
Subject: Re: Re: [RFC] [PATCH 0/3] containers: introduction
Posted by [Paul Menage](#) on Fri, 12 Jan 2007 19:03:48 GMT
[View Forum Message](#) <> [Reply to Message](#)

On 1/12/07, Serge E. Hallyn <serue@us.ibm.com> wrote:

>
> I agree, so long as "provided requirements aren't too different" is
> replaced by "provided there is commonality to be merged." Differences in
> lifetime rules and fs behavior could make it pounding a round peg into
> a square hole...

Well, at a minimum they have the commonality of "track this set of
processes, and all their children", and report how the set changes.

>
> We were thinking that each container directory would have a file
> representing each namespace in the nsproxy. To enter only a few
> namespaces out of an existing namespace container, then, you could
> create a directory for a new namespace container, link the namespaces
> you want out of other containers, then enter the container (presumably
> by doing 'echo (container_path) > /proc/\$\$/ns_container')
>
> So in some ways that's actually closer to what you currently have
> than the default container creation rules.

Very similar, yes - the only difference would be that in my model you'd do

echo \$\$ > (container_path)/tasks

But I thought that Eric (and others?) objected to the idea of being
able to move a task into an existing namespace/container on the
grounds of race conditions, etc?

If you were able to go with this model rather than the unshare/clone
model, that would be a lot simpler than implementing a new clone model
for containers. It would also make your namespace work more
useful/flexible, I think - there are definitely cases where I want to
be able to add a new process into an existing
container/namespace/virtual server, which isn't possible with the
unshare/clone model unless the root process in the container is
written to be able to spawn new processes for you.

Paul

Subject: Re: Re: [RFC] [PATCH 0/3] containers: introduction
Posted by [serue](#) on Fri, 12 Jan 2007 19:22:12 GMT

Quoting Paul Menage (menage@google.com):

> On 1/12/07, Serge E. Hallyn <serue@us.ibm.com> wrote:

> >

> >I agree, so long as "provided requirements aren't too different" is

> >replaced by "provided there is commonality to be merged." Differences in

> >lifetime rules and fs behavior could make it pounding a round peg into

> >a square hole...

>

> Well, at a minimum they have the commonality of "track this set of

> processes, and all their children", and report how the set changes.

>

> >

> >We were thinking that each container directory would have a file

> >representing each namespace in the nsproxy. To enter only a few

> >namespaces out of an existing namespace container, then, you could

> >create a directory for a new namespace container, link the namespaces

> >you want out of other containers, then enter the container (presumably

> >by doing 'echo (container_path) > /proc/\$\$/ns_container')

> >

> >So in some ways that's actually closer to what you currently have

> >than the default container creation rules.

>

> Very similar, yes - the only difference would be that in my model you'd do

>

> echo \$\$ > (container_path)/tasks

>

> But I thought that Eric (and others?) objected to the idea of being

> able to move a task into an existing namespace/container on the

> grounds of race conditions, etc?

It's something we have to be very careful about - that's the reason to require a subsequent exec to make the container change take effect - but not allowing this is pretty much a no-go from vserver and openvz points of view I think.

> If you were able to go with this model rather than the unshare/clone

> model, that would be a lot simpler than implementing a new clone model

> for containers.

We'll still need the clone, but I really don't see that being a big deal. I'll just do it, and see how it goes.

> It would also make your namespace work more

> useful/flexible, I think - there are definitely cases where I want to

> be able to add a new process into an existing

> container/namespace/virtual server, which isn't possible with the

> unshare/clone model unless the root process in the container is

> written to be able to spawn new processes for you.

>

> Paul
