Subject: Re: [ckrm-tech] [PATCH 0/2] resource control file system - aka containers on top of nsproxy!
Posted by Paul Menage on Mon, 12 Mar 2007 10:00:25 GMT
View Forum Message <> Reply to Message

On 3/11/07, Paul Jackson <pj@sgi.com> wrote:
>
> My current understanding of Paul Menage's container patch is that it is
> a useful improvement for some of the metered classes - those that could
> make good use of a file system like hierarchy for their interface.
> It probably doesn't benefit all metered classes, as they won't all
> benefit from a file system like hierarchy, or even have a formal name
> space, and it doesn't seem to benefit the name space implementation,
> which happily remains flat.

Well, what I was aiming at was a generic mechanism that can handle "namespaces", "metered classes" and other ways of providing per-task-group behaviour. So a system-call API doesn't necessarily have the right flexibility to implement the possible different kinds of subsystems I envisage.

For example, one way to easily tie groups of processes to different network queues is to have a tag associated with a container, allow that to propagate to the socket/skbuf priority field, and then use standard Linux traffic control to pick the appropriate outgoing queue based on the skbuf's tag.

This isn't really a namespace, and it isn't really a "metered class". It's just a way of associating a piece of data (the network tag) with a group of processes.

With a filesystem-based interface, it's easy to have a file as the method of reading/writing the tag; with a system call interface, then either the interface is sufficiently generic to allow this kind of data association (in which case you're sort of implementing a filesystem in the system call) or else you have to shoehorn into an unrelated API (e.g. if your system call talks about "resource limits" you might end up having to specify the network tag as a "maximum limit" since there's no other useful configuration data available).

As another example, I'd like to have a subsystem that shows me all the sockets that processes in the container have opened; again, easy to do in a filesystem interface, but hard to fit into a resource-meteting-centric or namespace-centric system call API.

Paul

_____
Containers mailing list