فعالیت کلاسی سیستم عامل

سارا سادات یونسی / 98533053

Question : Is it possible to share mutex between multiple processes?

Mutual exclusion locks (mutexes) prevent multiple threads from simultaneously executing critical sections of code that access shared data (that is, mutexes are used to serialize the execution of threads). All mutexes must be global. A successful call for a mutex lock by way of mutex_lock() will cause another thread that is also trying to lock the same mutex to block until the owner thread unlocks it by way of mutex_unlock(). Threads within the same process or within other processes can share mutexes.

Mutexes can synchronize threads within the same process or in *other processes*. Mutexes can be used to synchronize threads between processes if the mutexes are allocated in writable memory and shared among the cooperating processes (see mmap(2)), and have been initialized for this task.

Mutexes are either intra-process or inter-process, depending upon the argument passed implicitly or explicitly to the initialization of that mutex. A statically allocated mutex does not need to be explicitly initialized; by default, a statically allocated mutex is initialized with all zeros and its scope is set to be within the calling process.

For inter-process synchronization, a mutex needs to be allo- cated in memory shared between these processes. Since the memory for such a mutex must be allocated dynamically, the mutex needs to be explicitly initialized using mutex_init().

Source:

multithreading - Is it possible to use mutex in multiprocessing case on Linux/UNIX ? - Stack Overflow