NAME

shmget − allocates a shared memory segment

SYNOPSIS

#include <sys/ipc.h>
#include <sys/shm.h>

int shmget(key_t *key***, size_t** *size***, int** *shmflg***);**

DESCRIPTION

shmget() returns the identifier of the shared memory segment associated with the value of the argument *key*. A new shared memory segment, with size equal to the value of *size* rounded up to a multiple of **PAGE_SIZE**, is created if *key* has the value **IPC_PRIVATE** or *key* isn't **IPC_PRIVATE**, no shared memory segment corresponding to *key* exists, and **IPC_CREAT** is specified in *shmflg*.

If *shmflg* specifies both **IPC_CREAT** and **IPC_EXCL** and a shared memory segment already exists for *key*, then **shmget**() fails with *errno* set to **EEXIST**. (This is analogous to the effect of the combination **O_CREAT | O_EXCL** for **open**(2).)

The value *shmflg* is composed of:

IPC_CREAT
to create a new segment. If this flag is not used, then **shmget**() will find the segment associated with *key* and check to see if the user has permission to access the segment.

IPC_EXCL used with **IPC_CREAT** to ensure failure if the segment already exists.

mode_flags (least significant 9 bits) specifying the permissions granted to the owner, group, and world. These bits have the same format, and the same meaning, as the *mode* argument of **open**(2). Presently, the execute permissions are not used by the system.

SHM_HUGETLB (since Linux 2.6)
Allocate the segment using "huge pages." See the kernel source file *Documentation/vm/hugetlbpage.txt* for further information.

SHM_NORESERVE (since Linux 2.6.15)
This flag serves the same purpose as the **mmap**(2) **MAP_NORESERVE** flag. Do not reserve swap space for this segment. When swap space is reserved, one has the guarantee that it is possible to modify the segment. When swap space is not reserved one might get **SIGSEGV** upon a write if no physical memory is available. See also the discussion of the file */proc/sys/vm/overcommit_memory* in **proc**(5).

When a new shared memory segment is created, its contents are initialized to zero values, and its associated data structure, *shmid_ds* (see **shmctl**(2)), is initialized as follows:

shm_perm.cuid and *shm_perm.uid* are set to the effective user ID of the calling process.

shm_perm.cgid and *shm_perm.gid* are set to the effective group ID of the calling process.

The least significant 9 bits of *shm_perm.mode* are set to the least significant 9 bit of *shmflg*.

shm_segsz is set to the value of *size*.

shm_lpid, *shm_nattch*, *shm_atime* and *shm_dtime* are set to 0.

shm_ctime is set to the current time.

If the shared memory segment already exists, the permissions are verified, and a check is made to see if it is marked for destruction.

RETURN VALUE

A valid segment identifier, *shmid*, is returned on success, −1 on error.

ERRORS

On failure, *errno* is set to one of the following:

EACCES
The user does not have permission to access the shared memory segment, and does not have the **CAP_IPC_OWNER** capability.

EEXIST
IPC_CREAT | IPC_EXCL was specified and the segment exists.

EINVAL
A new segment was to be created and *size* < **SHMMIN** or *size* > **SHMMAX**, or no new segment was to be created, a segment with given key existed, but *size* is greater than the size of that segment.

ENFILE
The system limit on the total number of open files has been reached.

ENOENT
No segment exists for the given *key*, and **IPC_CREAT** was not specified.

ENOMEM
No memory could be allocated for segment overhead.

ENOSPC
All possible shared memory IDs have been taken (**SHMMNI**), or allocating a segment of the requested *size* would cause the system to exceed the system-wide limit on shared memory (**SHMALL**).

EPERM
The **SHM_HUGETLB** flag was specified, but the caller was not privileged (did not have the **CAP_IPC_LOCK** capability).

CONFORMING TO

SVr4, POSIX.1-2001.

SHM_HUGETLB is a non-portable Linux extension.

NOTES

IPC_PRIVATE isn't a flag field but a *key_t* type. If this special value is used for *key*, the system call ignores everything but the least significant 9 bits of *shmflg* and creates a new shared memory segment (on success).

The following limits on shared memory segment resources affect the **shmget**() call:

SHMALL
System wide maximum of shared memory pages (on Linux, this limit can be read and modified via */proc/sys/kernel/shmall*).

SHMMAX
Maximum size in bytes for a shared memory segment: policy dependent (on Linux, this limit can be read and modified via */proc/sys/kernel/shmmax*).

SHMMIN
Minimum size in bytes for a shared memory segment: implementation dependent (currently 1 byte, though **PAGE_SIZE** is the effective minimum size).

SHMMNI
System wide maximum number of shared memory segments: implementation dependent (currently 4096, was 128 before Linux 2.3.99; on Linux, this limit can be read and modified via */proc/sys/kernel/shmmni*).

The implementation has no specific limits for the per-process maximum number of shared memory segments (**SHMSEG**).

Linux Notes

Until version 2.3.30 Linux would return **EIDRM** for a **shmget**() on a shared memory segment scheduled for deletion.

BUGS

The name choice **IPC_PRIVATE** was perhaps unfortunate, **IPC_NEW** would more clearly show its function.

SEE ALSO

shmat(2), **shmctl**(2), **shmdt**(2), **ftok**(3), **capabilities**(7), **shm_overview**(7), **svipc**(7)

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/.