, (initialisation et contrôle)
- `semop`, (opérations sur le compteur du sémaphore, en particulier `P` et `V`)

Nous allons présenter rapidement ces outils et les structures de données associées puis en montrer l'utilisation sur un exemple.

Fonctionnement

Les sémaphores sont implantés par System V sous forme de tableaux de sémaphores. Un tel tableau est repéré par son identificateur, ce dernier est retourné par la fonction `semget`.

L'initialisation des compteurs associés à chacun des sémaphores de l'ensemble est faite par `semctl` qui permet aussi d'obtenir des informations sur l'état des sémaphores (processus en attente, valeur courante, ...).

La modification des compteurs (opérations appelées `P` et `V` dans les ouvrages sur les systèmes d'exploitation) associés aux sémaphores, appelés ici valeur des sémaphores, se fait par `semop` qui définit un ensemble d'opérations à effectuer sur un ou plusieurs sémaphores d'un ensemble qui aura été préalablement sélectionné par `semget`. Les structures de données nécessaires à la manipulation des sémaphores sont répertoriées dans :
`/usr/include/sys/sem.h`
`/usr/include/sys/ipc.h`
`/usr/include/sys/types.h`

Nous allons maintenant décrire ces trois fonctions : `semget`, `semop` et `semctl`.

1.1 semget

```
int semget(int sem_clef, int nsem, int drapeau)
```

semget renvoie l'identificateur interne de l'ensemble de sémaphores (appelé sem_id par la suite) désigné par la clef sem_clef. Si le bit IPC_CREAT est positionné dans la variable drapeau ou si «sem_clef = IPC_PRIVATE », alors semget crée un ensemble de nsem sémaphores, et initialise les champs des objets de type semid_ds. Sinon, il positionne simplement l'utilisateur sur cet ensemble de clef sem_clef.

1.2 semop

```
int semop(int sem_id, struct sembuf *sem_oper, int nb_op)
```

semop réalise une suite de nb_op opérations sur le tableau de sémaphores dont l'identifiant interne est sem_id (entier renvoyé par semget). sem_oper est un tableau, ou simplement un pointeur (si nb_op=1), sur nb_op objets de type sembuf (voir polycopié du cours) constitués des trois champs suivants :

sem_num : numéro du sémaphore de l'ensemble sem_id
sem_op : type de l'opération
sem_flg : paramètre de cette opération

Le point le plus délicat est l'utilisation de sem_op, que l'on peut résumer ainsi:

Valeur de sem_op	Action
N	Valeur du sémaphore + N
0	Attendre valeur du sémaphore = 0
-N	Attendre valeur du sémaphore >= N et lui retirer alors N

Les processus mis en attente sont débloqués si le tableau de sémaphores est détruit.

Remarque :

On constate que le champ sem_op peut prendre une valeur différente de 1, ce qui permet de faire des opérations du type P ou V qui ajoutent, ou retirent, plus d'une unité.

1.3 semctl

```
int semctl(int sem_id, int sem_num, int commande, ...)
```

semctl applique la commande «commande» sur le sémaphore sem_num de l'ensemble sem_id. Suivant la commande envoyée, cette fonction renvoie une valeur, ou range des informations dans arg. La définition de « arg» est variable suivant les commandes, on peut la représenter ainsi :

```
union sem_union{
        int valeur,
        struct semid_ds *sem_etat,
        short *tableau,
} arg;
```

Exemples de valeurs pour le paramètre commande (voir polycopié du cours) :

GETPID /* Pid du processus en attente */
GETVAL /* valeur d'un sémaphore */
GETALL /* valeur de tous les sémaphores */

Important :

Cette fonction permet l'initialisation des valeurs des compteurs des sémaphores (fonction `init` dans la littérature des systèmes d'exploitation) si on initialise la commande avec les valeurs SETVAL ou SETALL :

SETVAL /* initialiser un sémaphore */
SETALL /* initialiser tous les sémaphores */

EXERCICE :

Un groupe d'ouvrier (N personnes) doit prendre le matin un ascenseur (charge maximale 2 personnes) pour monter sur le lieu de travail.

Réaliser une simulation où le processus père gère l'ascenseur et les ouvriers sont simulés par des processus fils. L'ascenseur est réalisé par un sémaphore.

Les fils notent leur pid dans un segment de mémoire partagée quand ils montent dans l'ascenseur. L'ascenseur arrivé en haut, le processus père invite les fils à sortir de l'ascenseur (le père lit les pid dans le segment de mémoire et envoie le signal SIGUSR1 aux fils).