Subject: Re: [PATCH] Allow signalling container-init
Posted by serue on Thu, 09 Aug 2007 01:21:28 GMT
View Forum Message <> Reply to Message

Quoting Daniel Pittman (daniel@rimspace.net):
> sukadev@us.ibm.com writes:
>
> > Should we include this in the patchset ?
>
> [...]
>
> > Only the global-init process must be special - any other
> > container-init process must be killable to prevent run-away processes
> > in the system.
>
> One problem I hit while using OpenVZ is that some init processes --
> notable upstart used by Ubuntu -- depend on the special signal processing
> extended to init by the kernel.
>
> The problem here was that a signal the kernel would otherwise have
> blocked was sent to upstart, the default handler was invoked and init
> was terminated.
>
> > TODO: Ideally we should allow killing the container-init only from
> > ancestor containers and prevent it being killed from that or
> > descendant containers. But that is a more complex change and
> > will be addressed by a follow-on patch. For now allow the
> > container-init to be terminated by any process with sufficient
> > privileges.
>
> This will break, as far as I can see, by allowing the container root to
> send signals to init that it doesn't expect.

Yes, in the end what we want is for a container init to receive

 1. all signals from a (authorized) process in a parent
    pid namespace.
 2. for signals sent from inside it's pid namespace, only
    exactly those signals for which it has installed a
    custom signal handler, no others.

In other words to a process in an ancestor pid namespace, the init of a
container is like any other process. To a process inside the namespace
for which it is init, it is as /sbin/init is to the system now.

Actually achieving that without affecting performance for all signalers
is nontrivial. The current patchset is complex enough that I'd like to
see us settle on non-optimal semantics for now, and once these patches

have settled implement the ideal signaling.

-serge

_____