Full credit is given to the above companies including the OS that this PDF file was generated!

Red Hat Enterprise Linux Release 9.2 Manual Pages on 'mlockall.3p' command

$ man mlockall.3p

MLOCKALL(3P) POSIX Programmer's Manual MLOCKALL(3P)

PROLOG

This manual page is part of the POSIX Programmer's Manual. The Linux implementation of this interface may differ (consult the corresponding Linux manual page for details of Linux behavior), or the interface may not be implemented on Linux.

NAME

mlockall, munlockall ? lock/unlock the address space of a process (RE? ALTIME)

SYNOPSIS

```
#include <sys/mman.h>
int mlockall(int flags);
int munlockall(void);
```

DESCRIPTION

The mlockall() function shall cause all of the pages mapped by the ad? dress space of a process to be memory-resident until unlocked or until the process exits or execs another process image. The flags argument determines whether the pages to be locked are those currently mapped by the address space of the process, those that are mapped in the future, or both. The flags argument is constructed from the bitwise-inclusive OR of one or more of the following symbolic constants, defined in <sys/mman.h>:

MCL_CURRENT Lock all of the pages currently mapped into the address

space of the process.

MCL_FUTURE Lock all of the pages that become mapped into the address space of the process in the future, when those mappings are established.

If MCL_FUTURE is specified, and the automatic locking of future mappings eventually causes the amount of locked memory to exceed the amount of available physical memory or any other implementation-defined limit, the behavior is implementation-defined. The manner in which the implementation informs the application of these situations is also implementation-defined.

The munlockall() function shall unlock all currently mapped pages of the address space of the process. Any pages that become mapped into the address space of the process after a call to munlockall() shall not be locked, unless there is an intervening call to mlockall() specifying MCL_FUTURE or a subsequent call to mlockall() specifying MCL_CURRENT.

If pages mapped into the address space of the process are also mapped into the address spaces of other processes and are locked by those processes, the locks established by the other processes shall be unaffected by a call by this process to munlockall().

Upon successful return from the mlockall() function that specifies MCL_CURRENT, all currently mapped pages of the address space of the process shall be memory-resident and locked. Upon return from the munlockall() function, all currently mapped pages of the address space of the process shall be unlocked with respect to the address space of the process. The memory residency of unlocked pages is unspecified.

Appropriate privileges are required to lock process memory with mlockall().

RETURN VALUE

Upon successful completion, the mlockall() function shall return a value of zero. Otherwise, no additional memory shall be locked, and the function shall return a value of -1 and set errno to indicate the error. The effect of failure of mlockall() on previously existing locks in the address space is unspecified.

If it is supported by the implementation, the munlockall() function shall always return a value of zero. Otherwise, the function shall re? turn a value of -1 and set errno to indicate the error.

ERRORS

The mlockall() function shall fail if:

EAGAIN Some or all of the memory identified by the operation could not be locked when the call was made.

EINVAL The flags argument is zero, or includes unimplemented flags.

The mlockall() function may fail if:

ENOMEM Locking all of the pages currently mapped into the address space of the process would exceed an implementation-defined limit on the amount of memory that the process may lock.

EPERM The calling process does not have appropriate privileges to per? form the requested operation.

The following sections are informative.

EXAMPLES

None.

APPLICATION USAGE

None.

RATIONALE

None.

FUTURE DIRECTIONS

None.

SEE ALSO

exec, exit(), fork(), mlock(), munmap()

The Base Definitions volume of POSIX.1?2017, <sys_mman.h>

COPYRIGHT

The Open Group Standard, the original IEEE and The Open Group Standard is the referee document. The original Standard can be obtained online at http://www.opengroup.org/unix/online.html .

Any typographical or formatting errors that appear in this page are most likely to have been introduced during the conversion of the source files to man page format. To report such errors, see https://www.ker?nel.org/doc/man-pages/reporting_bugs.html .