Subject: Re: Thoughts on virtualizing task containers
Posted by Paul Menage on Wed, 26 Sep 2007 05:23:52 GMT
View Forum Message <> Reply to Message

On 9/11/07, Serge E. Hallyn <serue@us.ibm.com> wrote:
> Hi Paul,
>
> did any good ideas come up at the mini-summit or k-s, or were any
> decisions made?
>

Discussions were had, but decisions weren't really made.

My vague thoughts on how to do virtualization are below.

1) Add support for a subsystem state object to be shared with its children. specifically:
- for each subsystem, have a <subsystem.inherit> control file, which defaults to 0. This can only be changed when the cgroup has no children
- any children of a cgroup will share subsystem state with the parent for any subsystems whose <inherit> file is 1

This ties in with a request that Balbir made for being able to share resource limits between different levels of cgroups, but it's also useful for virtualization. It's something I wanted to describe in my OLS talk but didn't really have time for.

2) have a virtualization cgroup subsystem, which like other subsystems can be included in at most one hierarchy. The virtualization subsystem might perhaps be the same thing as the nsproxy subsystem?

3) when mounting a cgroup filesystem, if the virtualization subsystem is mounted, and the caller is not in its root cgroup (i.e. it's a guest), then:

- the guest can only see subsystems in the same hierarchy, which additionally have <inherit> set to 0

- the vfsmount returned from cgroup_get_sb() doesn't refer to the root of the hierarchy, but instead to the cgroup directory that the guest is in

- the guest can only mount a single hierarchy, (which therefore must be a subset of the hierarchy that the guest is running in)

- at the time of mount, the <inherit> bits for any subsystems *not* selected by the guest get set to 1, thus any guest processes share the

same subsystem state for those subsystems (this is analagous to having the subsystem not be mounted, at the root/host level).

This approach is a little more restrictive than I'd like, but I think it should support the basic nested virtual server model reasonable well.

These changes are going to require a little bit of plumbing in the core cgroup code, but should have very little effect on any subsystems themselves, except for a few ways:

- each subsystem will now have a private parent/child tree running through its subsystem states, rather than having to use the main cgroup tree

- there will no longer be a direct mapping from a subsystem state to a cgroup. I'm not sure that this will cause anyone a problem. we'll have to tweak the current cgroup iteration interfaces to instead iterate across all the processes in a subsystem state, which may include multiple cgroups

Paul