Subject: Re: [RFC][PATCH 4/7] RSS accounting hooks over the code
Posted by dev on Wed, 14 Mar 2007 16:16:26 GMT
View Forum Message <> Reply to Message

Nick,

>>Accounting becomes easy if we have a container pointer in struct page.
>> This can form base ground for building controllers since any memory
>>related controller would be interested in tracking pages. However we
>>still want to evaluate if we can build them without bloating the
>>struct page. Pagecache controller (2) we can implement with container
>>pointer in struct page or container pointer in struct address space.
>
>
> The thing is, you have to worry about actually getting anything in the
> kernel rather than trying to do fancy stuff.
>
> The approaches I have seen that don't have a struct page pointer, do
> intrusive things like try to put hooks everywhere throughout the kernel
> where a userspace task can cause an allocation (and of course end up
> missing many, so they aren't secure anyway)... and basically just
> nasty stuff that will never get merged.

User beancounters patch has got through all these...
The approach where each charged object has a pointer to the owner container,
who has charged it - is the most easy/clean way to handle
all the problems with dynamic context change, races, etc.
and 1 pointer in page struct is just 0.1% overehad.

> Struct page overhead really isn't bad. Sure, nobody who doesn't use
> containers will want to turn it on, but unless you're using a big PAE
> system you're actually unlikely to notice.

big PAE doesn't make any difference IMHO
(until struct pages are not created for non-present physical memory areas)

> But again, I'll say the node-container approach of course does avoid
> this nicely (because we already can get the node from the page). So
> definitely that approach needs to be discredited before going with this
> one.

But it lacks some other features:
1. page can't be shared easily with another container
2. shared page can't be accounted honestly to containers
   as fraction=PAGE_SIZE/containers-using-it
3. It doesn't help accounting of kernel memory structures.
   e.g. in OpenVZ we use exactly the same pointer on the page
   to track which container owns it, e.g. pages used for page

tables are accounted this way.
4. I guess container destroy requires destroy of memory zone,
   which means write out of dirty data. Which doesn't sound
   good for me as well.
5. memory reclamation in case of global memory shortage
   becomes a tricky/unfair task.
6. You cannot overcommit. AFAIU, the memory should be granted
   to node exclusive usage and cannot be used by by another containers,
   even if it is unused. This is not an option for us.

>>Building on this patchset is much simple and and we hope the bloat in
>>struct page will be compensated by the benefits in memory controllers
>>in terms of performance and simplicity.
>>
>>Adding too many controllers and accounting parameters to start with
>>will make the patch too big and complex. As Balbir mentioned, we have
>>a plan and we shall add new control parameters in stages.
>
> Everyone seems to have a plan ;) I don't read the containers list...
> does everyone still have *different* plans, or is any sort of consensus
> being reached?

hope we'll have it soon :)

Thanks,
Kirill

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