NAME

svipc − System V interprocess communication mechanisms

SYNOPSIS

#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/msg.h>
#include <sys/sem.h>
#include <sys/shm.h>

DESCRIPTION

This manual page refers to the Linux implementation of the System V interprocess communication (IPC) mechanisms: message queues, semaphore sets, and shared memory segments. In the following, the word *resource* means an instantiation of one among such mechanisms.

Resource Access Permissions

For each resource, the system uses a common structure of type *struct ipc_perm* to store information needed in determining permissions to perform an IPC operation. The *ipc_perm* structure, defined by the *<sys/ipc.h>* system header file, includes the following members:

```
struct ipc_perm {
    uid_t          cuid;   /* creator user ID */
    gid_t          cgid;   /* creator group ID */
    uid_t          uid;    /* owner user ID */
    gid_t          gid;    /* owner group ID */
    unsigned short mode;   /* r/w permissions */
};
```

The *mode* member of the *ipc_perm* structure defines, with its lower 9 bits, the access permissions to the resource for a process executing an IPC system call. The permissions are interpreted as follows:

0400	Read by user.
0200	Write by user.
0040	Read by group.
0020	Write by group.
0004	Read by others.
0002	Write by others.

Bits 0100, 0010, and 0001 (the execute bits) are unused by the system. Furthermore, "write" effectively means "alter" for a semaphore set.

The same system header file also defines the following symbolic constants:

IPC_CREAT	Create entry if key doesn't exist.
IPC_EXCL	Fail if key exists.
IPC_NOWAIT	Error if request must wait.
IPC_PRIVATE	Private key.
IPC_RMID	Remove resource.
IPC_SET	Set resource options.
IPC_STAT	Get resource options.

Note that **IPC_PRIVATE** is a *key_t* type, while all the other symbolic constants are flag fields and can be OR'ed into an *int* type variable.

Message Queues

A message queue is uniquely identified by a positive integer (its *msqid*) and has an associated data structure of type *struct msqid_ds*, defined in <*sys/msg.h*>, containing the following members:

```
struct msqid_ds {
    struct ipc_perm msg_perm;
    msgqnum_t       msg_qnum;    /* no of messages on queue */
    msglen_t        msg_qbytes;  /* bytes max on a queue */
    pid_t           msg_lspid;   /* PID of last msgsnd(2) call */
    pid_t           msg_lrpid;   /* PID of last msgrcv(2) call */
    time_t          msg_stime;   /* last msgsnd(2) time */
    time_t          msg_rtime;   /* last msgrcv(2) time */
    time_t          msg_ctime;   /* last change time */
};
```

msg_perm	*ipc_perm* structure that specifies the access permissions on the message queue.
msg_qnum	Number of messages currently on the message queue.
msg_qbytes	Maximum number of bytes of message text allowed on the message queue.
msg_lspid	ID of the process that performed the last **msgsnd**(2) system call.
msg_lrpid	ID of the process that performed the last **msgrcv**(2) system call.
msg_stime	Time of the last **msgsnd**(2) system call.
msg_rtime	Time of the last **msgrcv**(2) system call.
msg_ctime	Time of the last system call that changed a member of the *msqid_ds* structure.

Semaphore Sets

A semaphore set is uniquely identified by a positive integer (its *semid*) and has an associated data structure of type *struct semid_ds*, defined in <*sys/sem.h*>, containing the following members:

```
struct semid_ds {
    struct ipc_perm sem_perm;
    time_t          sem_otime;   /* last operation time */
    time_t          sem_ctime;   /* last change time */
    unsigned long   sem_nsems;   /* count of sems in set */
};
```

sem_perm	*ipc_perm* structure that specifies the access permissions on the semaphore set.
sem_otime	Time of last **semop**(2) system call.
sem_ctime	Time of last **semctl**(2) system call that changed a member of the above structure or of one semaphore belonging to the set.
sem_nsems	Number of semaphores in the set. Each semaphore of the set is referenced by a non-negative integer ranging from **0** to *sem_nsems−1*.

A semaphore is a data structure of type *struct sem* containing the following members:

```
struct sem {
    int semval;  /* semaphore value */
    int sempid;  /* PID for last operation */
};
```

semval	Semaphore value: a non-negative integer.
sempid	ID of the last process that performed a semaphore operation on this semaphore.

Shared Memory Segments

A shared memory segment is uniquely identified by a positive integer (its *shmid*) and has an associated data structure of type *struct shmid_ds*, defined in *<sys/shm.h>*, containing the following members:

```
struct shmid_ds {
    struct ipc_perm shm_perm;
    size_t          shm_segsz;   /* size of segment */
    pid_t           shm_cpid;    /* PID of creator */
    pid_t           shm_lpid;    /* PID, last operation */
    shmatt_t        shm_nattch;  /* no. of current attaches */
    time_t          shm_atime;   /* time of last attach */
    time_t          shm_dtime;   /* time of last detach */
    time_t          shm_ctime;   /* time of last change */
};
```

shm_perm	*ipc_perm* structure that specifies the access permissions on the shared memory segment.
shm_segsz	Size in bytes of the shared memory segment.
shm_cpid	ID of the process that created the shared memory segment.
shm_lpid	ID of the last process that executed a **shmat**(2) or **shmdt**(2) system call.
shm_nattch	Number of current alive attaches for this shared memory segment.
shm_atime	Time of the last **shmat**(2) system call.
shm_dtime	Time of the last **shmdt**(2) system call.
shm_ctime	Time of the last **shmctl**(2) system call that changed *shmid_ds*.

SEE ALSO

ipc(2), **msgctl**(2), **msgget**(2), **msgrcv**(2), **msgsnd**(2), **semctl**(2), **semget**(2), **semop**(2), **shmat**(2), **shmctl**(2), **shmdt**(2), **shmget**(2), **ftok**(3)

COLOPHON

This page is part of release 3.22 of the Linux *man-pages* project. A description of the project, and information about reporting bugs, can be found at http://www.kernel.org/doc/man-pages/.