NAME

acl_get_file — get an ACL by filename

LIBRARY

Linux Access Control Lists library (libacl, −lacl).

SYNOPSIS

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

acl_t
acl_get_file(const char *path_p, acl_type_t type);
```

DESCRIPTION

The **acl_get_file**() function retrieves the access ACL associated with a file or directory, or the default ACL associated with a directory. The pathname for the file or directory is pointed to by the argument *path_p*. The ACL is placed into working storage and **acl_get_file**() returns a pointer to that storage.

In order to read an ACL from an object, a process must have read access to the object's attributes.

The value of the argument *type* is used to indicate whether the access ACL or the default ACL associated with *path_p* is returned. If *type* is ACL_TYPE_ACCESS, the access ACL of *path_p* is returned. If *type* is ACL_TYPE_DEFAULT, the default ACL of *path_p* is returned. If *type* is ACL_TYPE_DEFAULT and no default ACL is associated with the directory *path_p*, then an ACL containing zero ACL entries is returned. If *type* specifies a type of ACL that cannot be associated with *path_p*, then the function fails.

This function may cause memory to be allocated. The caller should free any releasable memory, when the new ACL is no longer required, by calling *acl_free*(3) with the *(void*)acl_t* returned by **acl_get_file**() as an argument.

RETURN VALUE

On success, this function returns a pointer to the working storage. On error, a value of `(acl_t)NULL` is returned, and *errno* is set appropriately.

ERRORS

If any of the following conditions occur, the **acl_get_file**() function returns a value of `(acl_t)NULL` and sets *errno* to the corresponding value:

[EACCES]	Search permission is denied for a component of the path prefix or the object exists and the process does not have appropriate access rights. Argument *type* specifies a type of ACL that cannot be associated with *path_p*.
[EINVAL]	The argument *type* is not ACL_TYPE_ACCESS or ACL_TYPE_DEFAULT.
[ENAMETOOLONG]	The length of the argument *path_p* is too long.
[ENOENT]	The named object does not exist or the argument *path_p* points to an empty string.
[ENOMEM]	The ACL working storage requires more memory than is allowed by the hardware or system-imposed memory management constraints.
[ENOTDIR]	A component of the path prefix is not a directory.
[ENOTSUP]	The file system on which the file identified by *path_p* is located does not support ACLs, or ACLs are disabled.

STANDARDS

IEEE Std 1003.1e draft 17 ("POSIX.1e", abandoned)

SEE ALSO

acl_free(3), *acl_get_entry*(3), *acl_get_fd*(3), *acl_set_file*(3), *acl*(5)

AUTHOR

Derived from the FreeBSD manual pages written by Robert N M Watson <rwatson@FreeBSD.org>, and adapted for Linux by Andreas Gruenbacher <andreas.gruenbacher@gmail.com>.