
Subject: Re: [PATCH v2 04/11] kmem accounting basic infrastructure
Posted by [Ying Han](#) on Wed, 15 Aug 2012 18:01:41 GMT
[View Forum Message](#) <> [Reply to Message](#)

On Wed, Aug 15, 2012 at 5:39 AM, Michal Hocko <mhocko@suse.cz> wrote:

> On Wed 15-08-12 13:33:55, Glauber Costa wrote:

> [...]

>> > This can

>> > be quite confusing. I am still not sure whether we should mix the two

>> > things together. If somebody wants to limit the kernel memory he has to

>> > touch the other limit anyway. Do you have a strong reason to mix the

>> > user and kernel counters?

>>

>> This is funny, because the first opposition I found to this work was

>> "Why would anyone want to limit it separately?" =p

>>

>> It seems that a quite common use case is to have a container with a

>> unified view of "memory" that it can use the way he likes, be it with

>> kernel memory, or user memory. I believe those people would be happy to

>> just silently account kernel memory to user memory, or at the most have

>> a switch to enable it.

>>

>> What gets clear from this back and forth, is that there are people

>> interested in both use cases.

>

> I am still not 100% sure myself. It is just clear that the reclaim would

> need some work in order to do accounting like this.

>

>> > My impression was that kernel allocation should simply fail while user

>> > allocations might reclaim as well. Why should we reclaim just because of

>> > the kernel allocation (which is unreclaimable from hard limit reclaim

>> > point of view)?

>>

>> That is not what the kernel does, in general. We assume that if he wants

>> that memory and we can serve it, we should. Also, not all kernel memory

>> is unreclaimable. We can shrink the slabs, for instance. Ying Han

>> claims she has patches for that already...

>

> Are those patches somewhere around?

Yes, I am working on it to post it sometime *this week*. My last rebase is based on v3.3 and now I am trying to get it rebased to github-memcg. The patch itself has a functional dependency on kernel slab accounting, and I am trying to get that rebased on Glauber's tree but has some difficulty now. What I am planning to do is post the RFC w/ only complied version by far.

The patch handles dentry cache shrinker only at this moment. That is

what we discussed last time as well, where dentry contributes most of the reclaimable objects. (it pins inode, so we leave inode behind)

--Ying

>

> [...]

>> > This doesn't check for the hierarchy so kmem_accounted might not be in

>> > sync with its parents. mem_cgroup_create (below) needs to copy

>> > kmem_accounted down from the parent and the above needs to check if this

>> > is a similar dance like mem_cgroup_oom_control_write.

>> >

>>

>> I don't see why we have to.

>>

>> I believe in a A/B/C hierarchy, C should be perfectly able to set a

>> different limit than its parents. Note that this is not a boolean.

>

> Ohh, I wasn't clear enough. I am not against setting the _limit_ I just

> meant that the kmem_accounted should be consistent within the hierarchy.

>

> --

> Michal Hocko

> SUSE Labs

>

> --

> To unsubscribe, send a message with 'unsubscribe linux-mm' in

> the body to majordomo@kvack.org. For more info on Linux MM,

> see: <http://www.linux-mm.org/> .

> Don't email: <[a href=mailto:"dont@kvack.org"> email@kvack.org](mailto:dont@kvack.org)