

Re: sched_userret priority adjustment patch for sched_4bsd

Source: <http://unix.derkeiler.com/Mailing-Lists/FreeBSD/arch/2004-09/0224.html>

From: Julian Elischer (julian_at_elischer.org)

Date: 09/29/04

Date: Wed, 29 Sep 2004 13:40:57 -0700

To: John Baldwin <jhb@FreeBSD.org>

John Baldwin wrote:

>On Wednesday 29 September 2004 10:38 am, Stephan Uphoff wrote:

>

>

>>On Tue, 2004-09-28 at 10:56, John Baldwin wrote:

>>

>>

>>>If A has a priority boost from `tsleep()` this is intentional, however.

>>>The priority boosts from `tsleep()` are *supposed* to do this so as to

>>>favor interactive tasks. Note that if you add the code to always raise

>>>`td_priority` while in the kernel as below you may end up defeating this

>>>well-known feature of the 4BSD scheduler.

>>>

>>>

>>OK - you and Julian convinced me that this is a feature that I should

>>have known about. Without test cases or interactivity benchmarks

>>discussions if this is still a desirable feature are probably useless.

>>I will revisit this once test cases materialize or I have time to

>>think about a benchmark (Not likely anytime soon).

>>

>>

>

>That's ok. This discussion has been very fruitful on my end at least as

>talking this out has helped me get a much better grasp on how this stuff

>works on 4.x and should be done in 5.x to obtain at least somewhat similar

>behavior.

>

well if you've worked it out,.. do let the rest of us know :-)

I do think that there are several points that need work..

1/ kse threads are ephemeral, and so they don't gather any 'history'.

therefore it needs to be gathered somewhere else.. (e.g. the `ksegrp`, but what does that actually mean?)

freebsd-arch: Re: sched_userret priority adjustment patch for sched_4bsd

- 2/ what if the kg has both long-running and interactive threads?
- 3/ sibling thread affinity and how that affects priority and scheduling.

We COULD store information in the mailbox..
but then we need to trust the user with it..
So then where do we store it?

I have considered a store of 'cached' and "hashed" (like the buffer cache) sched-info structs that are recycled in a least-recently used manner.. when you get a thread with a mailbox you look for a sched-stats block corresponding with that mailbox address and use it..
if you don't find it then you know that thread has not run for a long time..
so you grab the least-recently used one and recycle it as that thread hasn't run for a while.
Basically the kernel could keep stats on behalf of the most active KSE threads in an efficient manner.
The small stats structs would need to be only about 8 words..
(4 for 2 x double links. one for mailbox addr/key, and 3 for sched stats.)
In effect the kernel keeps tabs on the most active user threads without the UTS knowing about it.

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"