Subject: Re: [RFC] cpuset update_cgroup_cpus_allowed
Posted by Paul Jackson on Tue, 16 Oct 2007 06:07:29 GMT
View Forum Message <> Reply to Message

> > + if (cpus_equal(*cpus, t->cpus_allowed))
> > + continue;
> > ...
> > + for (q = tasks; q < p; q++) {
> > + set_cpus_allowed(*q, *cpus);
> > + put_task_struct(*q);
> > + }
> > + }
> > +}
>
> Yet by not doing any locking here to prevent a cpu from being
> hot-unplugged, you can race and allow the hot-unplug event to happen
> before calling set_cpus_allowed(). That makes this entire function a
> no-op with set_cpus_allowed() returning -EINVAL for every call, which
> isn't caught, and no error is reported to userspace.
>
> Now all the tasks in the cpuset have an inconsistent state with respect to
> their p->cpuset->cpus_allowed, because that was already updated in
> update_cpumask().

My solution may be worse than that. Because set_cpus_allowed() will
fail if asked to set a non-overlapping cpumask, my solution could never
terminate. If asked to set a cpusets cpus to something that went off
line right then, this I'd guess this code could keep looping forever,
looking for cpumasks that didn't match, and then not noticing that it
was failing to set them so as they would match.

... it needs work ... or the alternative solution from Paul M.

--

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

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