Subject: Re: [PATCH 3/8] CGroup Files: Move the release_agent file to use typed handlers
Posted by Paul Menage on Tue, 24 Jun 2008 23:30:35 GMT
View Forum Message <> Reply to Message

On Tue, Jun 24, 2008 at 4:23 PM, Andrew Morton
<akpm@linux-foundation.org> wrote:

```
>> +/**
>> + * cgroup_lock_live_group - take cgroup_mutex and check that cgrp is alive.
>> + * @cgrp: the cgroup to be checked for liveness
>> + *
>> + * Returns true (with lock held) on success, or false (with no lock
>> + * held) on failure.
>> + */
>> +int cgroup_lock_live_group(struct cgroup *cgrp)
>> +{
>> +	mutex_lock(&cgroup_mutex);
>> +	if (cgroup_is_removed(cgrp)) {
>> +		mutex_unlock(&cgroup_mutex);
>> +		return false;
>> +	}
>> +	return true;
>> +}
>
> I think that if we're going to do this it would be nice to add a
> symmetrical cgroup_unlock_live_group()?
```

There's already a cgroup_unlock() function exported in cgroup.h - that's the counterpart to both cgroup_lock() and cgroup_lock_live_group(). I can add a comment about this in the docs for cgroup_lock_live_cgroup().

```
>
> Because code like this:
>
>> +	if (!cgroup_lock_live_group(cgrp))
>> +		return -ENODEV;
>> +	strcpy(cgrp->root->release_agent_path, buffer);
>> +	mutex_unlock(&cgroup_mutex);
>
> is a bit WTFish, no?  it forces each caller of cgroup_lock_live_group()
> to know about cgroup_lock_live_group() internals.
```

cgroup_mutex isn't directly exported outside of cgroup.c, so real callers would have no choice but to use cgroup_unlock() in this instance. I guess it could make sense to be consistent and use cgroup_unlock() within cgroup.c as well.

Paul

_____

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