Subject: Re: [ckrm-tech] [RFC][PATCH] UBC: user resource beancounters
Posted by Rohit Seth on Thu, 24 Aug 2006 01:44:41 GMT
View Forum Message <> Reply to Message

On Tue, 2006-08-22 at 11:55 -0700, Chandra Seetharaman wrote:
> On Mon, 2006-08-21 at 18:45 -0700, Rohit Seth wrote:
> >
> > > > this is not 100% true.
> > > > UBC itself doesn't prevent from changing context on the fly.
> > > > But since this leads to part of resources to be charged to
> > > > one UBC and another part to another UBC and so long and so
> > >
> > > Let the controllers and the users worry about that part.
> > >
> >
> > I think as the tasks move around, it becomes very heavy to move all the
> > pages belonging to previous container to a new container.
>
> Not for all resources, CPU could handle it very nicely, whereas memory
> would be heavy. My point is that the infrastructure should be open, and
> controller is the one that decides whether it wants to handle it or not.

With open you are implying being able to use different ones. It would
be nice to get one in and make sure it is stable and optimized...

>
> >
> > > > As I mentioned UBC might be perfect for container resource management,
> > > > but what I am talking for is resource management _without_ a container.
> > >
> >
> > Can you explain that part a bit more?
>
> Basically I was saying that even though resource management in container
> and non-container have mostly same requirements, there are few
> requirements that are critical in non-container scenario which are non-
> issue in container scenario (for example, moving tasks from one resource
> group to another).
>
> In effect, Design of the infrastructure should not limit non-container
> usages.
>
> IMO, non-container requirements are a superset of container requirements
> (resource management purposes only :).
>

hmm, non-container world (and its resource management part) already
exist. And sure those requirements are superset of this discussion.

And hopefully container support will not break/modify that much.

> >
> > > >
> > > > > - No ability to maintain resource specific data in the controller.
> > > > > it's false. fields can be added to user_beancounter struct easily.
> > > > > and that's what our controllers do.
> > > >
> > > > With the model of static array for resources (struct ubparm ub_parms
> > > > [UB_RESOURCES] in struct user_beancounter), it is not be possible to
> > > > attach _different_ "controller specific" information to each of the
> > > > entries.
> > > >
> > > > I do not think it is good idea to add controller specific information of
> > > > _different_ controllers to the user_beancounter. Think of all the fields
> > > > it will have when all the numproc controller needs is just the basic 3-4
> > > > fields.
> > > >
> > >
> > > IMO it is okay to add the fields whenever necessary as Kirill
> > > suggested. I think once the container subject gets baked a little more,
> > > the controllers will also get tightly coupled.
> >
> > I think my point is not understood. I do not think it is right to add
> > _controller specific_ fields to the generic data structure (struct
> > user_beancounter), as we will end up with a generic data structure which
> > will have so many fields that are not used in so many controllers.
> >

A single centralized structure that has fields that are mostly used by
every one should be okay I think.

> >
> > > > >
> > > > > - No ability to get the list of tasks belonging to a UBC.
> > > > > it is not true. it can be read from /proc or system calls interface,
> > > > > just like the way one finds all tasks belonging to one user :)
> > > > >
> > > > > BTW, what is so valueable in this feature?
> > > >
> > > > Again, it may not be useful for container type usages (you can probably
> > > > get the list from somewhere else, but for resource management it is
> > > > useful for sysadmins).
> > > >
> > >
> > > I'm also debating about whether printing task information is really any
> > > useful. If a sysadm wants to get information about any particular task
> > > then that can come from /proc/<pid>/container Though container list

> > will be one place where one can easily get the list of all the contained
> > tasks (and other resources like files).
>
> In non-container environment, there is _no_ /proc/pid/container, as
> there is no concept of container :). This will be useful for non-
> container scenario.
>

I'm sure when container support gets in then for the above scenario it
will read -1 ...

-rohit
