Subject: Re: [ckrm-tech] [RFC][PATCH] UBC: user resource beancounters
Posted by Chandra Seetharaman on Mon, 21 Aug 2006 21:45:21 GMT
View Forum Message <> Reply to Message

On Mon, 2006-08-21 at 17:24 +0400, Kirill Korotaev wrote:
> Chandra Seetharaman wrote:
> > Kirill,
> >
> > Here are some concerns I have (as of now) w.r.t using UBC for resource
> > management (in the context of resource groups).
> >
> > - guarantee support is missing. I do not see any code to provide the
> >   minimum amount of resource a group can get. It is important for
> >   providing QoS. (In a different email you did mention guarantee, i am
> >   referring it here for completeness).
> I mentioned a couple of times that this is a limited core functionality
> in this patch set.
> guarantees are implementable as a separate UBC parameters.

I will wait for oomguarpages patches :)

>
> > - Creation of a UBC and assignment of task to a UBC always happen in
> >   the context of the task that is affected. I can understand it works in
> >   OpenVZ environment, but IMO has issues if one wants it to be used for
> >   basic resource management
> >   - application needs to be changed to use this feature.
> >   - System administrator does not have the control to assign tasks to a
> >   UBC. Application does by itself.
> >   - Assignment of task to a UBC need to be transparent to the
> >   application.
> 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.

As I mentioned UBC might be perfect for container resource management,
but what I am talking for is resource management _without_ a container.

> forth, we believe that more clear and correct interface is
> something like fork()/exec()-required-application.
>
> So you can always execute new applications in desired UB and
> NO application modification are required.

For generic workload management/resource management _desired UB_ is not

necessarily decided at fork/exec time. It can happen anytime during the
life cycle of a task.

>
>
> > - UBC is deleted when the last task (in that UBC) exits. For resource
> > management purposes, UBC should be deleted only when the administrator
> > deletes it.
> 1. UBCs are freed when last _resource_ using it puts the last reference.
> not the task. And it is a BIG error IMHO to think that resource
> management should group tasks. No, it should group _objects_. Tasks
> are just the same objects like say sockets.

No argument there, that is how CKRM was early last year. But, I do not
see how is related to the point I am making above (" UBC should be
deleted only when the administrator deletes it").

> 2. this is easily changeable. You are the only who requested it so far.

It may be because I am the only one looking at it without the
"container" goggles on :).

> 3. kernel does so for many other objects like users and no one complains :)
>
> > - No ability to set resource specific configuration information.
> UBC model allows to _limit_ users. It is _core_.

I think you got me wrong here. What I want is the ability to
set/maintain a generic controller specific information.

For example, if the CPU controller wants to allow the user to set the
number of seconds over which the user wants the guarantee/limit to be
imposed.

> We want to do resource management step by step and send it patch by patch,
> while you are trying to solve everything at once.
>
> sys_open() for example doesn't allow to open sockets, does it?
> the same for UBC. They do what they are supposed to do.

I do not see how this relates !!

>
> > - 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.

>
> > - 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).

> do you want to have interfaces to find kernel structures and even pages
> which belong to the container? tasks are just one type of objects...
>
> > - Doesn't inform the resource controllers when limits(shares) change.
> As was answered and noted by Alan Cox:
> 1. no one defined what type of action should be done when limits change

let the controller decide it.
> 2. it is extendable _when_ needed. Do you want to introduce hooks just
> to have them?
> 3. is it so BIG obstacle for UBC patch? These 3-lines hooks code which
> is not used?
>
> > - Doesn't inform the resource controllers when a task's UBC has changed.
> the same as above. we don't add functionality which is not used YET
> (and no one even knows HOW).
>
> > - Doesn't recalculate the resource usage when a task's UBC has changed.
> > i.e doesn't uncharge the old UBC and charge new UBC.
> You probably missed my explanation, that most
> resources (except for the simplest one - numproc) can't be recharged
> easily. And nothing in UBC code prevents such recharge to be added later
> if requested.

My point is that controllers should have this control. I am ok with these being added later. Wondering if there is any design limitations that would prevent the later additions (like the _controller specific data above).

>
> > - For a system administrator name for identification of a UBC is
> > better than a number (uid).
> Have you any problems with pids, uids, gids and signals?

Again, in container land each UB is attached with a container hence no
issue.

In a non-container situation IMO it will be easier to manage/associate
"gold", "silver", "bronze", "plastic" groups than 0, 11, 83 and 113.

> It is a question of interface. I don't mind in changing UBC interface even
> to configfs if someone really wants it.
>
> Thanks,
> Kirill
>
>
--

---------- ----------
Chandra Seetharaman | Be careful what you choose....
- sekharan@us.ibm.com | .....you may get it.
---------- ----------