Subject: Re: [ckrm-tech] [PATCH 0/2] resource control file system - aka containers on top of nsproxy!
Posted by Srivatsa Vaddagiri on Tue, 06 Mar 2007 10:39:40 GMT
View Forum Message <> Reply to Message

On Mon, Mar 05, 2007 at 07:39:37PM +0100, Herbert Poetzl wrote:
> > Thats why nsproxy has pointers to resource control objects, rather
> > than embedding resource control information in nsproxy itself.
>
> which makes it a (name)space, no?

I tend to agree, yes!

> > This will let different nsproxy structures share the same resource
> > control objects (ctlr_data) and thus be governed by the same
> > parameters.
>
> as it is currently done for vfs, uts, ipc and soon
> pid and network l2/l3, yes?

yes (by vfs do you mean mnt_ns?)

> > Where else do you think the resource control information for a
> > container should be stored?
>
> an alternative for that is to keep the resource
> stuff as part of a 'context' structure, and keep
> a reference from the task to that (one less
> indirection, as we had for vfs before)

something like:

```
struct resource_context {
  int cpu_limit;
  int rss_limit;
  /* all other limits here */
}

struct task_struct {
  ...
  struct resource_context *rc;

}
```

?

With this approach, it makes it hard to have task-grouping that are unique to each resource.

For ex: lets say that CPU and Memory needs to be divided as follows:

CPU : C1 (70%), C2 (30%)
Mem : M1 (60%), M2 (40%)

Tasks T1, T2, T3, T4 are assigned to these resource classes as follows:

C1 : T1, T3
C2 : T2, T4
M1 : T1, T4
M2 : T2, T3

We had a lengthy discussion on this requirement here:

http://lkml.org/lkml/2006/11/6/95
http://lkml.org/lkml/2006/11/1/239

Linus also has expressed a similar view here:

http://lwn.net/Articles/94573/

Paul Menage's (and its clone rcfs) patches allows this flexibility by simply
mounting different hierarchies:

mount -t container -o cpu none /dev/cpu
mount -t container -o mem none /dev/mem

The task-groups created under /dev/cpu can be completely independent of
task-groups created under /dev/mem.

Lumping together all resource parameters in one struct (like
resource_context above) makes it difficult to provide this feature.

Now can we live w/o this flexibility? Maybe, I don't know for sure.
Since (stability of) user-interface is in question, we need to take a
carefull decision here.

> > then other derefences (->ctlr_data[] and ->limit) should be fast, as
> > they should be in the cache?
>
> please provide real world numbers from testing ...

What kind of testing did you have in mind?

--
Regards,

vatsa

_____

Containers mailing list
Containers@lists.osdl.org
https://lists.osdl.org/mailman/listinfo/containers