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 'sem_trywait.3p' command

$ man sem_trywait.3p

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

sem_trywait, sem_wait ? lock a semaphore

SYNOPSIS

```
#include <semaphore.h>
int sem_trywait(sem_t *sem);
int sem_wait(sem_t *sem);
```

DESCRIPTION

The sem_trywait() function shall lock the semaphore referenced by sem only if the semaphore is currently not locked; that is, if the sema? phore value is currently positive. Otherwise, it shall not lock the semaphore.

The sem_wait() function shall lock the semaphore referenced by sem by performing a semaphore lock operation on that semaphore. If the sema? phore value is currently zero, then the calling thread shall not return from the call to sem_wait() until it either locks the semaphore or the call is interrupted by a signal.

Upon successful return, the state of the semaphore shall be locked and

shall remain locked until the sem_post() function is executed and re?
turns successfully.

The sem_wait() function is interruptible by the delivery of a signal.

RETURN VALUE

The sem_trywait() and sem_wait() functions shall return zero if the calling process successfully performed the semaphore lock operation on the semaphore designated by sem. If the call was unsuccessful, the state of the semaphore shall be unchanged, and the function shall re?
turn a value of -1 and set errno to indicate the error.

ERRORS

The sem_trywait() function shall fail if:

EAGAIN The semaphore was already locked, so it cannot be immediately locked by the sem_trywait() operation.

The sem_trywait() and sem_wait() functions may fail if:

EDEADLK

A deadlock condition was detected.

EINTR A signal interrupted this function.

EINVAL The sem argument does not refer to a valid semaphore.

The following sections are informative.

EXAMPLES

None.

APPLICATION USAGE

Applications using these functions may be subject to priority inver?
sion, as discussed in the Base Definitions volume of POSIX.1?2017, Sec?
tion 3.291, Priority Inversion.

RATIONALE

None.

FUTURE DIRECTIONS

None.

SEE ALSO

semctl(), semget(), semop(), sem_post(), sem_timedwait()

The Base Definitions volume of POSIX.1?2017, Section 3.291, Priority Inversion, Section 4.12, Memory Synchronization, <semaphore.h>