How to convince my co-worker the linux kernel code is re-entrant?

Yeah I know ... Some people are sometimes hard to convince of what sounds natural to the rest of us, an I need your help right now SO community (or I'll go postal soon ..)

One of my co-worker is convinced the linux kernel code is **not** re-entrant as he reads it somewhere last time he get insterested in it, likely 7 years ago. Probably its reading was right at that time, remember that multi core architecture was not much widespread some time ago and linux project at its begining or so was not totally well writen and fully fledged with all fancy features.

Today is different. It's obvious that calling the same system call from different processes running in parallel on the same architecture won't lead to undefined behavior. Linux kernel is widespread now, and known for its reability even though running on multicore architectures. That is my argument for now. But what would be yours to **prove that objectively** ?

I was thinking to show him off some function in the linux kernel (on lxr website) as the mutex_lock() system call. Eveything is tuned to get it work in concurrent environnement. But the code could be not that obvious for newbie (as I am).

Please help me.. ;-)

linux multithreading linux-kernel reentrancy

edited Oct 12 '09 at 23:12 asked Oct 12 '09 at 23:03
yves Baumes
4,757 1 22 50

4 Reminds me of this: xkcd.com/386 Seriously, why expend energy on fools? – Paul Abbott Oct 12 '09 at 23:13

1 Because I am insterested in knowing other's point of view, and discussion brings more knowledge to everyone, rather still & ever guessing what is true – yves Baumes Oct 13 '09 at 7:43

4 Answers

Search the kernel mailing list archive for "BKL". That stands for "Big Kernel Lock", which is what used to be used to prevent problems. A lot of work has been put into breaking it up into pieces, to allow reentry as long different parts of the kernel are used by different processes. Most recent mentions of "BKL" (at least that I've noticed) have basically referred to somebody trying to make his own life easy by locking more than somebody else approved of, at which point they frequently say something about "returning to the days of the BKL", or something on that order.

answered Oct 12 '09 at 23:12
Jerry Coffin
257k 25 256 567

The easiest way to prove that multiple CPUs can execute in the kernel simultaneously would be to write a program that does a lot of work in-kernel (for example, looks up long pathnames in a tight loop), then run two copies of it at the same time on a dual-core machine and show that the "system" percentage in `top` goes above 50%.

answered Oct 13 '09 at 0:01
caf
122k 8 139 269

+1 for proposing a real test!!! – mikera Oct 12 '10 at 22:48

At the risk of being snarky: why not just read the code? If neither of you are expert enough to follow the code through an interrupt handler and into some subsystem or another where you can read out the synchronization code, then ... why bother? Isn't this just a dancing on the head of a pin argument? It's like a creationist demanding "proof" of evolution when they aren't interested in learning any biology.

answered Oct 13 '09 at 2:39
Andy Ross
7,439 13 20

Maybe you should have your friend prove Linux is *not* reentrant. Burden should not be on you to prove this.

answered Oct 28 '09 at 20:53
Andy Grover
393 1 5