> hm, this shouldnt be the case. Can you see this with -v14?

The group fairness patches fail to apply over -v14. Just to make it clearer, I am not talking about vanilla CFS but group fairness CFS, I have no problems with CFS.

The symptoms in the qemu box are:
o two busy loops with UID=1 remove a bit of interactivity only to UID=1 tasks
o two busy loops with UID=0 make the system unusable

Another tidbit: /proc/sched_debug now deals with users, so the task and PID fields show some random memory garbage.

My do_div patch was nonsense (do_div returns the remainder, not the quotient). Attached is a corrected patch.

> for containers it's exactly the right behavior: group scheduling is
> really a 'super' container concept that allows the allocation of CPU
> time regardless of how a group uses it. The only additional control we
> might want is to allocate different amount of CPU time to different
> groups. (i.e. a concept vaguely similar to "nice levels", but at the
> group level - using a different and saner API than nice levels.) Nice
> levels are really only meaningful at the lowest level.

OK for containers, but then this should not replace the standard nice implementation as this CPU repartition would be unexpected IMHO. I would expect the group CPU allocation to be averaged from the nice levels of its elements.

As a sidenote, while in CFS-v13 a nice=0 tasks seems to get 10x more CPU than a nice=10 one, with the group fairness patch, the ratio drops to less than 2x (for tasks with the same UID).

Regards.

--
Guillaume

_____
Containers mailing list
Containers@lists.linux-foundation.org