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 'pthread_attr_getstack.3p' command

$ man pthread_attr_getstack.3p

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

pthread_attr_getstack, pthread_attr_setstack ? get and set stack at? tributes

SYNOPSIS

```
#include <pthread.h>
int pthread_attr_getstack(const pthread_attr_t *restrict attr,
    void **restrict stackaddr, size_t *restrict stacksize);
int pthread_attr_setstack(pthread_attr_t *attr, void *stackaddr,
    size_t stacksize);
```

DESCRIPTION

The pthread_attr_getstack() and pthread_attr_setstack() functions, re? spectively, shall get and set the thread creation stack attributes stackaddr and stacksize in the attr object.

The stack attributes specify the area of storage to be used for the created thread's stack. The base (lowest addressable byte) of the stor? age shall be stackaddr, and the size of the storage shall be stacksize bytes. The stacksize shall be at least {PTHREAD_STACK_MIN}. The

pthread_attr_setstack() function may fail with [EINVAL] if stackaddr does not meet implementation-defined alignment requirements. All pages within the stack described by stackaddr and stacksize shall be both readable and writable by the thread.

If the pthread_attr_getstack() function is called before the stackaddr attribute has been set, the behavior is unspecified.

The behavior is undefined if the value specified by the attr argument to pthread_attr_getstack() or pthread_attr_setstack() does not refer to an initialized thread attributes object.

RETURN VALUE

Upon successful completion, these functions shall return a value of 0; otherwise, an error number shall be returned to indicate the error.

The pthread_attr_getstack() function shall store the stack attribute values in stackaddr and stacksize if successful.

ERRORS

The pthread_attr_setstack() function shall fail if:

EINVAL The value of stacksize is less than {PTHREAD_STACK_MIN} or ex?ceeds an implementation-defined limit.

The pthread_attr_setstack() function may fail if:

EINVAL The value of stackaddr does not have proper alignment to be used as a stack, or ((char *)stackaddr + stacksize) lacks proper alignment.

EACCES The stack page(s) described by stackaddr and stacksize are not both readable and writable by the thread.

These functions shall not return an error code of [EINTR].

The following sections are informative.

EXAMPLES

None.

APPLICATION USAGE

These functions are appropriate for use by applications in an environ?ment where the stack for a thread must be placed in some particular re?gion of memory.

While it might seem that an application could detect stack overflow by

providing a protected page outside the specified stack region, this cannot be done portably. Implementations are free to place the thread's initial stack pointer anywhere within the specified region to accommo? date the machine's stack pointer behavior and allocation requirements. Furthermore, on some architectures, such as the IA?64, ``overflow'' might mean that two separate stack pointers allocated within the region will overlap somewhere in the middle of the region.

After a successful call to pthread_attr_setstack(), the storage area specified by the stackaddr parameter is under the control of the imple? mentation, as described in Section 2.9.8, Use of Application-Managed Thread Stacks.

The specification of the stackaddr attribute presents several ambigui? ties that make portable use of these functions impossible. For example, the standard allows implementations to impose arbitrary alignment re? quirements on stackaddr. Applications cannot assume that a buffer ob? tained from malloc() is suitably aligned. Note that although the stack? size value passed to pthread_attr_setstack() must satisfy alignment re? quirements, the same is not true for pthread_attr_setstacksize() where the implementation must increase the specified size if necessary to achieve the proper alignment.

RATIONALE

If an implementation detects that the value specified by the attr argu? ment to pthread_attr_getstack() or pthread_attr_setstack() does not re? fer to an initialized thread attributes object, it is recommended that the function should fail and report an [EINVAL] error.

FUTURE DIRECTIONS

None.

SEE ALSO

pthread_attr_destroy(), pthread_attr_getdetachstate(), pthread_attr_getstacksize(), pthread_create()

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