

## Re: msleep() on recursively locked mutexes

---

*Source:* <http://unix.derkeiler.com/Mailing-Lists/FreeBSD/hackers/2007-04/msg00212.html>

---

- *From:* Julian Elischer <[julian@xxxxxxxxxxxxx](mailto:julian@xxxxxxxxxxxxx)>
  - *Date:* Thu, 26 Apr 2007 14:18:00 -0700
- 

Hans Petter Selasky wrote:

Hi,

In the new USB stack I have defined the following:

```
u_int32_t
mtx_drop_recurse(struct mtx *mtx)
{
    u_int32_t recurse_level = mtx->mtx_recurse;
    u_int32_t recurse_curr = recurse_level;

    mtx_assert(mtx, MA_OWNED);

    while(recurse_curr-->0) {
        mtx_unlock(mtx);
    }

    return recurse_level;
}
```

The reason that mutexes ever recurse in the first place is usually because one piece of code calls itself (or a related piece of code) in a blind manner.. in other words, it doesn't know it is doing so. The whole concept of recursing mutexes is a bit gross. The concept of blindly unwinding them is I think just asking for trouble.

Further the idea that holding a mutex "except for when we sleep" is a generally bright idea is also a bit odd to me.. If you hold a mutex and release it during sleep you probably should invalidate all assumptions you made during the period before you slept as whatever you were protecting has possibly been raped while you slept. I have seen too many instances where people just called msleep and dropped the mutex they held, picked it up again on wakeup, and then blithely continued on without checking what happened while they were asleep.

It seems to me that if someone else held that lock for some purpose when you slept, then you don't know what it was that they were trying to protect, so you don't know what to recheck when you wake up.. I think this is extremely dangerous as you don't know when you are in this situation..

## Re: msleep() on recursively locked mutexes

Personally I think Matt Dillon had a good idea when he suggested that the need to sleep when holding a lock should require the lock holder to back out as far as needed as to be able to see why the lock was held and to comfortably cope with the situation when the protected structures got frobbed while it slept. (He actually at one time committed some primitives to do this. I forget their names and they were never used, but the idea has some merit.)

This change may be OK in the situations you plan but it makes me very uncomfortable. It basically should have a big sign on it saying "You could spend years looking for the wierd bugs you are likely to get if you use this" on it.

```
void
mtx_pickup_recurse(struct mtx *mtx, u_int32_t recurse_level)
{
    mtx_assert(mtx, MA_OWNED);

    while(recurse_level-- > 0) {
        mtx_lock(mtx);
    }
    return;
}
```

When I do a msleep() I do it like this:

```
level = mtx_drop_recurse(ctd->p_mtx);

error = msleep(ctd, ctd->p_mtx, 0, "config td sleep", timeout);

mtx_pickup_recurse(ctd->p_mtx, level);
```

Are there any comments on integrating this functionality into msleep(), and adding mtx\_drop\_recurse() and mtx\_pickup\_recurse() to the FreeBSD kernel?

--HPS

---

freebsd-hackers@xxxxxxxxxxx mailing list  
<http://lists.freebsd.org/mailman/listinfo/freebsd-hackers>  
To unsubscribe, send any mail to "freebsd-hackers-unsubscribe@xxxxxxxxxxx"

---

freebsd-hackers@xxxxxxxxxxx mailing list  
<http://lists.freebsd.org/mailman/listinfo/freebsd-hackers>  
To unsubscribe, send any mail to "freebsd-hackers-unsubscribe@xxxxxxxxxxx"