



Rocky Enterprise Linux 9.2 Manual Pages on command 'fexecve.3'

C:\>man fexecve.3

FEXECVE(3) Linux Programmer's Manual FEXECVE(3)

NAME

fexecve - execute program specified via file descriptor

SYNOPSIS

```
#include <unistd.h>
```

```
int fexecve(int fd, char *const argv[], char *const envp[]);
```

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

fexecve():

Since glibc 2.10:

```
_POSIX_C_SOURCE >= 200809L
```

Before glibc 2.10:

```
_GNU_SOURCE
```

DESCRIPTION

fexecve() performs the same task as execve(2), with the difference that the file to be executed is specified via a file descriptor, fd, rather than via a pathname.

The file descriptor fd must be opened read-only (O_RDONLY) or with the O_PATH flag and the caller must have permission to execute the file that it refers to.

RETURN VALUE

A successful call to fexecve() never returns. On error, the function does return, with a result value of -1, and errno is set appropriately.

ERRORS

Errors are as for execve(2), with the following additions:

EINVAL fd is not a valid file descriptor, or argv is NULL, or envp is NULL.

ENOENT The close-on-exec flag is set on fd, and fd refers to a script. See BUGS.

ENOSYS The kernel does not provide the execveat(2) system call, and the /proc filesystem could not be accessed.

VERSIONS

fexecve() is implemented since glibc 2.3.2.

ATTRIBUTES

For an explanation of the terms used in this section, see attributes(7).

??

?Interface ? Attribute ? Value ?

??

?fexecve() ? Thread safety ? MT-Safe ?

??

CONFORMING TO

POSIX.1-2008. This function is not specified in POSIX.1-2001, and is not widely available on other systems. It is specified in POSIX.1-2008.

NOTES

On Linux with glibc versions 2.26 and earlier, fexecve() is implemented using the proc(5) filesystem, so /proc needs to be mounted and available at the time of the call. Since glibc 2.27, if the underlying kernel supports the execveat(2) system call, then fexecve() is implemented using that system call, with the benefit that /proc does not need to be mounted.

The idea behind fexecve() is to allow the caller to verify (checksum) the contents of an executable before executing it. Simply opening the file, checksumming the contents, and then doing an execve(2) would not suffice, since, between the two steps, the filename, or a directory prefix of the pathname, could have been exchanged (by, for example, modifying the target of a symbolic link). fexecve() does not mitigate the problem that the contents of a file could be changed between the checksumming and the call to fexecve(); for that, the solution is to ensure that the permissions on the file prevent it from being modified by malicious users.

The natural idiom when using fexecve() is to set the close-on-exec flag on fd, so that the file descriptor does not leak through to the program that is executed.

This approach is natural for two reasons. First, it prevents file descriptors be?

ing consumed unnecessarily. (The executed program normally has no need of a file descriptor that refers to the program itself.) Second, if `fexecve()` is used recursively, employing the `close-on-exec` flag prevents the file descriptor exhaustion that would result from the fact that each step in the recursion would cause one more file descriptor to be passed to the new program. (But see BUGS.)

BUGS

If `fd` refers to a script (i.e., it is an executable text file that names a script interpreter with a first line that begins with the characters `#!`) and the `close-on-exec` flag has been set for `fd`, then `fexecve()` fails with the error `ENOENT`. This error occurs because, by the time the script interpreter is executed, `fd` has already been closed because of the `close-on-exec` flag. Thus, the `close-on-exec` flag can't be set on `fd` if it refers to a script, leading to the problems described in NOTES.

SEE ALSO

`execve(2)`, `execveat(2)`

COLOPHON

This page is part of release 5.05 of the Linux man-pages project. A description of the project, information about reporting bugs, and the latest version of this page, can be found at <https://www.kernel.org/doc/man-pages/>.