Subject: Re: [PATCH 3/7] containers (V7): Add generic multi-subsystem API to containers
Posted by Srivatsa Vaddagiri on Mon, 12 Feb 2007 15:27:38 GMT
View Forum Message <> Reply to Message

On Mon, Feb 12, 2007 at 12:15:24AM -0800, menage@google.com wrote:
> +/*
> + * Call css_get() to hold a reference on the container; following a
> + * return of 0, this container subsystem state object is guaranteed
> + * not to be destroyed until css_put() is called on it.  A non-zero
> + * return code indicates that a reference could not be taken.
> + *
> + */
> +

Why can't we reuse container->count (or container_group->ref) to
refcount the per-subsystem object attached to a container? I think
that is how it is done for cpusets? That would make css_get/put
unnecessary?

> +static inline int css_get(struct container_subsys_state *css)
> +{
> + int retval = 0;
> + unsigned long flags;
> + /* Synchronize with container_rmdir() */
> + spin_lock_irqsave(&css->refcnt_lock, flags);
> + if (atomic_read(&css->refcnt) >= 0) {
> +  /* Container is still alive */
> +  atomic_inc(&css->refcnt);
> + } else {
> +  /* Container removal is in progress */
> +  retval = -EINVAL;
> + }
> + spin_unlock_irqrestore(&css->refcnt_lock, flags);
> + return retval;
> +}

--
Regards,
vatsa

Subject: Re: [PATCH 3/7] containers (V7): Add generic multi-subsystem API to containers
Posted by Paul Menage on Mon, 12 Feb 2007 18:40:52 GMT
View Forum Message <> Reply to Message

On 2/12/07, Srivatsa Vaddagiri <vatsa@in.ibm.com> wrote:

> On Mon, Feb 12, 2007 at 12:15:24AM -0800, menage@google.com wrote:
> > +/*
> > + * Call css_get() to hold a reference on the container; following a
> > + * return of 0, this container subsystem state object is guaranteed
> > + * not to be destroyed until css_put() is called on it.  A non-zero
> > + * return code indicates that a reference could not be taken.
> > + *
> > + */
> > +
>
> Why can't we reuse container->count (or container_group->ref) to
> refcount the per-subsystem object attached to a container? I think
> that is how it is done for cpusets? That would make css_get/put
> unnecessary?

I did consider that approach at one point. The reason I rejected it
was that then container->count would no longer even vaguely represent
the number of processes in a container. Now that we have the
container_group object, we have to use that for counting the number of
processes in a container anyway, so that objection goes away.

However, I think it's important to be able to provide some kind of a
reference count that subsystems can grab (e.g. to store a reference in
a non-task object such as a file struct) without taking manage_mutex
or callback_mutex (since that would be excessively heavyweight) but
which can still be "frozen" at zero at the point when you're trying to
destroy a container. Additionally, having it per subsystem will be
important for when we implement arbitrary binding/unbinding of
subsystems from hierarchies - at that point we need to be able know
which subsystems have external reference counts, and hence aren't
removeable.

Paul

---

Subject: Re: [PATCH 3/7] containers (V7): Add generic multi-subsystem API to containers
Posted by Srivatsa Vaddagiri on Tue, 13 Feb 2007 13:19:45 GMT
View Forum Message <> Reply to Message

On Mon, Feb 12, 2007 at 10:40:52AM -0800, Paul Menage wrote:
> I did consider that approach at one point. The reason I rejected it
> was that then container->count would no longer even vaguely represent
> the number of processes in a container. Now that we have the
> container_group object, we have to use that for counting the number of
> processes in a container anyway, so that objection goes away.

Yep.

> However, I think it's important to be able to provide some kind of a
> reference count that subsystems can grab (e.g. to store a reference in
> a non-task object such as a file struct) without taking manage_mutex
> or callback_mutex (since that would be excessively heavyweight) but
> which can still be "frozen" at zero at the point when you're trying to
> destroy a container.

Well, we already bump up reference count in fork() w/o grabbing those
mutexes don't we? Also if rmdir() sees container->count to be zero, then
it means no task is attached to the container. How will then a function
like bc_file_charge() bump up the reference count to such a container
(presuming it wanted to do so w/o manage/callback mutexes -and- that the
container pointer in bc_file_charge is derived from some task in
that container). I think it is safe to bump up container->count in
bc_file_charge w/o grabbing manage/callback mutexes.

> Additionally, having it per subsystem will be
> important for when we implement arbitrary binding/unbinding of
> subsystems from hierarchies - at that point we need to be able know
> which subsystems have external reference counts, and hence aren't
> removeable.

Are you talking about (un)bind of subsystem to/from hierararchies that
have non-zero containers in them? That sounds very icky. Anyway that
doesnt seem to be supported in current patches.

Basically I felt we should defer introducing css_get/put until we find a good
user for it, (and bc_file_(un)charge dont seem to be good users of it-
see above).

--
Regards,
vatsa

---

Subject: Re: [PATCH 3/7] containers (V7): Add generic multi-subsystem API to containers
Posted by Paul Menage on Thu, 15 Feb 2007 01:17:09 GMT
View Forum Message <> Reply to Message

On 2/13/07, Srivatsa Vaddagiri <vatsa@in.ibm.com> wrote:
>
> Well, we already bump up reference count in fork() w/o grabbing those
> mutexes don't we? Also if rmdir() sees container->count to be zero, then
> it means no task is attached to the container. How will then a function

> like bc_file_charge() bump up the reference count to such a container
> (presuming it wanted to do so w/o manage/callback mutexes -and- that the
> container pointer in bc_file_charge is derived from some task in
> that container). I think it is safe to bump up container->count in
> bc_file_charge w/o grabbing manage/callback mutexes.

Right, I was never suggesting that we take either of those mutexes for this operation. The spin lock in css_get() was an attempt to avoid that. But I think you're right that it was too heavyweight, and can be avoided with atomic operations. See my other email to Pavel.

>
> Are you talking about (un)bind of subsystem to/from hierararchies that
> have non-zero containers in them? That sounds very icky. Anyway that
> doesnt seem to be supported in current patches.

The bind/unbind from active hierarchies is supported in the user-space API, and it's implemented for hierarchies that have no child containers. Hence it's important, at least conceptually, for the reference count to be held by the subsystem state rather than the container.

Implementing a full bind/unbind for arbitrary subsystems and hierarchies will indeed be a lot of work, which is why I'm not trying to do it at this point.

Paul