Subject: Re: Containers: css_put() dilemma
Posted by Paul Menage on Tue, 17 Jul 2007 02:35:01 GMT
View Forum Message <> Reply to Message

On 7/16/07, Balbir Singh <balbir@linux.vnet.ibm.com> wrote:
>
> - if (notify_on_release(cont)) {
> + if (atomic_dec_and_test(&css->refcnt) && notify_on_release(cont)) {

This seems like a good idea, as long as atomic_dec_and_test() isn't noticeably more expensive than atomic_dec(). I assume it shouldn't need to be, since the bus locking operations are presumably the same in each case.

> mutex_lock(&container_mutex);
> set_bit(CONT_RELEASABLE, &cont->flags);
> - if (atomic_dec_and_test(&css->refcnt)) {
> - check_for_release(cont);
> - }
> + check_for_release(cont);
> mutex_unlock(&container_mutex);
>
> That way we set the CONT_RELEASABLE bit only when the ref count drops
> to zero.
>

That's probably a good idea, in conjunction with another part of my patch for this that frees container objects under RCU - as soon as you do the atomic_dec_and_test(), then in theory some other thread could delete the container (since we're no longer going to be taking container_mutex in this function). But as long as the container object remains valid until synchronize_rcu() completes, then we can safely set the CONT_RELEASABLE bit on it.

>
> Yes, that is correct, the advantage is that with can_destroy() we
> don't need to go through release synchronization each time we do
> a css_put().

I think the amount of release synchronization *needed* is going to be the same whether you have the refcounting done in the subsystem or in the framework. But I agree that right now we're doing one more atomic op than we strictly need to, and can remove it.

Paul

_____
Containers mailing list
Containers@lists.linux-foundation.org