NAME

p_candebug - determine debuggability of a process

SYNOPSIS

#include <sys/param.h>
#include <sys/proc.h>

int
p_candebug(*struct thread *td*, *struct proc *p*);

DESCRIPTION

This function determines if a given process *p* is debuggable by some thread *td*.

The following sysctl(8) variables directly influence the behaviour of **p_candebug**():

security.bsd.unprivileged_proc_debug
: Must be set to a non-zero value to allow unprivileged processes access to the kernel's debug facilities.

kern.securelevel
: Debugging of the init process is not allowed if this variable is 1 or greater.

Other such variables indirectly influence it; see cr_bsd_visible(9).

RETURN VALUES

The **p_candebug**() function returns 0 if the process denoted by *p* is debuggable by thread *td*, or a non-zero error return value otherwise.

ERRORS

[EPERM] An unprivileged process attempted to debug another process but the system is configured to deny it (see sysctl(8) variable *security.bsd.unprivileged_proc_debug* above).

[ESRCH] Thread *td* has been jailed and the process to debug does not belong to the same jail or one of its sub-jails, as determined by prison_check(9).

[ESRCH] cr_bsd_visible(9) denied visibility according to the BSD security policies in force.

[EPERM] Thread *td* lacks superuser credentials and its (effective) group set is not a superset of process *p*'s whole group set (including real, effective and saved group IDs).

[EPERM] Thread *td* lacks superuser credentials and its (effective) user ID does not match all user IDs of process *p*.

[EPERM] Thread *td* lacks superuser credentials and process *p* is executing a set-user-ID or set-group-ID executable.

[EPERM] Process *p* denotes the initial process **initproc**() and the sysctl(8) variable *kern.securelevel* is greater than zero.

[EBUSY] Process *p* is in the process of being **exec**()'ed.

[EPERM] Process *p* denied debuggability (see procctl(2), command PROC_TRACE_CTL).

SEE ALSO

prison_check(9), mac(9), cr_bsd_visible(9), procctl(2), p_cansee(9)