Re: [RFC] CPU controllers?

Source: http://linux.derkeiler.com/Mailing−Lists/Kernel/2006−06/msg04442.html

- *From*: Sam Vilain <sam@xxxxxxxxxx>
- *Date*: Fri, 16 Jun 2006 09:52:16 +1200

Srivatsa Vaddagiri wrote:

> One possibility is to add a basic controller, that addresses some minimal
> requirements, to begin with and progressively enhance it capabilities. From this
> pov, both the f−series resource group controller and cpu rate−cap seem to be
> good candidates for a minimal controller to begin with.
>
> Thoughts?

Sounds like you're on the right track, but I don't know whether we can truly be happy making the performance/guarantee trade−off decision for the user.

You could grossly put the solutions into several camps;

1. solutions which have very low impact and provide soft assurances only
2. solutions which provide hard limits
3. solutions which provide guarantees

I think it's almost invariant that the latter solutions have more of a performance impact, and that it's quite important that normal system throughput does not suffer from the "scheduling namespace" solution that we come up with.

> Salient features of various CPU controllers that have been proposed so far are
> summarized below. I have not captured OpenVZ and Vserver controller aspects
> well. Request the maintainers to fill−in!
> [...]
> 2. Timeslice scaling (Maeda Naoaki and Kurosawa Takahiro)
>
> Features:
> * Provide guaranteed CPU execution rate on a per−task−group basis
> Guarantee provided over an interval of 5 seconds.
> * Hooked to Resource Group infrastructure currently and hence
> guarantee/limit set thr' Resource Group's RCFS interface.
> * Achieves guaranteed execution by scaling down timeslice of tasks

> who are above their guaranteed execution rate. Timeslice can be
> scaled down only to a minimum of 1 slice.
> * Does not scale down timeslice of interactive tasks (even if their
> CPU usage is beyond what is guaranteed) and does not avoid requeue
> of interactive tasks.
> * Patch is quite simple
>
> Limitations:
> * Does not support limiting task−group CPU execution rate
>
> Drawbacks:
> (Some of the drawbacks listed are probably being addressed currently
> with a redesign − which we are yet to see)
>
> * Interactive tasks (and their requeuing) can come in the way of
> providing guaranteed execution rate to other tasks
> * SMP load balancing does not take into account guarantee provided to
> task groups.
> * It may not be possible to restrict CPU usage of a task group to only
> its guaranteed usage if the task−group has large number of tasks
> (each task is run for a minimum of 1 timeslice)
> * May not handle bursty loads
>
> [...]
> 4. VServer CPU controller
>
> Features:
> − Token−bucket based

The VServer scheduler is also timeslice scaling − it just uses the token bucket to know how much to scale the timeslices. It doesn't care about interactive bonuses, although it does lessen the interactivity bonus a notch or two (to −5..+5).

This means that it's performance neutral in the general case.

> Drawbacks:
> − ?

It fits into category 1 (or, using Herbert Poetzl's enhancements, 2), so does not provide guarantees.

> Limitations:
> − ?

Doesn't deal with huge numbers of processes; but with task group ulimits
that problem goes away in practice.

Sam.
−