Behold the levels of lock!#

Grasshopper - this is the entry level. Memory is allocated, but you are not yet worthy of pagetable entries. Upon first attempt you are doomed to failure, and must sit in **STATE_WAITPAGE** while the master process decides your fate. Failure to initialize the page will result in your immediate receipt of the dreaded portent of public transport doom - **SIGBUS**
Cricket - congratulations, you have read the documentation! You may now lock all or portions of your address space.
- HOWEVER -
the master is not yet satisfied you quite know what you are doing. ***INSOLENT DOG!***
Pages you THINK are **PROT_WRITE** will still actually be **PROT_READ**. This is so that on first write the kernel may be alerted that a **MAP_PRIVATE** page now is different from the shared backing store, and must be privatised.
Supreme Grand Master - you are the superuser, and you must have complete control. More specifically you may temporarily pause the passage of time (via **InterruptDisable()**) and hence the Pagefault Djinns are powerless to touch your pages. Thus the lazy Djinns must be flogged beforehand and enjoined to privatize, set permissions and otherwise initialize all memories, such that *NO FAULTS* will cause the fabric of space-time to unravel.
This, my friends, is the ***SUPERLOCK!***
For the weary, you can relax and simply set -ml to set cricket status for your application - thus all pages are at least initalized (if still only set to **PROT_READ**).
If you need no page faulting whatsoever, then -mL is for you. Here you pay your costs at **mmap()** time. No further taxation will be demanded.
As usual, **MAP_LAZY** will always pagefault, and the dreaded **SIGBUS** may run you over, should you empty the pool of limited memory resource.
Choose wisely, my friends....