NAME

tkill, tgkill − send a signal to a thread

SYNOPSIS

int tkill(int *tid***, int** *sig***);**

int tgkill(int *tgid***, int** *tid***, int** *sig***);**

DESCRIPTION

tgkill() sends the signal *sig* to the thread with the thread ID *tid* in the thread group *tgid*. (By contrast, **kill**(2) can only be used to send a signal to a process (i.e., thread group) as a whole, and the signal will be delivered to an arbitrary thread within that process.)

tkill() is an obsolete predecessor to **tgkill**(). It only allows the target thread ID to be specified, which may result in the wrong thread being signaled if a thread terminates and its thread ID is recycled. Avoid using this system call.

If *tgid* is specified as −1, **tgkill**() is equivalent to **tkill**().

These are the raw system call interfaces, meant for internal thread library use.

RETURN VALUE

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

ERRORS

EINVAL
An invalid thread ID, thread group ID, or signal was specified.

EPERM
Permission denied. For the required permissions, see **kill**(2).

ESRCH
No process with the specified thread ID (and thread group ID) exists.

VERSIONS

tkill() is supported since Linux 2.4.19 / 2.5.4. **tgkill**() was added in Linux 2.5.75.

CONFORMING TO

tkill() and **tgkill**() are Linux-specific and should not be used in programs that are intended to be portable.

NOTES

See the description of **CLONE_THREAD** in **clone**(2) for an explanation of thread groups.

Glibc does not provide wrappers for these system calls; call them using **syscall**(2).

SEE ALSO

clone(2), **gettid**(2), **kill**(2)

COLOPHON

This page is part of release 3.22 of the Linux *man-pages* project. A description of the project, and information about reporting bugs, can be found at http://www.kernel.org/doc/man-pages/.