

stable kqueue locking up and running on SMP

Source: <http://unix.derkeiler.com/Mailing-Lists/FreeBSD/arch/2004-04/0075.html>

From: Brian Fundakowski Feldman (*green_at_FreeBSD.org*)

Date: 04/21/04

To: arch@FreeBSD.org

Date: Wed, 21 Apr 2004 00:39:52 -0400

Please test out and provide feedback for the latest iteration:

<<http://green.homeunix.org/~green/kqueue-giant-locking.2.patch>>

Things are now working in exactly the way I described I was going to implement them -- there is one kqueue lock, and if, as I suspect, it will `_not_` be a point of contention, there is no reason at all to complicate things further.

To do finer-grained locking, klists will have to be overhauled to become an actual, protected, object instead of hanging along with the object they represent. More complication in the form of read-locking (or holds) will also be necessary for that. For now, though, they belong to `_kqueue_`, and not `struct selinfo` or the object that contains them normally. A more complicated scheme for kqueues, knots, klist(-like)s would benefit greatly from divorce of kqueue from `struct filedesc`, if anyone is ever to tackle that (if there is ever a return on investment; `MUTEX_PROFILING` should help us figure that out.)

The only known issue on my SMP box is this warning at boot time; I am unable to track down why `witness(4)` believes there to be a lock order reversal, but I welcome someone else to help identify this one:

lock order reversal

1st 0xc0661500 global kqueue lock (global kqueue lock) @ kern/kern_event.c:413

2nd 0xc45bc438 filedesc structure (filedesc structure) @ kern/kern_event.c:422

Stack backtrace:

backtrace(c0616b80,c45bc438,c060fd6e,c060fd6e,c0610941) at backtrace+0x17

witness_checkorder(c45bc438,9,c0610941,1a6,c455d594) at witness_checkorder+0x6f6

_mtx_lock_flags(c45bc438,0,c0610938,1a6,c455d594) at _mtx_lock_flags+0x9a

kqueue(c45b73f0,db138d14,c19970ec,8133000,0) at kqueue+0x120

syscall(2f,2f,2f,8133000,2d) at syscall+0x272

Xint0x80_syscall() at Xint0x80_syscall+0x1f

--- syscall (362), eip = 0x282a7b8f, esp = 0xbfbfbcac, ebp = 0xbfbfc368 ---

The major stress testers I know about are using `make -j` with `-DUSE_KQUEUE` compilation flags on `make(1)` and `src/tools/regression/gaithstress`. I notice no problems, but I haven't also tested to make sure nested kqueues are doing what is expected. The only missing feature is the ill-conceived `NOTE_TRACK`. I do not think that returning for that one note type `EINVAL` is a deal-breaker

freebsd-arch: stable kqueue locking up and running on SMP

when trying to make kqueue not suck for 5.3; it is far more useful as a novelty than utility, especially considering I've never seen mention of it in actual software that uses kqueue.

--

Brian Fundakowski Feldman
<> green@FreeBSD.org
Opinions expressed are my own.

```
\'[ FreeBSD ]'''''''''''\
 \  The Power to Serve!  \
 \.....\
```

freebsd-arch@freebsd.org mailing list

<http://lists.freebsd.org/mailman/listinfo/freebsd-arch>

To unsubscribe, send any mail to "freebsd-arch-unsubscribe@freebsd.org"