## NAME

**getfh**, **lgetfh**, **getfhat** - get file handle

## LIBRARY

Standard C Library (libc, -lc)

## SYNOPSIS

**#include <sys/param.h>**
**#include <sys/mount.h>**

*int*
**getfh**(*const char *path, fhandle_t *fhp*);

*int*
**lgetfh**(*const char *path, fhandle_t *fhp*);

*int*
**getfhat**(*int fd, const char *path, fhandle_t *fhp, int flag*);

## DESCRIPTION

The **getfh**() system call returns a file handle for the specified file or directory in the file handle pointed to by *fhp*.

The **lgetfh**() system call is like **getfh**() except in the case where the named file is a symbolic link, in which case **lgetfh**() returns information about the link, while **getfh**() returns information about the file the link references.

The **getfhat**() system call is equivalent to **getfh**() and **lgetfh**() except when the *path* specifies a relative path. For **getfhat**() and relative *path*, the status is retrieved from a file relative to the directory associated with the file descriptor *fd* instead of the current working directory.

The values for the *flag* are constructed by a bitwise-inclusive OR of flags from this list, defined in *<fcntl.h>*:

AT_SYMLINK_NOFOLLOW
: If *path* names a symbolic link, the status of the symbolic link is returned.

AT_RESOLVE_BENEATH
: Only walk paths below the directory specified by the *fd* descriptor. See the description of the O_RESOLVE_BENEATH flag in the open(2) manual page.

If **getfhat**() is passed the special value AT_FDCWD in the *fd* parameter, the current working directory is used and the behavior is identical to a call to **getfth**() or **lgetfh**() respectively, depending on whether or not the AT_SYMLINK_NOFOLLOW bit is set in *flag*.

When **getfhat**() is called with an absolute *path*, it ignores the *fd* argument.

These system calls are restricted to the superuser.

## RETURN VALUES

Upon successful completion, the value 0 is returned; otherwise the value -1 is returned and the global variable *errno* is set to indicate the error.

## ERRORS

The **getfh**() and **lgetfh**() system calls fail if one or more of the following are true:

[EPERM] The caller does not have appropriate privilege to perform the operation.

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

[ENAMETOOLONG] The length of a component of *path* exceeds 255 characters, or the length of *path* exceeds 1023 characters.

[ENOENT] The file referred to by *path* does not exist.

[EACCES] Search permission is denied for a component of the path prefix of *path*.

[ELOOP] Too many symbolic links were encountered in translating *path*.

[EFAULT] The *fhp* argument points to an invalid address.

[EFAULT] The *path* argument points to an invalid address.

[EIO] An I/O error occurred while reading from or writing to the file system.

[EINTEGRITY] Corrupted data was detected while reading from the file system.

[ESTALE] The file handle *fhp* is no longer valid.

In addition to the errors returned by **getfh**(), and **lgetfh**(), the **getfhat**() system call may fail if:

[EBADF] The *path* argument does not specify an absolute path and the *fd* argument, is neither AT_FDCWD nor a valid file descriptor open for searching.

[EINVAL] The value of the *flag* argument is not valid.

[ENOTDIR] The *path* argument is not an absolute path and *fd* is neither AT_FDCWD nor a file descriptor associated with a directory.

## SEE ALSO

fhopen(2), open(2), stat(2)

## HISTORY

The **getfh**() system call first appeared in 4.4BSD. The **lgetfh**() system call first appeared in FreeBSD 5.3. The **getfhat**() system call first appeared in FreeBSD 12.1.