



**Full credit is given to the above companies including the Operating System (OS) that this PDF file was generated!**

### ***Rocky Enterprise Linux 9.2 Manual Pages on command 'setgid32.2'***

**\$ man setgid32.2**

SETGID(2)                      Linux Programmer's Manual                      SETGID(2)

#### NAME

setgid - set group identity

#### SYNOPSIS

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

```
#include <unistd.h>
```

```
int setgid(gid_t gid);
```

#### DESCRIPTION

setgid() sets the effective group ID of the calling process. If the calling process is privileged (more precisely: has the CAP\_SETGID capability in its user namespace), the real GID and saved set-group-ID are also set.

Under Linux, setgid() is implemented like the POSIX version with the \_POSIX\_SAVED\_IDS feature. This allows a set-group-ID program that is not set-user-ID-root to drop all of its group privileges, do some un-privileged work, and then reengage the original effective group ID in a secure manner.

#### RETURN VALUE

On success, zero is returned. On error, -1 is returned, and errno is set appropriately.

#### ERRORS

EINVAL The group ID specified in gid is not valid in this user namespace.

EPERM The calling process is not privileged (does not have the CAP\_SETGID capability in its user namespace), and gid does not match the real group ID or saved set-group-ID of the calling process.

#### CONFORMING TO

POSIX.1-2001, POSIX.1-2008, SVr4.

## NOTES

The original Linux `setgid()` system call supported only 16-bit group IDs. Subsequently, Linux 2.4 added `setgid32()` supporting 32-bit IDs. The glibc `setgid()` wrapper function transparently deals with the variation across kernel versions.

### C library/kernel differences

At the kernel level, user IDs and group IDs are a per-thread attribute. However, POSIX requires that all threads in a process share the same credentials. The NPTL threading implementation handles the POSIX requirements by providing wrapper functions for the various system calls that change process UIDs and GIDs. These wrapper functions (including the one for `setgid()`) employ a signal-based technique to ensure that when one thread changes credentials, all of the other threads in the process also change their credentials. For details, see `nptl(7)`.

## SEE ALSO

`getgid(2)`, `setegid(2)`, `setregid(2)`, `capabilities(7)`, `credentials(7)`, `user_namespaces(7)`

## COLOPHON

This page is part of release 5.10 of the Linux man-pages project. A description of the project, information about reporting bugs, and the latest version of this page, can be found at <https://www.kernel.org/doc/man-pages/>.

Linux

2019-03-06

SETGID(2)