Subject: Re: Containers: css_put() dilemma
Posted by Balbir Singh on Tue, 17 Jul 2007 17:40:59 GMT
View Forum Message <> Reply to Message

Paul (??) Menage wrote:
> Because as soon as you do the atomic_dec_and_test() on css->refcnt and
> the refcnt hits zero, then theoretically someone other thread (that
> already holds container_mutex) could check that the refcount is zero
> and free the container structure.
>

Hi, Paul,

That sounds correct. I wonder now if the solution should be some form
of delegation for deletion of unreferenced containers (HINT: work queue
or kernel threads).

> Adding a synchronize_rcu in container_diput() guarantees that the
> container structure won't be freed while someone may still be
> accessing it.
>

Do we take rcu_read_lock() in css_put() path or use call_rcu() to
free the container?

>>
>> Could you please elaborate as to why using a release agent is broken
>> when the memory controller is attached to it?
>
> Because then it will try to take container_mutex in css_put() if it
> drops the last reference to a container, which is the thing that you
> said you had to avoid since you called css_put() in contexts that
> couldn't sleep.
>
> Paul

--
Warm Regards,
Balbir Singh
Linux Technology Center
IBM, ISTL

_____
Containers mailing list
Containers@lists.linux-foundation.org
https://lists.linux-foundation.org/mailman/listinfo/containers