Subject: Re: Re: Hang with fair cgroup scheduler (reproducer is attached.)
Posted by Dmitry Adamushko on Sat, 15 Dec 2007 11:15:16 GMT
View Forum Message <> Reply to Message

On 15/12/2007, Dhaval Giani <dhaval@linux.vnet.ibm.com> wrote:
> On Sat, Dec 15, 2007 at 11:22:08AM +0100, Dmitry Adamushko wrote:
> > On 14/12/2007, Steven Rostedt <rostedt@goodmis.org> wrote:
> > >
> > > On Fri, 14 Dec 2007, Dmitry Adamushko wrote:
> > >
> > > >
> > > > argh... it's a consequence of the 'current is not kept within the tree" indeed.
> > > >
> > >
> > > Thanks Dmitry for tracking this down.
> >
> > My analysis was flawed (hmm... me was under control of Belgium beer :-)
> >
> > The task in not on the runqueue (p->on_rq == 0) at the moment when
> > put_prev_task_fair() and set_curr_task_fair() get its turn in
> > sched_move_task()... so dequeue/enqueue_entity() are not triggered,
> > that's good.
> >
>
> Again, I am probably missing something, but if on_rq == 0, then how is
> set_curr_task_fair() getting called?
>

...
      running = task_running(rq, tsk);
      on_rq = tsk->se.on_rq;

// let's say on_rq == 1 , i.e. the task is on the runqueue

      if (on_rq) {
            dequeue_task(rq, tsk, 0);

// now tsk->se.on_rq becomes 0

            if (unlikely(running))
                  tsk->sched_class->put_prev_task(rq, tsk);

// put_prev_task() --> put_prev_entity() checks for 'tsk->se.on_rq' to
determine whether __enqueue_entity() must be done ---> and it's 0 in
our case.

[it can be non-zero for the following path : schedule() -->
put_prev_task(..., prev) when deactivate_task(..., prev) was not

previously called in schedule(), i.e. 'prev' was preempted]

tsk->se.on_rq will become 1 only after enqueue_task(). As a result,
tsk->se.on_rq is still 0 when set_curr_task() is executed.

does it make sense now?

> --
> regards,
> Dhaval
>

--
Best regards,
Dmitry Adamushko

_____