

Re: duplicate read/write locks in net/pfil.c and netinet/ip_fw2.c

Source: <http://unix.derkeiler.com/Mailing-Lists/FreeBSD/arch/2005-08/0144.html>

From: Julian Elischer (julian_at_elischer.org)

Date: 08/17/05

Date: Tue, 16 Aug 2005 20:20:17 -0700

To: Max Laier <max@love2party.net>

Max Laier wrote:

> *On Wednesday 17 August 2005 02:05, Luigi Rizzo wrote:*
>
>> *[apologies for the cross post but it belongs both to arch and net.]*
>>
>> *I notice that net/pfil.c and netinet/ip_fw2.c have two copies of*
>> *aisimilar but slightly different implementation of*
>> *multiple-reader/single-writer locks, which brings up the question(s):*
>>
>> *1. should we rather put this code in the generic kernel code so that other*
>> *subsystems could make use of it ? E.g. the routing table is certainly*
>> *a candidate,*
>
>
> *I have asked this several time on -arch and IRC, but never found anyone*
> *willing to pursue it. However, the problem is ...*
>
>
>> *and especially*
>>
>> *2. should we implement it right ?*
>>
>> *Both implementations are subject to starvation for the writers*
>> *(which is indeed a problem here, because we might want to modify*
>> *a ruleset and be prevented from doing it because of incoming traffic*
>> *that keeps readers active).*
>> *Also the PFIL_TRY_WLOCK will in fact be blocking if a writer*
>> *is already in - i have no idea how problematic is this in the*
>> *way it is actually used.*
>
>
> *... really this. I didn't find a clean way out of the starvation issue. What*
> *I do for pfil is that I set a flag and simply stop serving[2] shared requests*
> *once a writer waits for the lock. If a writer can't sleep[1] then we return*
> *EBUSY and don't. However, for pfil it's almost ever safe to assume that a*

freebsd-arch: Re: duplicate read/write locks in net/pfil.c and netinet/ip_fw2.c

- > write may sleep (as it is for most instances of this kind of sx-lock where
- > you have *BIGNUM*xreads:1xwrite).
- >
- > [1] Note that there is a **big** difference between blocking and sleeping.
- > These two are usually confused. While it is almost always okay to block it
- > is seldom okay to sleep. The existing sx(9) api has the problem that it
- > **sleeps** in the shared path which renders it unusable for this usecase (as we
- > might be holding other locks and must not sleep in the shared path).
- > However, sleeping in the shared path is one (?the only?) way out of the
- > starvation problem – other than a problem specific as done for pfil.
- >
- > [2] See pfil(9) BUGS.

netgraph has yet another implementation of R/W locks.

It relies on the fact that every lock action is done on behalf of a command request or a data processing request, each of which is queueable, and each RW lock is associated with a queue.

Instead of blocking, the item is queued instead for later processing.

>

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"