Pamięć współdzielona

Systemy Operacyjne 2

Piotr Zierhoffer

17 listopada 2011

Mechanizmy IPC

IPC — Inter Process Communication

- kolejki komunikatów,
- pamięć współdzielona
- semafory
- polecenia bash: `ipcs, ipcrm`

Pamięć współdzielona

- segment wirtualnej przestrzeni adresowej
- identyfikowany kluczem
- o określonej długości i prawach dostępu
- współdzielony przez wiele procesów
- zawartość usuwana przy usunięciu segmentu
- często wymaga synchronizacji semaforami

Stworzenie segmentu

- Funkcja:
  `int shmget(key_t key, size_t size, int shmflg)`
- Parametry:
  - `key` — klucz identyfikujący segment (“nazwa”)
  - `size` — rozmiar segmentu w bajtach
  - `shmflg` — flagi
- Wartość zwracana:
  - -1 w wypadku błędu
  - identyfikator segmentu powiązanego z kluczem `key`

Stworzenie segmentu, cd.

Opis parametrów:

- `key` — klucz identyfikujący segment ("nazwa")
  - `IPC_PRIVATE` — stwórz nowy segment o dowolnym identyfikatorze, ignoruj `shmflg` oprócz praw dostępu
  - zwyczajowo wartość heksadecymalna
- size — rozmiar segmentu pamięci, zaokrąglany w górę do wielokrotności rozmiaru strony (`getconf PAGE_SIZE`)
- `shmflg` — flagi, sumowane bitowo
  - prawa dostępu — nieużywane
  - `IPC_CREAT` — tworzenie kolejki
  - `IPC_EXCL` — "jeżeli kolejka istnieje i użyto `IPC_CREAT`, zwróć błąd"

Stworzenie segmentu, cd.

Następujące ograniczenia odnoszące się do zasobów pamięci wspólnej dotyczą funkcji `shmget`:

SHMALL Maksymalna liczba stron pamięci użytych do stworzenia segmentów pamięci wspólnej: zależna od systemu.

SHMMAX Maksymalny rozmiar pojedynczego segmentu pamięci wspólnej: zależny od implementacji (aktualnie 4MB).

SHMMIN Minimalny rozmiar pojedynczego segmentu pamięci wspólnej: zależny od implementacji (aktualnie 1 bajt, ale efektywny minimalny rozmiar wynosi `PAGE_SIZE`).

SHMMNI Maksymalna liczba segmentów pamięci wspólnej w systemie: zależna od implementacji (aktualnie 4096, ale w wersjach Linuksa wcześniejszych niż 2.3.99 wynosiła 128)

Aktualne wartości: `tail /proc/sys/kernel/shm*`

Przyłączenie segmentu

- Funkcja:
  `void* shmat(int shmid, char *shmaddr, int shmflg)`
- Parametry:
  - `shmid` — identyfikator segmentu zwrócony przez `shmget`
  - `shmaddr` — adres, pod którym segment ma być widoczny w procesie
  - `shmflg` — flagi
- Wartość zwracana:
  - -1 w wypadku błędu
  - adres, pod którym widoczny jest przyłączony segment pamięci

Przyłączenie segmentu, cd.

- `shmflg` — flagi specyfikujące zachowanie funkcji
  - `SHM_RDONLY` — przyłącz pamięć tylko do odczytu
  - `SHM_REMAP` — podmień dowiązany pod danym adresem segment nowym segmentem
  - `SHM_RND` — dokonaj zaokrąglenia `shmaddr` w dół do wielokrotności `SHMLBA` (równe `PAGE_SIZE`)
- `shmaddr` — adres, pod którym segment ma być widoczny
  - `NULL` — zalecana wartość, system sam określa adres (różne adresy w różnych procesach!)
  - dowolny adres wyrównany do granicy strony pamięci (chyba, że podano `SHM_RND`)

Odłączenie segmentu

- Funkcja:
  `int shmdt(const void* shmaddr)`
- Parametry:
  - `shmaddr` — adres początku segmentu pamięci, zwrócony przez `shmat`
- Uwagi:
  - odłączenie segmentu we wszystkich procesach nie powoduje jego usunięcia
  - segment nie zostanie usunięty aż wszystkie procesy go odłączą oraz zostanie oznaczony do usunięcia
- Wartość zwracana:
  - 0 lub -1 w wypadku błędu

Zarządzanie segmentami

- Funkcja:
  `int shmctl(int shmid, int cmd, struct shmid_ds *buf)`
- Parametry:
  - `msqid` — identyfikator segmentu pamięci zwrócony przez `shmget`
  - `cmd` — operacja do wykonania
    - IPC_RMID — zaznaczenie segmentu do usunięcia
    - usunięcie po odłączeniu wszystkich procesów
    - IPC_SET — zmiana parametrów segmentu: właściciela, grupy i praw na podstawie `buf`
    - IPC_STAT — pobranie parametrów segmentu do `buf`
  - `buf` — struktura na parametry operacji
- Wartość zwracana:
  - 0 lub -1 w wypadku błędu

Wywołania systemowe

Wpływ wywołań systemowych na segmenty pamięci współdzielonej:

fork() w wyniku wywołania fork() proces potomny dziedziczy dołączone segmenty pamięci wspólnej.

exec() po wykonaniu exec() wszystkie odwzorowane segmenty są odłączane (nie są usuwane).

exit() po wykonaniu exit() wszystkie dołączone segmenty pamięci wspólnej są odłączane (nie są usuwane).

Uwaga!

Jeden segment może być podłączony wiele razy w ramach jednego procesu!

Wykorzystanie pamięci współdzielonej

- Do pamięci współdzielonej odwołujemy się przez wskaźniki.
- Zapisać można dowolne dane z wyjątkiem wskaźników — są one zazwyczaj sensowne tylko w obrębie jednego procesu!
- Pamięci współdzielonej nie trzeba dodatkowo alokować.

```
int *base_addr;

int shmid = shmget(0xABC, MAX_SIZE * sizeof(int), IPC_CREAT);
if (shmid == -1) return -1;

addr = (int *) shmat(shmid, NULL, 0);

for(i = 0; i < MAX_SIZE, ++i)
     base_addr[i] = (int)i*i;


shmdt(addr);
```