Subject: Re: [patch00/05]: Containers(V2)- Introduction
Posted by Rohit Seth on Wed, 20 Sep 2006 23:22:22 GMT
View Forum Message <> Reply to Message

On Wed, 2006-09-20 at 15:51 -0700, Paul Jackson wrote:
> Seth wrote:
> > But am not sure
> > if this number of nodes can change dynamically on the running machine or
> > a reboot is required to change the number of nodes.
>
> The current numa=fake=N kernel command line option is just boottime,
> and just x86_64.
>

Ah okay.

> I presume we'd have to remove these two constraints for this to be
> generally usable to containerize memory.
>

Right.

> We also, in my current opinion, need to fix up the node_distance
> between such fake numa sibling nodes, to correctly reflect that they
> are on the same real node (LOCAL_DISTANCE).
>
> And some non-trivial, arch-specific, zonelist sorting and reconstruction
> work will be needed.
>
> And an API devised for the above mentioned dynamic changing.
>
> And this will push on the memory hotplug/unplug technology.
>

Yes, if we use the existing notion of nodes for other purposes then you
have captured the right set of changes that will be needed to make that
happen. Such changes are not required for container patches as such.

> All in all, it could avoid anything more than trivial changes to the
> existing memory allocation code hot paths. But the infrastructure
> needed for managing this mechanism needs some non-trivial work.
>
>
> > Though when you want to have in access of 100 containers then the cpuset
> > function starts popping up on the oprofile chart very aggressively.
>
> As the linux-mm discussion last weekend examined in detail, we can
> eliminate this performance speed bump, probably by caching the

> last zone on which we found some memory. The linear search that was
> implicit in __alloc_pages()'s use of zonelists for many years finally
> become explicit with this new usage pattern.
>

Okay.

>
> > Containers also provide a mechanism to move files to containers. Any
> > further references to this file come from the same container rather than
> > the container which is bringing in a new page.
>
> I haven't read these patches enough to quite make sense of this, but I
> suspect that this is not a distinction between cpusets and these
> containers, for the basic reason that cpusets doesn't need to 'move'
> a file's references because it has no clue what such are.
>

But container support will allow the certain files pages to come from
the same container irrespective of who is using them. Something useful
for shared libs etc.

-rohit

>
> > In future there will be more handlers like CPU and disk that can be
> > easily embeded into this container infrastructure.
>
> This may be a deciding point.
>