Subject: Re: [PATCH 00/10] Containers(V10): Generic Process Containers
Posted by Paul Menage on Thu, 28 Jun 2007 21:27:25 GMT
View Forum Message <> Reply to Message

On 5/30/07, William Lee Irwin III <wli@holomorphy.com> wrote:
> On Wed, May 30, 2007 at 12:14:55AM -0700, Andrew Morton wrote:
> > So how do we do this?
> > Is there any sneaky way in which we can modify the kernel so that this new
> > code gets exercised more? Obviously, tossing init into some default
> > system-wide container would be a start. But I wonder if we can be
> > sneakier - for example, create a new container on each setuid(), toss the
> > task into that. Or something along those lines?
>
> How about a container for each thread group, pgrp, session, and user?
>

I've been thinking about this, and figured that it could be quite
useful to be able to mount a container tree that groups tasks by
userid or thread group - for doing per-user resource controls, for
example, without having to write a controller that specifically
handles the per-user case.

One option would be to add a mount option, something like

mount -t container -ogroupkey=<X>

where X could be one of: uid, gid, pgrp, sid, tgid

And put hooks in the various places where these ids could change, in
order to move tasks between contaners as appropriate. But after some
thought it seems to me that this is putting complexity in the kernel
that probably doesn't belong there, and additionally is probably not
sufficently flexible for some real-life situations. (E.g. the user
wants all users in the "student" group to be lumped into the same
container, but each user in the "professor" group gets their own
container).

So maybe this would be better handled in userspace? Have a daemon
listing on a process connector socket, and move processes between
containers based on notifications from the connector and user-defined
rules.

We'd probably also want to add some new connector events, such as
PROC_EVENT_PGRP, and PROC_EVENT_SID

A simple daemon that handles the case where we're classifying based on
a single key with no complex rules shouldn't be too hard to write.

It also sounds rather like the classification engine that
ResourceGroups had originally in the kernel and moved to userspace, so
I'll take a look at that and see if it's adaptable for this.

Paul