Subject: Re: Loadable cgroup subsystems
Posted by Paul Menage on Tue, 08 Apr 2008 09:41:16 GMT
View Forum Message <> Reply to Message

On Tue, Apr 8, 2008 at 2:40 AM, Nikanth Karthikesan <knikanth@suse.de> wrote:
> >
> > I'd rather not add support for this without a strong case of a
> > subsystem that really needs to be dynamically loaded.
>
> There were some band-width control patches based on cfq + cgroups, which
> I guess will mandate cfq to be built-in?
>

Yes, or else have built-in stubs for the cgroup subsystem that load
cfq and the code that uses cfq the first time someone tries to mount
that subsystem.

Actually, it probably wouldn't be too hard to have cgroups do that
automatically - support the concept of a stub cgroup that was known
about at compile time but wasn't active until its first bind, and have
cgroups dynamically load the relevant modules when the user tried to
mount it.

Dynamically loading subsystems that aren't even known about at compile
time would be probably a bit uglier.

Paul

_____
Containers mailing list
Containers@lists.linux-foundation.org
https://lists.linux-foundation.org/mailman/listinfo/containers

Subject: RE: Loadable cgroup subsystems
Posted by Satoshi UCHIDA on Wed, 09 Apr 2008 08:31:58 GMT
View Forum Message <> Reply to Message

> > There were some band-width control patches based on cfq + cgroups, which
> > I guess will mandate cfq to be built-in?
> >
>
> Yes, or else have built-in stubs for the cgroup subsystem that load
> cfq and the code that uses cfq the first time someone tries to mount
> that subsystem.
>

I think that it is too early to decide which band-width control for
cgroups will $B!! (Bmandate to be built-in.

Before that, we need more discussion about a band-width control and
its $B!! (Bimplementation.
I seems that both implementation are still prototype and immature.
Both implementation would have good points and bad points respectively.

If you want to manage these patchset by one source tree, we should
choice adopting means by a build option.
So, I think that a build option is the better at present, but I guess
that it should be better to dynamically load subsystem in future.

I have idea that handles multiple implementation as provisional solution.
Now, all bandwidth control expands CFQ original code and is valid by build options.
It is good to be re-implemented as new I/O scheduler which reuses original code.
Therefore, it became a selectable such as other I/O scheduler (anticipatory, cfq, noop, etc.)
User can select a suitable scheduler through "scheduler" entry in sysfs.
In addition, it's possible to use a new controller simultaneously with existing I/O scheduler.
Unfortunately, this idea would require much modification to each patchset.

Satoshi UCHIDA