

Chapitre 15

LES SEGMENTS DE MEMOIRE PARTAGEE

Un segment de mémoire partagée est une zone d'octets désignée par un pointeur (adresse de base). La communication entre processus par fichiers, tubes ou files de messages suppose d'abord la copie des données de l'espace d'adressage de l'émetteur vers le noyau, puis la copie des données du noyau vers l'espace d'adressage du processus destinataire. Pour ces opérations, le processus bascule nécessairement du mode utilisateur en mode noyau.

Au contraire, un segment de mémoire partagée est utilisable par plusieurs processus sans avoir à le recopier dans leurs environnements respectifs.

Un segment de mémoire partagée est identifié par le système grâce à une clé entière et accessible par d'autres processus grâce à cette clé. L'adresse de base de la mémoire partagée, vue de deux processus, n'a aucune obligation d'être identique : les processus ne sont pas obligés d'utiliser la zone d'octets à partir de son "début".

A tout segment de mémoire partagée, est associée une structure d'informations de *type struct shmids*, définie dans `/usr/include/sys/shm.h`, et consultable par la fonction système `ipcs`. Ce fichier définit également un certain nombre de constantes.

Une action `exit` sur un processus ayant créé des segments libère ceux-ci.

1. CREATION D'UN SEGMENT DE MEMOIRE PARTAGEE

La fonction prototypée par :

int shmget (key_t cle, int taille, int option)

retourne l'identificateur d'un segment de mémoire partagée associée à une clé donnée (-1 en cas d'erreur), avec :

taille : la taille du segment en octets (comprise entre un minimum et un maximum définis lors de la configuration du système)

option : combinaison, avec l'opérateur `|`, de droits d'accès et de constantes prédéfinies:

SHM_R : permission en lecture (ou 0400)

SHM_W : permission en écriture (ou 0200)

IPC_CREAT, IPC_EXCL

cle : clé de type `key_t` (défini dans `/usr/include/sys/ipc.h`, équivalent à long)

si **cle** = `IPC_PRIVATE`, un nouveau segment est créé

si non si **cle** ne correspond pas à un segment existant,

si `IPC_CREAT` n'est pas positionné : erreur et retour de -1

sinon, un nouveau segment est créé et associé à cette clé avec les droits d'accès mentionnés. Le propriétaire et le créateur sont le

propriétaire effectif du processus. Le groupe propriétaire et le groupe créateur sont le groupe propriétaire effectif du processus

sinon si `IPC_CREAT` **et** `IPC_EXCL` sont positionnés : erreur, retour de -1
sinon, la fonction retourne l'identificateur du segment

Exemple : récupération de l'identificateur d'un segment
`num = shmget (ftok (CHEMIN, cle) , taille, 0)`
`/* ou bien : num = shmget (ftok(CHEMIN, cle) , 0 , 0) */`

Exemple de création d'un segment :

```
#include <stdio.h>
#include <sys/types.h>
#include <sys/shm.h>

main ()
{
    int num;
    if ((num = shmget (IPC_PRIVATE, 1024, IPC_CREAT | IPC_EXCL | 0666)) == -1)
        fprintf (stderr, "erreur\n");
    else printf ("segment d'identificateur %d créé\n", num);
}
```

2. OPERATIONS DE CONTROLE

La fonction prototypée par :

int shmctl (int id, int cmd, struct shmid_ds *buf)

retourne 0 (- 1 en cas d'erreur), avec :

id : identificateur du segment (le retour de shmget)

buf : pointeur sur la structure shmid_ds, contenant une copie des informations sur le segment

cmd : `IPC_STAT` : la structure shmid_ds associée au segment est copiée dans la structure pointée par buf

`IPC_SET` : la structure shmid_ds associée au segment est modifiée par les valeurs des champs de la structure pointée par buf

`IPC_RMID` : suppression du segment de la mémoire. La suppression sera différée si le segment est encore attaché à un processus.

3. ATTACHEMENT D'UN SEGMENT A UN PROCESSUS

La fonction prototypée par :

void * shmat (int id, const void *adr, int flag)

retourne adr ou NULL en cas d'erreur, avec :

id : identificateur du segment (le retour de shmget)

adr : si adr = NULL, le système choisit la première adresse disponible et la fonction la retourne.

Alors, adr pointe sur le 1er octet du segment.

Avantage : l'adresse sera correctement construite

si adr != NULL, et si SHM_RND est positionné dans flag, le segment est attaché à l'adresse adr - (adr modulo SHMLBA, *segment low boundary address*, adresse de base de la mémoire partagée)

si adr != NULL, et si SHM_RND n'est pas positionné dans flag, le segment est attaché à l'adresse adr

si SHM_RDONLY (010000) est positionné dans flag, le segment n'est attaché qu'en lecture. Toute tentative d'écriture génère le signal SIGSEGV

Attention, les segments commencent à des limites de pages.

4. DETACHEMENT D'UN SEGMENT

La fonction prototypée par :

int shmdt (const void *ptr)

retourne 0 (- 1 en cas d'échec)

ptr : adresse retournée par shmat.

Cette fonction détache le segment du processus qui l'utilise.

Attention, ptr garde sa valeur et on évitera de l'utiliser après un appel à shmdt. La fonction shmdt **ne détruit pas** le segment. Seul un appel à shmctl peut le faire.

La terminaison d'un processus entraîne le détachement de tous les segments qu'il a préalablement créés.