Subject: Re: [ckrm-tech] [PATCH 0/7] containers (V7): Generic Process Containers
Posted by Paul Menage on Tue, 13 Feb 2007 00:42:15 GMT
View Forum Message <> Reply to Message

On 2/12/07, Sam Vilain <sam@vilain.net> wrote:
> Ask yourself this - what do you need the container structure for so
> badly, that virtualising the individual resources does not provide for?

Primarily, that otherwise every module that wants to affect/monitor behaviour of a group of associated processes has to implement its own process grouping abstraction.

As an example, the CPU accounting patch that in included in my patch set as an illustration of a simple resource monitoring module is just 250 lines, almost entirely in one file; if it also had to handle associating tasks together into groups and presenting a filesystem interface to the user it would be far larger and would have a much bigger footprint on the kernel.

>From the point of view of the virtual server containers, the advantage is that you're integrated with a standard filesystem interface for determining group membership. It does become simpler to combine virtual servers and resource controllers, although I grant you that you could juggle that from userspace without the additional kernel support.

Paul

_____
Containers mailing list
Containers@lists.osdl.org
https://lists.osdl.org/mailman/listinfo/containers

Subject: Re: [ckrm-tech] [PATCH 0/7] containers (V7): Generic Process Containers
Posted by Sam Vilain on Tue, 13 Feb 2007 01:13:07 GMT
View Forum Message <> Reply to Message

Paul Menage wrote:
>> Ask yourself this - what do you need the container structure for so
>> badly, that virtualising the individual resources does not provide for?
>>
> Primarily, that otherwise every module that wants to affect/monitor
> behaviour of a group of associated processes has to implement its own
> process grouping abstraction.
>

Not every module, you just make them on sensible, planned groupings. The danger is that the "container" group becomes a fallback grouping for

things when people can't be bothered thinking about it properly, and
everything including the kitchen sink gets thrown in. Then later you
find a real use case where you don't want them together, but it's too
late because it's already a part of the official API.

> As an example, the CPU accounting patch that in included in my patch
> set as an illustration of a simple resource monitoring module is just
> 250 lines, almost entirely in one file; if it also had to handle
> associating tasks together into groups and presenting a filesystem
> interface to the user it would be far larger and would have a much
> bigger footprint on the kernel.
>

It's also less flexible. What if I want to do CPU accounting on some
other boundaries than the "virtual server" a process is a part of?

> From the point of view of the virtual server containers, the advantage
> is that you're integrated with a standard filesystem interface for
> determining group membership. It does become simpler to combine
> virtual servers and resource controllers, although I grant you that
> you could juggle that from userspace without the additional kernel
> support.
>

I'm not disagreeing it's a pragmatic shortcut that has been successful
for a number of projects including vserver which I use every day. But
it reduces "synergy" by excluding the people working with virtualisation
in ways that don't fit its model.

Yes, there should be a similarity in the way that you manage namespaces
and it should be easy to develop new namespaces without constantly
re-inventing the wheel. But why does that imply making binding
decisions about the nature of how you can virtualise? IMHO those
decisions should be made on a per-subsystem basis.

Sam.

_____
Containers mailing list
Containers@lists.osdl.org
https://lists.osdl.org/mailman/listinfo/containers