NAME

move_pages – move individual pages of a process to another node

LIBRARY

NUMA (Non-Uniform Memory Access) policy library (*libnuma*, *−lnuma*)

SYNOPSIS

#include <numaif.h>

long move_pages(int *pid***, unsigned long** *count***, void ****pages***[.***count***],**
    const int *nodes***[.***count***], int** *status***[.***count***], int** *flags***);**

DESCRIPTION

move_pages() moves the specified *pages* of the process *pid* to the memory nodes specified by *nodes*. The result of the move is reflected in *status*. The *flags* indicate constraints on the pages to be moved.

pid is the ID of the process in which pages are to be moved. If *pid* is 0, then **move_pages**() moves pages of the calling process.

To move pages in another process requires the following privileges:

• Up to and including Linux 4.12: 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.

• The older rules allowed the caller to discover various virtual address choices made by the kernel that could lead to the defeat of address-space-layout randomization for a process owned by the same UID as the caller, the rules were changed starting with Linux 4.13. Since Linux 4.13, permission is governed by a ptrace access mode **PTRACE_MODE_READ_REALCREDS** check with respect to the target process; see **ptrace**(2).

count is the number of pages to move. It defines the size of the three arrays *pages*, *nodes*, and *status*.

pages is an array of pointers to the pages that should be moved. These are pointers that should be aligned to page boundaries. Addresses are specified as seen by the process specified by *pid*.

nodes is an array of integers that specify the desired location for each page. Each element in the array is a node number. *nodes* can also be NULL, in which case **move_pages**() does not move any pages but instead will return the node where each page currently resides, in the *status* array. Obtaining the status of each page may be necessary to determine pages that need to be moved.

status is an array of integers that return the status of each page. The array contains valid values only if **move_pages**() did not return an error. Preinitialization of the array to a value which cannot represent a real numa node or valid error of status array could help to identify pages that have been migrated.

flags specify what types of pages to move. **MPOL_MF_MOVE** means that only pages that are in exclusive use by the process are to be moved. **MPOL_MF_MOVE_ALL** means that pages shared between multiple processes can also be moved. The process must be privileged (**CAP_SYS_NICE**) to use **MPOL_MF_MOVE_ALL**.

Page states in the status array

The following values can be returned in each element of the *status* array.

0..MAX_NUMNODES

Identifies the node on which the page resides.

−EACCES

The page is mapped by multiple processes and can be moved only if **MPOL_MF_MOVE_ALL** is specified.

−EBUSY

The page is currently busy and cannot be moved. Try again later. This occurs if a page is undergoing I/O or another kernel subsystem is holding a reference to the page.

−EFAULT

This is a zero page or the memory area is not mapped by the process.

−EIO   Unable to write back a page. The page has to be written back in order to move it since the page is dirty and the filesystem does not provide a migration function that would allow the move of dirty pages.

−EINVAL

A dirty page cannot be moved. The filesystem does not provide a migration function and has no ability to write back pages.

−ENOENT

The page is not present.

−ENOMEM

Unable to allocate memory on target node.

RETURN VALUE

On success **move_pages**() returns zero. On error, it returns −1, and sets *errno* to indicate the error. If positive value is returned, it is the number of nonmigrated pages.

ERRORS

Positive value

The number of nonmigrated pages if they were the result of nonfatal reasons (since Linux 4.17).

E2BIG   Too many pages to move. Since Linux 2.6.29, the kernel no longer generates this error.

EACCES

One of the target nodes is not allowed by the current cpuset.

EFAULT

Parameter array could not be accessed.

EINVAL

Flags other than **MPOL_MF_MOVE** and **MPOL_MF_MOVE_ALL** was specified or an attempt was made to migrate pages of a kernel thread.

ENODEV

One of the target nodes is not online.

EPERM

The caller specified **MPOL_MF_MOVE_ALL** without sufficient privileges (**CAP_SYS_NICE**). Or, the caller attempted to move pages of a process belonging to another user but did not have privilege to do so (**CAP_SYS_NICE**).

ESRCH

Process does not exist.

VERSIONS

move_pages() first appeared in Linux 2.6.18.

STANDARDS

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 current cpuset. Note that this information is subject to change at any time by manual or automatic reconfiguration of the cpuset.

Use of this function 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 **move_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)