Subject: Re: [PATCH v3 04/13] kmem accounting basic infrastructure
Posted by Glauber Costa on Wed, 26 Sep 2012 20:02:14 GMT
View Forum Message <> Reply to Message

On 09/26/2012 11:56 PM, Tejun Heo wrote:
> Hello,
>
> On Wed, Sep 26, 2012 at 11:46:37PM +0400, Glauber Costa wrote:
>> Besides not being part of cgroup core, and respecting very much both
>> cgroups' and basic sanity properties, kmem is an actual feature that
>> some people want, and some people don't. There is no reason to believe
>> that applications that want will live in the same environment with ones
>> that don't want.
>
> I don't know. It definitely is less crazy than .use_hierarchy but I
> wouldn't say it's an inherently different thing. I mean, what does it
> even mean to have u+k limit on one subtree and not on another branch?
> And we worry about things like what if parent doesn't enable it but
> its chlidren do.
>

It is inherently different. To begin with, it actually contemplates two
use cases. It is not a work around.

The meaning is also very well defined. The meaning of having this
enabled in one subtree and not in other is: Subtree A wants to track
kernel memory. Subtree B does not. It's that, and never more than that.
There is no maybes and no buts, no magic knobs that makes it behave in a
crazy way.

If a children enables it but the parent does not, this does what every
tree does: enable it from that point downwards.

> This is a feature which adds complexity. If the feature is necessary
> and justified, sure. If not, let's please not and let's err on the
> side of conservativeness. We can always add it later but the other
> direction is much harder.
>

I disagree. Having kmem tracking adds complexity. Having to cope with
the use case where we turn it on dynamically to cope with the "user page
only" use case adds complexity. But I see no significant complexity
being added by having it per subtree. Really.

You have the use_hierarchy fiasco in mind, and I do understand that you
are raising the flag and all that.

But think in terms of functionality: This thing here is a lot more

similar to swap than use_hierarchy. Would you argue that memsw should be per-root ?

The reason why it shouldn't: Some people want to limit memory consumption all the way to the swap, some people don't. Same with kmem.