Rocky Enterprise Linux 9.2 Manual Pages on command 'rt_sigqueueinfo.2'

$ man rt_sigqueueinfo.2

RT_SIGQUEUEINFO(2) Linux Programmer's Manual RT_SIGQUEUEINFO(2)

NAME

rt_sigqueueinfo, rt_tgsigqueueinfo - queue a signal and data

SYNOPSIS

```
int rt_sigqueueinfo(pid_t tgid, int sig, siginfo_t *info);
int rt_tgsigqueueinfo(pid_t tgid, pid_t tid, int sig,
                      siginfo_t *info);
```

Note: There are no glibc wrappers for these system calls; see NOTES.

DESCRIPTION

The rt_sigqueueinfo() and rt_tgsigqueueinfo() system calls are the low-level interfaces used to send a signal plus data to a process or thread. The receiver of the signal can obtain the accompanying data by establishing a signal handler with the sigaction(2) SA_SIGINFO flag.

These system calls are not intended for direct application use; they are provided to allow the implementation of sigqueue(3) and pthread_sigqueue(3).

The rt_sigqueueinfo() system call sends the signal sig to the thread group with the ID tgid. (The term "thread group" is synonymous with

"process", and tid corresponds to the traditional UNIX process ID.)
The signal will be delivered to an arbitrary member of the thread group
(i.e., one of the threads that is not currently blocking the signal).
The info argument specifies the data to accompany the signal. This ar?
gument is a pointer to a structure of type siginfo_t, described in
sigaction(2) (and defined by including <sigaction.h>). The caller
should set the following fields in this structure:

si_code

This should be one of the SI_* codes in the Linux kernel source file include/asm-generic/siginfo.h. If the signal is being sent to any process other than the caller itself, the following re? strictions apply:

* The code can't be a value greater than or equal to zero. In particular, it can't be SI_USER, which is used by the kernel to indicate a signal sent by kill(2), and nor can it be SI_KERNEL, which is used to indicate a signal generated by the kernel.
* The code can't (since Linux 2.6.39) be SI_TKILL, which is used by the kernel to indicate a signal sent using tgkill(2).

si_pid This should be set to a process ID, typically the process ID of the sender.

si_uid This should be set to a user ID, typically the real user ID of the sender.

si_value

This field contains the user data to accompany the signal. For more information, see the description of the last (union sigval) argument of sigqueue(3).

Internally, the kernel sets the si_signo field to the value specified in sig, so that the receiver of the signal can also obtain the signal number via that field.

The rt_tgsigqueueinfo() system call is like rt_sigqueueinfo(), but sends the signal and data to the single thread specified by the combi? nation of tgid, a thread group ID, and tid, a thread in that thread

group.

RETURN VALUE

On success, these system calls return 0. On error, they return -1 and errno is set to indicate the error.

ERRORS

EAGAIN The limit of signals which may be queued has been reached. (See signal(7) for further information.)

EINVAL sig, tgid, or tid was invalid.

EPERM The caller does not have permission to send the signal to the target. For the required permissions, see kill(2).

EPERM tgid specifies a process other than the caller and info->si_code is invalid.

ESRCH rt_sigqueueinfo(): No thread group matching tgid was found.

rt_tgsigqueinfo(): No thread matching tgid and tid was found.

VERSIONS

The rt_sigqueueinfo() system call was added to Linux in version 2.2.

The rt_tgsigqueueinfo() system call was added to Linux in version 2.6.31.

CONFORMING TO

These system calls are Linux-specific.

NOTES

Since these system calls are not intended for application use, there are no glibc wrapper functions; use syscall(2) in the unlikely case that you want to call them directly.

As with kill(2), the null signal (0) can be used to check if the speci? fied process or thread exists.

SEE ALSO

kill(2), pidfd_send_signal(2), sigaction(2), sigprocmask(2), tgkill(2), pthread_sigqueue(3), sigqueue(3), signal(7)

COLOPHON

This page is part of release 5.10 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/.