NAME

subpage_prot − define a subpage protection for an address range

SYNOPSIS

long subpage_prot(unsigned long *addr***, unsigned long** *len***,**
uint32_t ****map*);**

Note: There is no glibc wrapper for this system call; see NOTES.

DESCRIPTION

The PowerPC-specific **subpage_prot**() system call provides the facility to control the access permissions on individual 4 kB subpages on systems configured with a page size of 64 kB.

The protection map is applied to the memory pages in the region starting at *addr* and continuing for *len* bytes. Both of these arguments must be aligned to a 64-kB boundary.

The protection map is specified in the buffer pointed to by *map*. The map has 2 bits per 4 kB subpage; thus each 32-bit word specifies the protections of 16 4 kB subpages inside a 64 kB page (so, the number of 32-bit words pointed to by *map* should equate to the number of 64-kB pages specified by *len*). Each 2-bit field in the protection map is either 0 to allow any access, 1 to prevent writes, or 2 or 3 to prevent all accesses.

RETURN VALUE

On success, **subpage_prot**() returns 0. Otherwise, one of the error codes specified below is returned.

ERRORS

EFAULT
The buffer referred to by *map* is not accessible.

EINVAL
The *addr* or *len* arguments are incorrect. Both of these arguments must be aligned to a multiple of the system page size, and they must not refer to a region outside of the address space of the process or to a region that consists of huge pages.

ENOMEM
Out of memory.

VERSIONS

This system call is provided on the PowerPC architecture since Linux 2.6.25. The system call is provided only if the kernel is configured with **CONFIG_PPC_64K_PAGES**. No library support is provided.

CONFORMING TO

This system call is Linux-specific.

NOTES

Glibc does not provide a wrapper for this system call; call it using **syscall**(2).

Normal page protections (at the 64-kB page level) also apply; the subpage protection mechanism is an additional constraint, so putting 0 in a 2-bit field won't allow writes to a page that is otherwise write-protected.

Rationale

This system call is provided to assist writing emulators that operate using 64-kB pages on PowerPC systems. When emulating systems such as x86, which uses a smaller page size, the emulator can no longer use the memory-management unit (MMU) and normal system calls for controlling page protections. (The emulator could emulate the MMU by checking and possibly remapping the address for each memory access in software, but that is slow.) The idea is that the emulator supplies an array of protection masks to apply to a specified range of virtual addresses. These masks are applied at the level where hardware page-table entries (PTEs) are inserted into the hardware page table based on the Linux PTEs, so the Linux PTEs are not affected. Implicit in this is that the regions of the address space that are protected are switched to use 4-kB hardware pages rather than 64-kB hardware pages (on machines with hardware 64-kB page support).

SEE ALSO

mprotect(2), **syscall**(2)

Documentation/admin-guide/mm/hugetlbpage.rst in the Linux kernel source tree

COLOPHON

This page is part of release 5.05 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/.