Full credit is given to the above companies including the OS that this PDF file was generated!

Red Hat Enterprise Linux Release 9.2 Manual Pages on 'semctl.3p' command

$ man semctl.3p

SEMCTL(3P) POSIX Programmer's Manual SEMCTL(3P)

PROLOG

This manual page is part of the POSIX Programmer's Manual. The Linux implementation of this interface may differ (consult the corresponding Linux manual page for details of Linux behavior), or the interface may not be implemented on Linux.

NAME

semctl ? XSI semaphore control operations

SYNOPSIS

```
#include <sys/sem.h>
int semctl(int semid, int semnum, int cmd, ...);
```

DESCRIPTION

The semctl() function operates on XSI semaphores (see the Base Defini? tions volume of POSIX.1?2017, Section 4.17, Semaphore). It is unspeci? fied whether this function interoperates with the realtime interprocess communication facilities defined in Section 2.8, Realtime.

The semctl() function provides a variety of semaphore control opera? tions as specified by cmd. The fourth argument is optional and depends upon the operation requested. If required, it is of type union semun, which the application shall explicitly declare:

```
union semun {
    int val;
    struct semid_ds *buf;
```

```
    unsigned short  *array;
  } arg;
```

Each operation shall be performed atomically.

The following semaphore control operations as specified by cmd are exe? cuted with respect to the semaphore specified by semid and semnum. The level of permission required for each operation is shown with each com? mand; see Section 2.7, XSI Interprocess Communication. The symbolic names for the values of cmd are defined in the <sys/sem.h> header:

GETVAL Return the value of semval; see <sys/sem.h>. Requires read permission.

SETVAL Set the value of semval to arg.val, where arg is the value of the fourth argument to semctl(). When this command is successfully executed, the semadj value corresponding to the specified semaphore in all processes is cleared. Also, the sem_ctime timestamp shall be set to the current time, as described in Section 2.7.1, IPC General Description. Requires alter permission; see Section 2.7, XSI Interpro? cess Communication.

GETPID Return the value of sempid. Requires read permission.

GETNCNT Return the value of semncnt. Requires read permission.

GETZCNT Return the value of semzcnt. Requires read permission.

The following values of cmd operate on each semval in the set of sema? phores:

GETALL Return the value of semval for each semaphore in the sema? phore set and place into the array pointed to by arg.array, where arg is the fourth argument to semctl(). Requires read permission.

SETALL Set the value of semval for each semaphore in the semaphore set according to the array pointed to by arg.array, where arg is the fourth argument to semctl(). When this command is successfully executed, the semadj values corresponding to each specified semaphore in all processes are cleared. Also, the sem_ctime timestamp shall be set to the current

time, as described in Section 2.7.1, IPC General Descrip?
tion. Requires alter permission.

The following values of cmd are also available:

IPC_STAT Place the current value of each member of the semid_ds data structure associated with semid into the structure pointed to by arg.buf, where arg is the fourth argument to sem?ctl(). The contents of this structure are defined in <sys/sem.h>. Requires read permission.

IPC_SET Set the value of the following members of the semid_ds data structure associated with semid to the corresponding value found in the structure pointed to by arg.buf, where arg is the fourth argument to semctl():

sem_perm.uid

sem_perm.gid

sem_perm.mode

The mode bits specified in Section 2.7.1, IPC General De?scription are copied into the corresponding bits of the sem_perm.mode associated with semid. The stored values of any other bits are unspecified. The sem_ctime timestamp shall be set to the current time, as described in Section 2.7.1, IPC General Description.

This command can only be executed by a process that has an effective user ID equal to either that of a process with appropriate privileges or to the value of sem_perm.cuid or sem_perm.uid in the semid_ds data structure associated with semid.

IPC_RMID Remove the semaphore identifier specified by semid from the system and destroy the set of semaphores and semid_ds data structure associated with it. This command can only be exe?cuted by a process that has an effective user ID equal to either that of a process with appropriate privileges or to the value of sem_perm.cuid or sem_perm.uid in the semid_ds data structure associated with semid.

RETURN VALUE

If successful, the value returned by semctl() depends on cmd as fol?

lows:

GETVAL The value of semval.

GETPID The value of sempid.

GETNCNT The value of semncnt.

GETZCNT The value of semzcnt.

All others 0.

Otherwise, semctl() shall return -1 and set errno to indicate the er?

ror.

ERRORS

The semctl() function shall fail if:

EACCES Operation permission is denied to the calling process; see Sec?

tion 2.7, XSI Interprocess Communication.

EINVAL The value of semid is not a valid semaphore identifier, or the

value of semnum is less than 0 or greater than or equal to

sem_nsems, or the value of cmd is not a valid command.

EPERM The argument cmd is equal to IPC_RMID or IPC_SET and the effec?

tive user ID of the calling process is not equal to that of a

process with appropriate privileges and it is not equal to the

value of sem_perm.cuid or sem_perm.uid in the data structure as?

sociated with semid.

ERANGE The argument cmd is equal to SETVAL or SETALL and the value to

which semval is to be set is greater than the system-imposed

maximum.

The following sections are informative.

EXAMPLES

Refer to semop().

APPLICATION USAGE

The fourth parameter in the SYNOPSIS section is now specified as "..."

in order to avoid a clash with the ISO C standard when referring to the

union semun (as defined in Issue 3) and for backwards-compatibility.

The POSIX Realtime Extension defines alternative interfaces for inter?

process communication. Application developers who need to use IPC should design their applications so that modules using the IPC routines described in Section 2.7, XSI Interprocess Communication can be easily modified to use the alternative interfaces.

RATIONALE

None.

FUTURE DIRECTIONS

None.

SEE ALSO

Section 2.7, XSI Interprocess Communication, Section 2.8, Realtime, semget(), semop(), sem_close(), sem_destroy(), sem_getvalue(), sem_init(), sem_open(), sem_post(), sem_trywait(), sem_unlink()

The Base Definitions volume of POSIX.1?2017, Section 4.17, Semaphore, <sys_sem.h>