NAME
       splice - splice data to/from a pipe

SYNOPSIS
       #define _GNU_SOURCE /* See feature_test_macros(7)
       */"
       #include <fcntl.h>

       ssize_t splice(int *fd_in*, **loff_t ****off_in*, **int** *fd_out*,
        loff_t *off_out, **size_t** *len*, **unsigned int** *flags*)**;**

DESCRIPTION
       splice() moves data between two file descriptors without copying between kernel address space and user
       address space. It transfers up to *len* bytes of data from the file descriptor *fd_in* to the file descriptor *fd_out*,
       where one of the file descriptors must refer to a pipe.

       The following semantics apply for *fd_in* and *off_in*:

       *   If *fd_in* refers to a pipe, then *off_in* must be NULL.

       *   If *fd_in* does not refer to a pipe and *off_in* is NULL, then bytes are read from *fd_in* starting from the
           file offset, and the file offset is adjusted appropriately.

       *   If *fd_in* does not refer to a pipe and *off_in* is not NULL, then *off_in* must point to a buffer which speci-
           fies the starting offset from which bytes will be read from *fd_in*; in this case, the file offset of *fd_in* is
           not changed.

       Analogous statements apply for *fd_out* and *off_out*.

       The *flags* argument is a bit mask that is composed by ORing together zero or more of the following values:

       SPLICE_F_MOVE   Attempt to move pages instead of copying. This is only a hint to the ker-
                                   nel: pages may still be copied if the kernel cannot move the pages from
                                   the pipe, or if the pipe buffers don't refer to full pages. The initial imple-
                                   mentation of this flag was buggy: therefore starting in Linux 2.6.21 it is a
                                   no-op (but is still permitted in a **splice**() call); in the future, a correct
                                   implementation may be restored.

       SPLICE_F_NONBLOCK   Do not block on I/O. This makes the splice pipe operations nonblocking,
                                   but **splice**() may nevertheless block because the file descriptors that are
                                   spliced to/from may block (unless they have the **O_NONBLOCK** flag set).

       SPLICE_F_MORE   More data will be coming in a subsequent splice. This is a helpful hint
                                   when the *fd_out* refers to a socket (see also the description of
                                   MSG_MORE in send(2), and the description of **TCP_CORK** in tcp(7)).

       SPLICE_F_GIFT   Unused for **splice**(); see vmsplice(2).

RETURN VALUE
       Upon successful completion, **splice**() returns the number of bytes spliced to or from the pipe.

       A return value of 0 means end of input. If *fd_in* refers to a pipe, then this means that there was no data to
       transfer, and it would not make sense to block because there are no writers connected to the write end of the
       pipe.

       On error, **splice**() returns -1 and *errno* is set to indicate the error.

ERRORS
       EAGAIN
               SPLICE_F_NONBLOCK was specified in *flags*, and the operation would block.

       EBADF
               One or both file descriptors are not valid, or do not have proper read-write mode.

EINVAL
Target filesystem doesn't support splicing; target file is opened in append mode; neither of the file descriptors refers to a pipe; or offset given for nonseekable device.

ENOMEM
Out of memory.

ESPIPE
Either *off_in* or *off_out* was not NULL, but the corresponding file descriptor refers to a pipe.

VERSIONS
The **splice**() system call first appeared in Linux 2.6.17; library support was added to glibc in version 2.5.

CONFORMING TO
This system call is Linux-specific.

NOTES
The three system calls **splice**(), vmsplice(2), and tee(2), provide user-space programs with full control over an arbitrary kernel buffer, implemented within the kernel using the same type of buffer that is used for a pipe. In overview, these system calls perform the following tasks:

splice()   moves data from the buffer to an arbitrary file descriptor, or vice versa, or from one buffer to another.

tee(2) "copies" the data from one buffer to another.

vmsplice(2) "copies" data from user space into the buffer.

Though we talk of copying, actual copies are generally avoided. The kernel does this by implementing a pipe buffer as a set of reference-counted pointers to pages of kernel memory. The kernel creates "copies" of pages in a buffer by creating new pointers (for the output buffer) referring to the pages, and increasing the reference counts for the pages: only pointers are copied, not the pages of the buffer.

EXAMPLE
See tee(2).

SEE ALSO
copy_file_range(2), sendfile(2), tee(2), vmsplice(2)

COLOPHON
This page is part of release 4.10 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/.