

**BAHIR DAR UNIVERSITY**

**Faculty of computing**

**Department of software engineering**

**Operating system and system programming**

**second assignment**

**system call**

**Name: Salem Solomon**

**ID:1306271**

1 what is long move\_page system call

**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.

How it work?

To move pages in another process requires the following

privileges:

\* In kernels 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**

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.

why this system call used?

move individual pages of a process to another node

2,briefly describe about the list of parameter and flag.

*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. Pre initialization 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.

3, list the flags, their purpose with code implementation .

*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.

|  |
| --- |
| if (use\_non\_thp\_migration || use\_thp\_migration) |
|  |

|  |
| --- |
| mm\_manage\_flags |= MPOL\_MF\_MOVE; |

**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**.

|  |
| --- |
| if(move\_hot\_and\_cold\_pages) |
| mm\_manage\_flags |= MPOL\_MF\_MOVE\_ALL; |
|  |

|  |
| --- |
| mm\_manage\_flags |= MPOL\_MF\_MOVE\_ALL; |
|  |

MPOL\_MF\_MOVE

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  | | --- | | if (use\_non\_thp\_migration || use\_thp\_migration) | |  |  |  | | --- | | mm\_manage\_flags |= MPOL\_MF\_MOVE; | |  |  |  | | --- | | if (use\_concur\_migration) | |  |  |  | | --- | | mm\_manage\_flags |= MPOL\_MF\_MOVE|MPOL\_MF\_MOVE\_CONCUR; | |  |  |  | | --- | | if (use\_opt\_migration) | |  |  |  | | --- | | mm\_manage\_flags |= MPOL\_MF\_MOVE|MPOL\_MF\_MOVE\_MT|MPOL\_MF\_MOVE\_CONCUR; | |  |  |  | | --- | | if (use\_basic\_exchange\_pages) | |  |  |  | | --- | | mm\_manage\_flags |= MPOL\_MF\_MOVE|MPOL\_MF\_EXCHANGE; | |  |  |  | | --- | | if (use\_concur\_only\_exchange\_pages) | |  |  |  | | --- | | mm\_manage\_flags |= MPOL\_MF\_MOVE|MPOL\_MF\_MOVE\_CONCUR|MPOL\_MF\_EXCHANGE; | |  |  |  | | --- | | if (use\_exchange\_pages) | |  |  |  | | --- | | mm\_manage\_flags |= MPOL\_MF\_MOVE|MPOL\_MF\_MOVE\_MT|MPOL\_MF\_MOVE\_CONCUR|MPOL\_MF\_EXCHANGE; | |  |  |  | | --- | |  | |  |  |  | | --- | | if (shrink\_page\_lists) | |  |  |  | | --- | | mm\_manage\_flags |= MPOL\_MF\_SHRINK\_LISTS; | |  |  |  | | --- | | if (move\_hot\_and\_cold\_pages) | |  |  |  | | --- | | mm\_manage\_flags |= MPOL\_MF\_MOVE\_ALL; | |  |  |  | | --- | | if (no\_migration) | |  |  |  | | --- | | mm\_manage\_flags &= ~MPOL\_MF\_MOVE; | |  |  |  | | --- | |  | |  |  |  | | --- | | /\* cpu\_mask overwrites node\_mask \*/ | |  |  |  | | --- | | if (cpu\_mask) | |  |  |  | | --- | | { | |  |  |  | | --- | | numa\_bitmask\_free(node\_mask); | |  |  |  | | --- | | node\_mask = NULL; | |  | |

|  |
| --- |
|  |

