Subject: Re: Containers: css_put() dilemma
Posted by Paul Menage on Tue, 17 Jul 2007 16:15:58 GMT
View Forum Message <> Reply to Message

On 7/17/07, Dave Hansen <haveblue@us.ibm.com> wrote:
> On Tue, 2007-07-17 at 08:49 -0700, Paul ($BJuN\ (B) 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.
>
> Then that other task had a reference and itself should have bumped the
> count, and the other user would never have seen it hit zero.

Nope. The container could have been empty (of tasks) and hence had a zero count.

The liveness model used by containers is that when the refcount hits zero, the container isn't immediately destroyed (since it can contain useful historical usage data, etc) but simply becomes eligible for destruction by userspace via an rmdir().

>
> Even if there are still pages attached to the container, why not just
> have those take a reference, and don't bother actually freeing the
> container until the last true reference is dropped?

Yes, we could potentially just use the main count variable rather than having separate per-subsystem extra refcounts. The main reasons to do it this way are:

- the root subsystem state for a subsystem can shift between different "struct container" objects if it was previously inactive and gets mounted as part of a hierarchy (or similarly, gets unmounted and goes inactive). Possibly we could get around this by simply saying not doing refcounting on the subsys states attached to root containers since they can never be freed anyway.

- At some point I'd like to be able to support shifting subsystems between active hierarchies, at least in limited cases such as where the hierarchies are isomorphic, or binding/unbinding subsystems to/from active hierarchies. At that point we definitely need to be able to split out the different subsystem state refcounts from one another in the same hierarchy.

- we'd still have the issue that Balbir wants to be able to drop a reference in a non-sleeping context, and we want to avoid doing excessive synchronization in the normal case when the css_put() doesn't put the reference count to zero.

>
> Does it matter if the destruction callbacks don't happen until well
> after an attempt to destroy the container is made?

Well that's sort of the point of putting a synchronize_rcu() in
container_diput() - it ensures that the actual destruction of the
object doesn't occur until sufficiently after the destruction attempt
is initiated that no one is still using the reference.

The alternative would be something that polls to spot whether
refcounts have reached zero and if so runs the userspace helper. That
doesn't seem particularly palatable when you have large numbers of
containers, if we can avoid it easily.

Paul

_____