Subject: Re: [PATCH 1/1, v6] cgroup/freezer: add per freezer duty ratio control
Posted by Matt Helsley on Thu, 10 Feb 2011 19:11:17 GMT

View Forum Message <> Reply to Message

On Wed, Feb 09, 2011 at 07:06:15PM -0800, Arjan van de Ven wrote:
> On 2/9/2011 7:04 PM, Matt Helsley wrote:
> >On Tue, Feb 08, 2011 at 05:05:41PM -0800, jacob.jun.pan@linux.intel.com wrote:
> >>From: Jacob Pan<jacob.jun.pan@linux.intel.com>
> >>
> >>Freezer subsystem is used to manage batch jobs which can start
> >>stop at the same time. However, sometime it is desirable to let
> >>the kernel manage the freezer state automatically with a given
> >>duty ratio.
> >>For example, if we want to reduce the time that backgroup apps
> >>are allowed to run we can put them into a freezer subsystem and
> >>set the kernel to turn them THAWED/FROZEN at given duty ratio.
> >>
> >>This patch introduces two file nodes under cgroup
> >>freezer.duty_ratio_pct and freezer.period_sec
> >>
> >>Usage example: set period to be 5 seconds and frozen duty ratio 90%
> >>[root@localhost aoa]# echo 90> freezer.duty_ratio_pct
> >>[root@localhost aoa]# echo 5000> freezer.period_ms
> >I kept wondering how this was useful when we've got the "cpu" subsystem
> >because for some reason "duty cycle" made me think this was a scheduling
> >policy knob. In fact, I'm pretty sure it is -- it just happens to
> >sometimes reduce power consumption.
> >
> >Have you tried using the cpu cgroup subsystem's share to see if it can
> >have a similar effect?
>
> does the cpu cgroup system work on a 20 to 30 second time window?

I don't think so -- it works directly with the scheduler IIRC.

> the objective is to have the CPU idle, without wakeups, for that long...
> (to save power)

Hmm. Maybe these need some "scheduler slack" so that when they're
runnable they'll either run with other scheduled entities that are
keeping the cpu awake or wait until the slack runs out before doing
a wakeup. Then you can toss this in the cgroup timer slack subsystem
and rename it to the "wakeup slack" subsystem or something. That
will probably get you better race-to-idle behavior, avoid/reduce
latencies added by duty cycles, and avoid shoehorning into the cgroup
freezer subsystem. Of course it would require some scheduler hacking
which will probably be much more controversial than modifying a cgroup
subsystem :).

Cheers,
 -Matt Helsley

____
Containers mailing list
Containers@lists.linux-foundation.org
https://lists.linux-foundation.org/mailman/listinfo/containe rs