NAME

chmod, fchmod − change permissions of a file

SYNOPSIS

#include <sys/stat.h>

int chmod(const char ****path*, mode_t** *mode***);**
int fchmod(int *fd***, mode_t** *mode***);**

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

fchmod(): _BSD_SOURCE || _XOPEN_SOURCE >= 500

DESCRIPTION

These system calls change the permissions of a file. They differ only in how the file is specified:

* **chmod**() changes the permissions of the file specified whose pathname is given in *path*, which is dereferenced if it is a symbolic link.
* **fchmod**() changes the permissions of the file referred to by the open file descriptor *fd*.

The new file permissions are specified in *mode*, which is a bit mask created by ORing together zero or more of the following:

S_ISUID	(04000)	set-user-ID (set process effective user ID on **execve**(2))
S_ISGID	(02000)	set-group-ID (set process effective group ID on **execve**(2); mandatory locking, as described in **fcntl**(2); take a new file's group from parent directory, as described in **chown**(2) and **mkdir**(2))
S_ISVTX	(01000)	sticky bit (restricted deletion flag, as described in **unlink**(2))
S_IRUSR	(00400)	read by owner
S_IWUSR	(00200)	write by owner
S_IXUSR	(00100)	execute/search by owner ("search" applies for directories, and means that entries within the directory can be accessed)
S_IRGRP	(00040)	read by group
S_IWGRP	(00020)	write by group
S_IXGRP	(00010)	execute/search by group
S_IROTH	(00004)	read by others
S_IWOTH	(00002)	write by others
S_IXOTH	(00001)	execute/search by others

The effective UID of the calling process must match the owner of the file, or the process must be privileged (Linux: it must have the **CAP_FOWNER** capability).

If the calling process is not privileged (Linux: does not have the **CAP_FSETID** capability), and the group of the file does not match the effective group ID of the process or one of its supplementary group IDs, the **S_ISGID** bit will be turned off, but this will not cause an error to be returned.

As a security measure, depending on the file system, the set-user-ID and set-group-ID execution bits may be turned off if a file is written. (On Linux this occurs if the writing process does not have the **CAP_FSETID** capability.) On some file systems, only the superuser can set the sticky bit, which may have a special meaning. For the sticky bit, and for set-user-ID and set-group-ID bits on directories, see **stat**(2).

On NFS file systems, restricting the permissions will immediately influence already open files, because the access control is done on the server, but open files are maintained by the client. Widening the permissions

may be delayed for other clients if attribute caching is enabled on them.

RETURN VALUE

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

ERRORS

Depending on the file system, other errors can be returned. The more general errors for **chmod**() are listed below:

EACCES
Search permission is denied on a component of the path prefix. (See also **path_resolution**(7).)

EFAULT
path points outside your accessible address space.

EIO An I/O error occurred.

ELOOP
Too many symbolic links were encountered in resolving *path*.

ENAMETOOLONG
path is too long.

ENOENT
The file does not exist.

ENOMEM
Insufficient kernel memory was available.

ENOTDIR
A component of the path prefix is not a directory.

EPERM
The effective UID does not match the owner of the file, and the process is not privileged (Linux: it does not have the **CAP_FOWNER** capability).

EROFS
The named file resides on a read-only file system.

The general errors for **fchmod**() are listed below:

EBADF
The file descriptor *fd* is not valid.

EIO See above.

EPERM
See above.

EROFS
See above.

CONFORMING TO

4.4BSD, SVr4, POSIX.1-2001.

SEE ALSO

chown(2), **execve**(2), **fchmodat**(2), **open**(2), **stat**(2), **path_resolution**(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/.