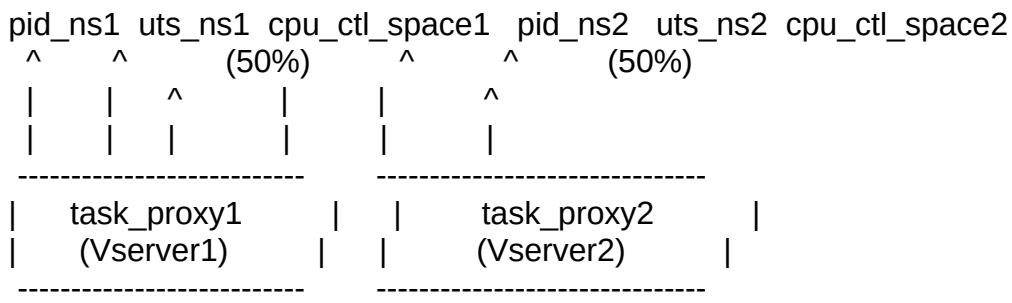
Subject: Re: Summary of resource management discussion
Posted by Srivatsa Vaddagiri on Thu, 15 Mar 2007 17:04:35 GMT
View Forum Message <> Reply to Message

On Thu, Mar 15, 2007 at 04:24:37AM -0700, Paul Menage wrote:
> If there really was a grouping that was always guaranteed to match the
> way you wanted to group tasks for e.g. resource control, then yes, it
> would be great to use it. But I don't see an obvious candidate. The
> pid namespace is not it, IMO.

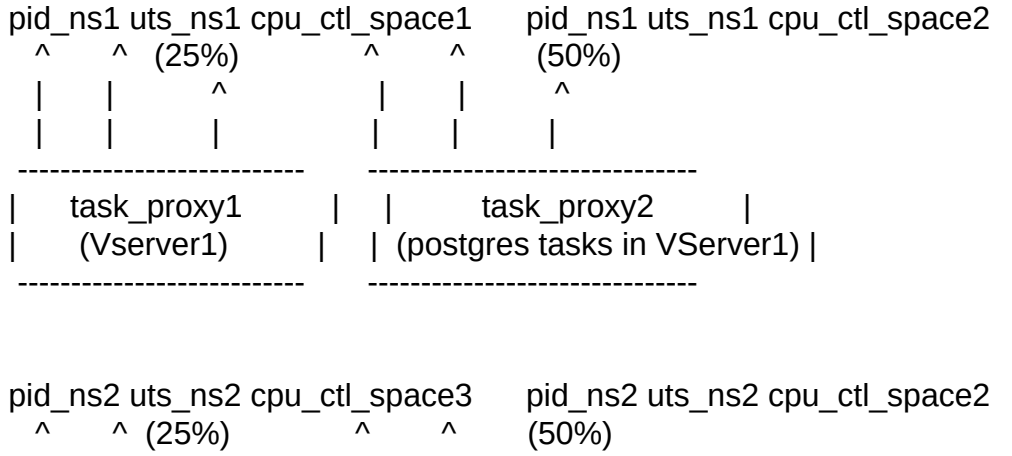
In vserver context, what is the "normal" case then? Atleast for Linux Vserver pid namespace seems to be normal unit of resource control (as per Herbert).

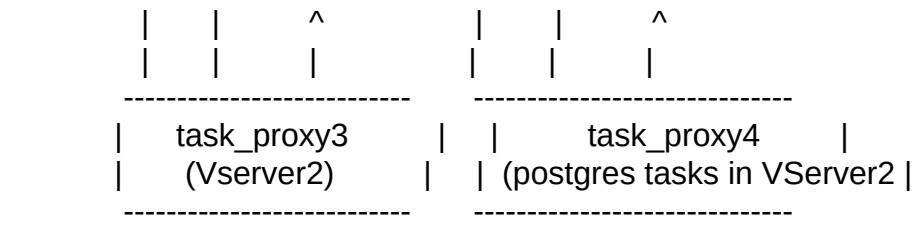
Even if one wanted to manage a arbitrary group of tasks in vserver context, IMHO its still possible to construct that arbitrary group using the existing pointer, ns[/task]proxy, and not break existing namespace semantics/functionality.

So the normal case I see is:



But, if someone wanted to manage cpu resource differently, and say that postgres tasks from both vservers should be in same cpu resource class, the above becomes:





(the best I could draw using ASCII art!)

The benefit I see of this approach is it will avoid introduction of additional pointers in struct task_struct and also additional structures (struct container etc) in the kernel, but we will still be able to retain same user interfaces you had in your patches.

Do you see any drawbacks of doing like this? What will break if we do this?

> Resource control (and other kinds of task grouping behaviour) shouldn't
> require virtualization.

Certainly. AFAICS, nsproxy[.c] is unconditionally available in the kernel (even if virtualization support is not enabled). When reused for pure resource control purpose, I see that as a special case of virtualization where only resources are virtualized and namespaces are not.

I think an interesting question would be : what more task-grouping behavior do you want to implement using an additional pointer that you can't reusing ->task_proxy?

> >a. Paul Menage's patches:
> >
> > (tsk->containers->container[cpu_ctlr.subsys_id] - X)->cpu_limit
>
> So what's the '-X' that you're referring to

Oh ..that's to seek pointer to begining of the cpulimit structure (subsys pointer in 'struct container' points to a structure embedded in a larger structure. -X gets you to point to the larger structure).

> >6. As tasks move around namespaces/resource-classes, their
> > tsk->nsproxy/containers object will change. Do we simple create
> > a new nsproxy/containers object or optimize storage by searching
> > for one which matches the task's new requirements?
>
> I think the latter.

Yes me too. But maybe to keep in simple in initial versions, we should avoid that optimisation and at the same time get statistics on duplicates?.

--
Regards,
vatsa

_____

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