NAME

getgroups, setgroups − get/set list of supplementary group IDs

SYNOPSIS

```
#include <sys/types.h>
#include <unistd.h>

int getgroups(int size, gid_t list[]);

#include <grp.h>

int setgroups(size_t size, const gid_t *list);
```

Feature Test Macro Requirements for glibc (see **feature_test_macros**(7)):

setgroups(): _BSD_SOURCE

DESCRIPTION

getgroups() returns the supplementary group IDs of the calling process in *list*. The argument *size* should be set to the maximum number of items that can be stored in the buffer pointed to by *list*. If the calling process is a member of more than *size* supplementary groups, then an error results. It is unspecified whether the effective group ID of the calling process is included in the returned list. (Thus, an application should also call **getegid**(2) and add or remove the resulting value.)

If *size* is zero, *list* is not modified, but the total number of supplementary group IDs for the process is returned. This allows the caller to determine the size of a dynamically allocated *list* to be used in a further call to **getgroups**().

setgroups() sets the supplementary group IDs for the calling process. Appropriate privileges (Linux: the **CAP_SETGID** capability) are required. The *size* argument specifies the number of supplementary group IDs in the buffer pointed to by *list*.

RETURN VALUE

On success, **getgroups**() returns the number of supplementary group IDs. On error, −1 is returned, and *errno* is set appropriately.

On success, **setgroups**() returns 0. On error, −1 is returned, and *errno* is set appropriately.

ERRORS

EFAULT
list has an invalid address.

getgroups() can additionally fail with the following error:

EINVAL
size is less than the number of supplementary group IDs, but is not zero.

setgroups() can additionally fail with the following errors:

EINVAL
size is greater than **NGROUPS_MAX** (32 before Linux 2.6.4; 65536 since Linux 2.6.4).

ENOMEM
Out of memory.

EPERM
The calling process has insufficient privilege.

CONFORMING TO

SVr4, 4.3BSD. The **getgroups**() function is in POSIX.1-2001. Since **setgroups**() requires privilege, it is not covered by POSIX.1-2001.

NOTES

A process can have up to **NGROUPS_MAX** supplementary group IDs in addition to the effective group ID. The set of supplementary group IDs is inherited from the parent process, and preserved across an **execve**(2).

The maximum number of supplementary group IDs can be found using **sysconf**(3):

```
long ngroups_max;
ngroups_max = sysconf(_SC_NGROUPS_MAX);
```

The maximum return value of **getgroups**() cannot be larger than one more than this value.

SEE ALSO

getgid(2), **setgid**(2), **getgrouplist**(3), **initgroups**(3), **capabilities**(7), **credentials**(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/.