

---

Subject: Re: [RFC] Transactional CGroup task attachment  
Posted by [Paul Menage](#) on Sat, 12 Jul 2008 00:18:18 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

On Fri, Jul 11, 2008 at 5:03 PM, Matt Helsley <matthlrc@us.ibm.com> wrote:

```
>> struct cgroup_attach_state {  
>  
> nit: How about naming it cgroup_attach_request or  
> cgroup_attach_request_state? I suggest this because it's not really  
> "state" that's kept beyond the prepare-then-(commit|abort) sequence.
```

State doesn't have to be long-lived to be state. But I'm not too worried about the exact name for it, if people have other preferences.

```
>  
> What about the task->alloc_lock? Might that need to be taken by multiple  
> subsystems? See my next comment.
```

My thought was that cgroups would take that anyway prior to calling `prepare_attach_nosleep()`, since it's a requirement for changing `task->cgroups` anyway.

```
>  
> Rather than describing what might be called later for each API entry  
> separately it might be simpler to prefix the whole API/protocol  
> description with something like:  
> =====  
> A successful return from prepare_X will cause abort_X to be called if  
> any of the preparatory calls fail. (where X is either sleep or nosleep)  
>  
> A successful return from prepare_X will cause commit to be called if all  
> of the preparatory calls succeed. (where X is either sleep or nosleep)  
>  
> Otherwise no calls to abort_X or commit will be made. (where X is either  
> sleep or nosleep)
```

I'll play with working that into the description.

```
> I think that's correct based on your descriptions. Of course changing  
> this only makes sense if this proposal will go into Documentation/ in  
> some form..
```

Yes, we'd definitely need to document this in some detail.

```
>  
> Allowing prepare_X to hold locks when it has exited seems ripe for  
> introducing two separate subsystems that inadvertently take locks out of  
> order.
```

Yes, but I'm not sure that there's much that we can do about that. If we want to guarantee to be able to rollback one subsystem when a later subsystem fails then we have to let the earlier subsystems continue to hold locks. Or is this too ambitious a goal to support?

Paul

---

Containers mailing list

Containers@lists.linux-foundation.org

<https://lists.linux-foundation.org/mailman/listinfo/containers>

---