NAME

attr_get, attr_getf − get the value of a user attribute of a filesystem object

C SYNOPSIS

#include <attr/attributes.h>

int attr_get (const char ****path*, const char *****attrname***,**
char *****attrvalue*, **int *****valuelength***, **int** *flags***);**

int attr_getf (int *fd***, const char *****attrname***,**
char *****attrvalue*, **int *****valuelength***, **int** *flags***);**

DESCRIPTION

The **attr_get** and **attr_getf** functions provide a way to retrieve the value of an attribute.

Path points to a path name for a filesystem object, and *fd* refers to the file descriptor associated with a file. If the attribute *attrname* exists, the value associated with it will be copied into the *attrvalue* buffer. The *valuelength* argument is an input/output argument that on the call to **attr_get** should contain the maximum size of attribute value the process is willing to accept. On return, the *valuelength* will have been modified to show the actual size of the attribute value returned. The *flags* argument can contain the following symbols bitwise OR'ed together:

ATTR_ROOT
: Look for *attrname* in the **root** address space, not in the **user** address space. (limited to use by super-user only)

ATTR_DONTFOLLOW
: Do not follow symbolic links when resolving a *path* on an **attr_get** function call. The default is to follow symbolic links.

attr_get will fail if one or more of the following are true:

[ENOATTR] The attribute name given is not associated with the indicated filesystem object.

[E2BIG] The value of the given attribute is too large to fit into the buffer. The integer that the *valuelength* argument points to has been modified to show the actual number of bytes that would be required to store the value of that attribute.

[ENOENT] The named file does not exist.

[EPERM] The effective user ID does not match the owner of the file and the effective user ID is not super-user.

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

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

[EINVAL] A bit was set in the *flag* argument that is not defined for this system call.

[EFAULT] *Path*, *attrname*, *attrvalue*, or *valuelength* points outside the allocated address space of the process.

[ELOOP] A path name lookup involved too many symbolic links.

[ENAMETOOLONG] The length of *path* exceeds {*MAXPATHLEN*}, or a pathname component is longer than {*MAXNAMELEN*}.

attr_getf will fail if:

[ENOATTR] The attribute name given is not associated with the indicated filesystem object.

[E2BIG] The value of the given attribute is too large to fit into the buffer. The integer that the *valuelength* argument points to has been modified to show the actual number of bytes that would be required to store the value of that attribute.

[EINVAL] A bit was set in the *flag* argument that is not defined for this system call, or *fd* refers to a socket, not a file.

[EFAULT] *Attrname*, *attrvalue*, or *valuelength* points outside the allocated address space of the process.

[EBADF] *Fd* does not refer to a valid descriptor.

DIAGNOSTICS

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

SEE ALSO

attr(1), **attr_list**(3), **attr_multi**(3), **attr_remove**(3), **attr_set**(3)