Subject: Re: [PATCH 0/2] resource control file system - aka containers on top of nsproxy!
Posted by Herbert Poetzl on Sat, 03 Mar 2007 17:32:44 GMT
View Forum Message <> Reply to Message

On Sat, Mar 03, 2007 at 03:06:55PM +0530, Srivatsa Vaddagiri wrote:
> On Thu, Mar 01, 2007 at 11:39:00AM -0800, Paul Jackson wrote:
> > vatsa wrote:
> > > I suspect we can make cpusets also work
> > > on top of this very easily.
> >
> > I'm skeptical, and kinda worried.
> >
> > ... can you show me the code that does this?
>
> In essense, the rcfs patch is same as the original containers
> patch. Instead of using task->containers->container[cpuset->hierarchy]
> to get to the cpuset structure for a task, it uses
> task->nsproxy->ctlr_data[cpuset->subsys_id].
>
> So if the original containers patches could implement cpusets on
> containers abstraction, I don't see why it is not possible to implement
> on top of nsproxy (which is essentialy same as container_group in Paul
> Menage's patches). Any way code speaks best and I will try to post
> something soon!
>
> > Namespaces are not the same thing as actual resources
> > (memory, cpu cycles, ...).  Namespaces are fluid mappings;
> > Resources are scarce commodities.
>
> Yes, perhaps this overloads nsproxy more than what it was intended for.
> But, then if we have to to support resource management of each
> container/vserver (or whatever group is represented by nsproxy),
> then nsproxy seems the best place to store this resource control
> information for a container.

well, the thing is, as nsproxy is working now, you
will get a new one (with a changed subset of entries)
every time a task does a clone() with one of the
space flags set, which means, that you will end up
with quite a lot of them, but resource limits have
to address a group of them, not a single nsproxy
(or act in a deeply hierarchical way which is not
there atm, and probably will never be, as it simply
adds too much overhead)

> > I'm wagering you'll break either the semantics, and/or the
> > performance, of cpusets doing this.

>
> It should have the same perf overhead as the original
> container patches (basically a double dereference -
> task->containers/nsproxy->cpuset - required to get to the
> cpuset from a task).

on every limit accounting or check? I think that
is quite a lot of overhead ...

best,
Herbert

> Regarding semantics, can you be more specific?
>
> In fact I think it will facilitate containers to use cpusets more
> easily. You can for example divide the system into two (exclusive)
> cpusets A and B, and have container C1 work inside A while C2 uses C2.
> So c1's nsproxy->cpuset will point to A will c2's nsproxy->cpuset will
> point to B. If you dont want to split the cpus into cpusets like that,
> then all nsproxy's->cpuset will point to the top_cpuset.
>
> Basically the rcfs patches demonstrate that is possible to keep track
> of hierarchial relationship in resource objects using corresponding
> file system objects itself (like dentries). Also if we are hooked to
> nsproxy, lot of hard work to mainain life-time of nsproxy's (ref count
> ) is already in place -

> we just reuse that work. These should help us avoid the container
> structure abstraction in Paul Menage's patches (which was the main
> point of objection from last time).
>
> --
> Regards,
> vatsa
> _______________________________________________
> Containers mailing list
> Containers@lists.osdl.org
> https://lists.osdl.org/mailman/listinfo/containers
_______________________________________________
Containers mailing list
Containers@lists.osdl.org
https://lists.osdl.org/mailman/listinfo/containers