

Re: Lockless uidinfo.

Source: <http://unix.derkeiler.com/Mailing-Lists/FreeBSD/arch/2007-08/msg00054.html>

- *From:* John Baldwin <jhb@xxxxxxxxxxxx>
 - *Date:* Tue, 21 Aug 2007 17:53:34 -0400
-

On Tuesday 21 August 2007 03:19:02 pm Pawel Jakub Dawidek wrote:

Memory barriers on another CPU don't mean anything about the CPU thread
2

is

on. Memory barriers do not flush caches on other CPUs, etc. Normally

when

objects are refcounted in a table, the table holds a reference on the

object,

but that doesn't seem to be the case here. [...]

But the memory barrier from 'mtx_lock(&uihashtbl_mtx)' above
'if (uip->ui_ref > 0)' would do the trick and I can safely avoid using
atomic read in this if statement, right?

Yes, though that may be rather non-obvious to other people, or to yourself 6
months down the road. You should probably document it.

[...] Have you tried doing something
very simple in uifree():

```
{
mtx_lock(&uihashtbl_mtx);
if (refcount_release(...)) {
LIST_REMOVE();
mtx_unlock(&uihashtbl_mtx);
...
free();
} else
mtx_unlock(&uihashtbl_mtx);
```

Re: Lockless uidinfo.

}

I wouldn't use a more complex algo in uifree() unless the simple one is shown

to perform badly. Needless complexity is a hindrance to future maintenance.

Of course we could do that, but I was trying really hard to remove contention in the common case. Before we used UIDINFO_LOCK() in the common case, now you suggesting using global lock here, and I'd really, really prefer using one atomic only.

sigh You ignored my last sentence. You need to think about other people who come and read this later or yourself 12 months from now when you've paged out all the uidinfo stuff from your head.

Hmm, what happens if you get this:

thread A

refcount_release() – drops to 0

preempted

thread B

uihold() – refcount goes to 1

...

uifree() – refcount goes to 0

removes object from table and frees it

resumes

mtx_lock(&uihashtbl);

sees refcount of 0 so tries to destroy object again

BOOM (corruption, panic, etc.)

This is the race that the current code handles by taking a reference on the uid while it drops the uidinfo lock to acquire the table lock.

Probably you need to not drop the last reference unless you hold the uihashtbl_mtx, which means not using refcount_release() and manually use an atomic_cmpset_int() if the refcount is > 1 to drop a ref to optimize the

Re: Lockless uidinfo.

Re: Lockless uidinfo.

common case:

```
old = uip->ui_refs;
if (old > 1) {
if (atomic_cmpset_int(&uip->ui_refs, old, old - 1))
return;
}
```

```
mtx_lock(&uihashtbl_mtx);
if (refcount_release(&uip->ui_refs)) {
/* remove from table and free */
}
mtx_unlock(&uihashtbl_mtx);
```

--

John Baldwin

freebsd-arch@xxxxxxxxxxxxx mailing list

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

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