NAME

migrate_pages − move all pages in a process to another set of nodes

SYNOPSIS

#include <numaif.h>

long migrate_pages(int *pid***, unsigned long** *maxnode,*
**const unsigned long ****old_nodes,*
const unsigned long ****new_nodes*);**

Link with *−lnuma*.

DESCRIPTION

migrate_pages() attempts to move all pages of the process *pid* that are in memory nodes *old_nodes* to the memory nodes in *new_nodes*. Pages not located in any node in *old_nodes* will not be migrated. As far as possible, the kernel maintains the relative topology relationship inside *old_nodes* during the migration to *new_nodes*.

The *old_nodes* and *new_nodes* arguments are pointers to bit masks of node numbers, with up to *maxnode* bits in each mask. These masks are maintained as arrays of unsigned *long* integers (in the last *long* integer, the bits beyond those specified by *maxnode* are ignored). The *maxnode* argument is the maximum node number in the bit mask plus one (this is the same as in **mbind**(2), but different from **select**(2)).

The *pid* argument is the ID of the process whose pages are to be moved. To move pages in another process, the caller must be privileged (**CAP_SYS_NICE**) or the real or effective user ID of the calling process must match the real or saved-set user ID of the target process. If *pid* is 0, then **migrate_pages**() moves pages of the calling process.

Pages shared with another process will be moved only if the initiating process has the **CAP_SYS_NICE** privilege.

RETURN VALUE

On success **migrate_pages**() returns the number of pages that could not be moved (i.e., a return of zero means that all pages were successfully moved). On error, it returns −1, and sets *errno* to indicate the error.

ERRORS

EFAULT

Part or all of the memory range specified by *old_nodes*/*new_nodes* and *maxnode* points outside your accessible address space.

EINVAL

The value specified by *maxnode* exceeds a kernel-imposed limit. Or, *old_nodes* or *new_nodes* specifies one or more node IDs that are greater than the maximum supported node ID. Or, none of the node IDs specified by *new_nodes* are on-line and allowed by the process's current cpuset context, or none of the specified nodes contain memory.

EPERM

Insufficient privilege (**CAP_SYS_NICE**) to move pages of the process specified by *pid*, or insufficient privilege (**CAP_SYS_NICE**) to access the specified target nodes.

ESRCH

No process matching *pid* could be found.

VERSIONS

The **migrate_pages**() system call first appeared on Linux in version 2.6.16.

CONFORMING TO

This system call is Linux-specific.

NOTES

For information on library support, see **numa**(7).

Use **get_mempolicy**(2) with the **MPOL_F_MEMS_ALLOWED** flag to obtain the set of nodes that are allowed by the calling process's cpuset. Note that this information is subject to change at any time by manual

or automatic reconfiguration of the cpuset.

Use of **migrate_pages**() may result in pages whose location (node) violates the memory policy established for the specified addresses (see **mbind**(2)) and/or the specified process (see **set_mempolicy**(2)). That is, memory policy does not constrain the destination nodes used by **migrate_pages**().

The *<numaif.h>* header is not included with glibc, but requires installing *libnuma-devel* or a similar package.

SEE ALSO

get_mempolicy(2), **mbind**(2), **set_mempolicy**(2), **numa**(3), **numa_maps**(5), **cpuset**(7), **numa**(7), **migratepages**(8), **numastat**(8)

Documentation/vm/page_migration.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/.