NAME
>   nptl – Native POSIX Threads Library

DESCRIPTION
>   NPTL (Native POSIX Threads Library) is the GNU C library POSIX threads implementation that is used
>   on modern Linux systems.

NPTL and signals
>   NPTL makes internal use of the first two real-time signals (signal numbers 32 and 33). One of these signals
>   is used to support thread cancelation and POSIX timers (see **timer_create**(2)); the other is used as part of a
>   mechanism that ensures all threads in a process always have the same UIDs and GIDs, as required by
>   POSIX. These signals cannot be used in applications.
>
>   To prevent accidental use of these signals in applications, which might interfere with the operation of the
>   NPTL implementation, various glibc library functions and system call wrapper functions attempt to hide
>   these signals from applications, as follows:
>
>   •   **SIGRTMIN** is defined with the value 34 (rather than 32).
>
>   •   The **sigwaitinfo**(2), **sigtimedwait**(2), and **sigwait**(3) interfaces silently ignore requests to wait for these
>   two signals if they are specified in the signal set argument of these calls.
>
>   •   The **sigprocmask**(2) and **pthread_sigmask**(3) interfaces silently ignore attempts to block these two
>   signals.
>
>   •   The **sigaction**(2), **pthread_kill**(3), and **pthread_sigqueue**(3) interfaces fail with the error **EINVAL**
>   (indicating an invalid signal number) if these signals are specified.
>
>   •   **sigfillset**(3) does not include these two signals when it creates a full signal set.

NPTL and process credential changes
>   At the Linux kernel level, credentials (user and group IDs) are a per-thread attribute. However, POSIX re-
>   quires that all of the POSIX threads in a process have the same credentials. To accommodate this require-
>   ment, the NPTL implementation wraps all of the system calls that change process credentials with func-
>   tions that, in addition to invoking the underlying system call, arrange for all other threads in the process to
>   also change their credentials.
>
>   The implementation of each of these system calls involves the use of a real-time signal that is sent (using
>   **tgkill**(2)) to each of the other threads that must change its credentials. Before sending these signals, the
>   thread that is changing credentials saves the new credential(s) and records the system call being employed
>   in a global buffer. A signal handler in the receiving thread(s) fetches this information and then uses the
>   same system call to change its credentials.
>
>   Wrapper functions employing this technique are provided for **setgid**(2), **setuid**(2), **setegid**(2), **seteuid**(2),
>   **setregid**(2), **setreuid**(2), **setresgid**(2), **setresuid**(2), and **setgroups**(2).

STANDARDS
>   For details of the conformance of NPTL to the POSIX standard, see **pthreads**(7).

NOTES
>   POSIX says that any thread in any process with access to the memory containing a process-shared
>   (**PTHREAD_PROCESS_SHARED**) mutex can operate on that mutex. However, on 64-bit x86 systems,
>   the mutex definition for x86-64 is incompatible with the mutex definition for i386, meaning that 32-bit and
>   64-bit binaries can't share mutexes on x86-64 systems.

SEE ALSO
>   **credentials**(7), **pthreads**(7), **signal**(7), **standards**(7)