NAME

arch_prctl − set architecture-specific thread state

LIBRARY

Standard C library (*libc*, *−lc*)

SYNOPSIS

#include <asm/prctl.h> /* Definition of **ARCH_*** constants */
#include <sys/syscall.h> /* Definition of **SYS_*** constants */
#include <unistd.h>

int syscall(SYS_arch_prctl, int *code***, unsigned long** *addr***);**
int syscall(SYS_arch_prctl, int *code***, unsigned long *****addr***);**

Note: glibc provides no wrapper for **arch_prctl**(), necessitating the use of **syscall**(2).

DESCRIPTION

arch_prctl() sets architecture-specific process or thread state. *code* selects a subfunction and passes ar gument *addr* to it; *addr* is interpreted as either an *unsigned long* for the "set" operations, or as an *unsigned long* *, for the "get" operations.

Subfunctions for both x86 and x86-64 are:

ARCH_SET_CPUID (since Linux 4.12)
Enable (*addr != 0*) or disable (*addr == 0*) the *cpuid* instruction for the calling thread. The instruction is enabled by default. If disabled, any execution of a *cpuid* instruction will instead generate a **SIGSEGV** signal. This feature can be used to emulate *cpuid* results that differ from what the underlying hardware would have produced (e.g., in a paravirtualization setting).

The **ARCH_SET_CPUID** setting is preserved across **fork**(2) and **clone**(2) but reset to the default (i.e., *cpuid* enabled) on **execve**(2).

ARCH_GET_CPUID (since Linux 4.12)
Return the setting of the flag manipulated by **ARCH_SET_CPUID** as the result of the system call (1 for enabled, 0 for disabled). *addr* is ignored.

Subfunctions for x86-64 only are:

ARCH_SET_FS
Set the 64-bit base for the *FS* register to *addr*.

ARCH_GET_FS
Return the 64-bit base value for the *FS* register of the calling thread in the *unsigned long* pointed to by *addr*.

ARCH_SET_GS
Set the 64-bit base for the*GS* re gister to *addr*.

ARCH_GET_GS
Return the 64-bit base value for the*GS* re gister of the calling thread in the *unsigned long* pointed to by *addr*.

RETURN VALUE

On success, **arch_prctl**() returns 0; on error, −1 is returned, and *errno* is set to indicate the error.

ERRORS

EFAULT
addr points to an unmapped address or is outside the process address space.

EINVAL
code is not a valid subcommand.

ENODEV
ARCH_SET_CPUID was requested, but the underlying hardware does not support CPUID faulting.

EPERM
addr is outside the process address space.

STANDARDS

arch_prctl() is a Linux/x86-64 extension and should not be used in programs intended to be portable.

NOTES

arch_prctl() is supported only on Linux/x86-64 for 64-bit programs currently.

The 64-bit base changes when a new 32-bit segment selector is loaded.

ARCH_SET_GS is disabled in some kernels.

Context switches for 64-bit segment bases are rather expensive. As an optimization, if a 32-bit TLS base address is used, **arch_prctl**() may use a real TLS entry as if **set_thread_area**(2) had been called, instead of manipulating the segment base register directly. Memory in the first 2 GB of address space can be allocated by using **mmap**(2) with the **MAP_32BIT** flag.

Because of the aforementioned optimization, using **arch_prctl**() and **set_thread_area**(2) in the same thread is dangerous, as they may overwrite each other's TLS entries.

FS may be already used by the threading library. Programs that use**ARCH_SET_FS** directly are v ery likely to crash.

SEE ALSO

mmap(2), **modify_ldt**(2), **prctl**(2), **set_thread_area**(2)

AMD X86-64 Programmer's manual