Subject: Re: semantics for namespace naming
Posted by serue on Thu, 14 Dec 2006 18:59:47 GMT
View Forum Message <> Reply to Message

Quoting Dave Hansen (haveblue@us.ibm.com):
> On Thu, 2006-12-14 at 09:36 -0600, Serge E. Hallyn wrote:
> > one container corresponds to one nsproxy which is one set of namespaces.
>
> On container has at least one nsproxy associated with it. Did you mean
> to say here that each container has one and only one nsproxy?

Sorry, that was struct container, not container.

Since the containers are hierarchical, you can say that a "container"
"has" all the nsproxies of all it's child containers.

So when there is a container for /vserver1, and from that vserver:

 1. serge logs in and unshares the mount namespace for his
 private /tmp and /home
 2. dave starts a checkpointable job in a private container

The struct container for /vserver1 has just one nsproxy, but it has a
child container for serge's login, and a child container for dave's
checkpointable job, so you can say the vserver1 container has three
nsproxies.

> > As I said, once a process is in a container, it never leaves that
> > container. It only enters additional ones. That model fits everyone's
> > needs, without needing some funky API.
>
> This makes logical sense to me. In practice this has the feel of
> ptracing where the ptracer becomes a temporary parent of the tracee.
>
> The process entering a container temporarily becomes a member of that
> container, but it doesn't completely _stop_ being a member of its
> container. The real_parent of a process being ptraced may not be doing
> all of the parental duties during a ptrace, but it doesn't _stop_ being
> the real_parent.
>
> Maybe I'm stretching the analogy too far :)

Maybe, or maybe you're showing me a kink in my reasoning. I was in
fact thinking that entering a new container would be the one way
to fully disengage from the old container. Meaning it would be best
if it were forced to be done on a clone+exec. But even so, is that
reasonable?

-serge

_____