NAME

acl_extended_file, acl_extended_file_nofollow — test for information in ACLs by file name

LIBRARY

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

SYNOPSIS

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

int
acl_extended_file(const char *path_p);

int
acl_extended_file_nofollow(const char *path_p);
```

DESCRIPTION

The **acl_extended_file**() function returns `1` if the file or directory referred to by the argument *path_p* is associated with an extended access ACL, or if the directory referred to by *path_p* is associated with a default ACL. The function returns `0` if the file has neither an extended access ACL nor a default ACL.

An extended ACL is an ACL that contains entries other than the three required entries of tag types ACL_USER_OBJ, ACL_GROUP_OBJ and ACL_OTHER. If the result of the**acl_extended_file**() function for a file object is `0`, then ACLs define no discretionary access rights other than those already defined by the traditional file permission bits.

Access to the file object may be further restricted by other mechanisms, such as Mandatory Access Control schemes. The *access*(2) system call can be used to check whether a given type of access to a file object would be granted.

acl_extended_file_nofollow() is identical to **acl_extended_file**(), except in the case of a symbolic link, where the link itself is interrogated, not the file that it refers to. Since symbolic links have no ACL themselves, the operation is supposed to fail on them.

RETURN VALUE

If successful, the **acl_extended_file**() function returns `1` if the file object referred to by *path_p* has an extended access ACL or a default ACL, and `0` if the file object referred to by *path_p* has neither an extended access ACL nor a default ACL. Otherwise, the value `-1` is returned and the global variable *errno* is set to indicate the error.

ERRORS

If any of the following conditions occur, the **acl_extended_file**() function returns `-1` and sets *errno* to the corresponding value:

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

This is a non-portable, Linux specific extension to the ACL manipulation functions defined in IEEE Std 1003.1e draft 17 (“POSIX.1e”, abandoned).

SEE ALSO

access(2), *acl_get_file*(3), *acl*(5)

AUTHOR

Written by Andreas Gruenbacher <andreas.gruenbacher@gmail.com>.