Semafori in Linux

- `semget()`

  per ottenere un vettore di semafori

- `semop()`

  per eseguire le normali operazioni sui semafori corrispondenti alle primitive di sincronizzazione

- `semctl()`

  per varie operazioni di controllo, inclusa la rimozione di un vettore di semafori

semget()

```
#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/sem.h>
int semget(key_t key, int nsems, int semflg)
```

man 2 semget

- restituisce un identificatore intero univocamente associato a `key`
- viene creato un vettore di `nsems` semafori se
  - `key` è `IPC_PRIVATE`
  - `key` non è associato a nessun vettore di semafori preesistenti e `semflg` è `IPC_CREAT`
- `semflg` contiene 9 bit di permessi ed eventualmente i flag
  - `IPC_CREAT`
  - `IPC_EXCL` (creazione in esclusiva)
- restituisce `-1` in caso di errore

semop()

```
#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/sem.h>
int semop(int semid,struct sembuf *sops,unsigned nsops)
```

man 2 semop

- esegue `nsops` operazioni sul vettore di semafori `semid`
- ciascuna operazione è specificata da una struttura `sembuf` contenente almeno:

```
short sem_num;  /* semaphore number: 0 = first */
short sem_op;   /* semaphore operation */
short sem_flg;  /* operation flags */
```

- `sem_flg` può assumere i flag
  - `IPC_NOWAIT` (causa fallimenti della chiamata anziché attese)
  - `IPC_UNDO` (se un processo esegue `exit()` viene eseguito l'undo di tutte le modifiche che ha apportato al vettore di semafori con questo flag attivo)
- restituisce `0` in caso di successo e `-1` in caso di errore

semop(): operazioni

```
int semop(int semid,struct sembuf *sops,unsigned nsops)
```

Il campo sem_op

- > 0 sem_op viene sommato al valore del semaforo. In questo modo intende evidenziare che una risorsa protetta da questo semafore è stata da lui rilasciata
- < 0 Se il valore assoluto di sem_op è maggiore del valore del semaforo, il processo viene bloccato sino a quando il semaforo raggiunge il valore assoluto di sem_op. Quindi questo valore assoluto viene sottratto al valore del semaforo. In questo modo intende evidenziare che una risorsa protetta da questo semaforo è stata da lui allocata
- Zero Il processo viene sospeso sino a quando il semaforo torna a 0

semctl()

```
#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/sem.h>
int semctl(int semid, int semnum, int cmd, union semun arg)
```

man 2 semctl

- esegue operazioni di controllo sul vettore di semafori `semid`
- ciascuna operazione è specificata da `cmd` ed utilizza una struttura unione (su alcune installazioni va dichiarata esplicitamente):

```
union semun {
  int val;                          /*value for SETVAL */
  struct semid_ds *buf;  /*buffer for IPC_STAT,IPC_SET*/
  unsigned short int *array; /*array for GETALL,SETALL*/
  struct seminfo *__buf;        /*buffer for IPC_INFO */
};
```

- restituisce `-1` in caso di errore ed in caso di successo un valore non negativo dipendente da `cmd`

semctl(): comandi

```
int semctl(int semid, int semnum, int cmd, union semun arg)
```

- valori permessi per `cmd`:
  - `IPC_RMID` rimuove il vettore di semafori
  - `GETALL` restituisce il valore dei semafori dentro `arg.array`
  - `GETVAL` restituisce il valore del semaforo `semnum`
  - `GETZCNT` restituisce il numero di processi in attesa del semaforo `semnum`
  - `SETALL` fissa il valore di tutti i semafori del vettore usando `arg.array`
  - `SETVAL` fissa il valore del semaforo `semnum` to `arg.val`
  - `GETNCNT`…
  - `GETPID`…

man 2 semctl