Subject: Re: [ckrm-tech] [PATCH 1/7] containers (V7): Generic container system abstracted from cpusets code
Posted by Paul Jackson on Sun, 25 Mar 2007 05:43:48 GMT
View Forum Message <> Reply to Message

vatsa wrote:
> Not just this, continuing further we have more trouble:
>
> ---------- --------
> CPU0 (attach_task T1 to CS2) CPU1 (T1 is exiting)
> ---------- --------
>
> synchronize_rcu()
> atomic_dec(&CS1->count);
> [CS1->count = 0]
>
> if atomic_dec_and_test(&oldcs->count))
> [CS1->count = -1]
> ...
> 2nd race is tricky. We probably need to do this to avoid it:
>
> task_lock(tsk);
>
> /* Check if tsk->cpuset is still same. We may have raced with
> * cpuset_exit changing tsk->cpuset again under our feet.
> */
> if (tsk->cpuset == cs && atomic_dec_and_test(&oldcs->count)) {

I'm unsure here, but this 'tsk->cpuset == cs' test feels fragile to me.

How about a bit earlier in attach_task(), right at the point we overwrite the victim tasks cpuset pointer, we decrement the count on the old cpuset, and if it went to zero, remember that we'll need to release it, once we've dropped some locks:

```
static int attach_task(struct cpuset *cs, char *pidbuf, char **ppathbuf)
{
 ...
 struct cpuset *oldcs;
 struct cpuset *oldcs_tobe_released;

 ...

        task_lock(tsk);
        oldcs = tsk->cpuset;
        ...
        if (tsk->flags & PF_EXITING) {
                ...
```

```
        }
        atomic_inc(&cs->count);
        rcu_assign_pointer(tsk->cpuset, cs);
 oldcs_tobe_released = NULL;
 if (atomic_dec_and_test(&oldcs->count))
  oldcs_tobe_released = oldcs;
        task_unlock(tsk);

 ...
 put_task_struct(tsk);
        synchronize_rcu();
        if (oldcs_tobe_released)
                check_for_release(oldcs_tobe_released, ppathbuf);
        return 0;
}

--
                I won't rest till it's the best ...
                Programmer, Linux Scalability
                Paul Jackson <pj@sgi.com> 1.925.600.0401
```