Subject: Re: containers development plans
Posted by dev on Thu, 12 Jul 2007 10:32:27 GMT
View Forum Message <> Reply to Message

Paul Menage wrote:
> On 7/2/07, Serge E. Hallyn <serge@hallyn.com> wrote:
>
>> 4. task containers functionality
>
>
> How about if we adopt "process containers" or "task containers" as the
> term for the generic container framework, to distinguish from more
> general user-space containers? In the same way that "task_struct" in
> the kernel is understood to be separate from the concept of a "task"
> in a job scheduling system in userspace.
>
>
>> base features
>
>
> Features that I'd like to see in the short and medum term:
>
> - support for virtualized containerfs mounts, so that virtual servers
> can mount their own containerfs and manage sub-containers
>
> - automatically prefixing control file names with the subsystem name,
> unless changed or disabled by the user at mount time
>
> - removing unnecessary locking where possible.
>
> - simplifying the control file API
>
> - a userspace RBCE along with simple configuration so that you can
> easily use generic containers to apply subsystem controls on a
> per-user, per-group, per-pgrp, per-executable, etc, basis. (E.g. to
> easily apply CFS to be fair between pgrps rather than fair between
> processes)
>
>
>> specific containers
>> poll to see who has plans
>
>
> Some possible subsystems that I'm thinking of include:
>
> - splitting the memory and cpu isolation parts of cpusets into two
> separate subsystems (still backwards-compatible)
>

> - some kind of network connect/bind/accept controller. Eric came up
> with a nice way of doing this by adding iptables hooks for
> connect/bind/accept, and then adding an iptables match module that
> could match based on container id. This would give us all the
> flexibility of iptables and the existing iptables tools. The drawback
> is that it could be rather tricky to virtualize. A less flexible
> solution that just allowed you to specify permitted
> local-port-range/remote-port-range/remote-netmask tuples would be more
> virtualizable, even if it doesn't make as much reuse of existing
> iptables support.

Not sure why it requires some additional controller, but surely
it is possible to create a match for iptables matching container ID.
Is it what you are thinkinh about or I got something wrong?

Thanks,
Kirill
