Subject: Re: [PATCH 17/23] kmem controller charge/uncharge infrastructure
Posted by Glauber Costa on Wed, 25 Apr 2012 14:43:28 GMT
View Forum Message <> Reply to Message

On 04/24/2012 07:54 PM, David Rientjes wrote:
> On Tue, 24 Apr 2012, Glauber Costa wrote:
>
>>> Yes, for user memory, I see charging to p->mm->owner as allowing that
>>> process to eventually move and be charged to a different memcg and there's
>>> no way to do proper accounting if the charge is split amongst different
>>> memcgs because of thread membership to a set of memcgs. This is
>>> consistent with charges for shared memory being moved when a thread
>>> mapping it moves to a new memcg, as well.
>>
>> But that's the problem.
>>
>> When we are dealing with kernel memory, we are allocating a whole slab page.
>> It is essentially impossible to track, given a page, which task allocated
>> which object.
>>
>
> Right, so you have to make the distinction that slab charges cannot be
> migrated by memory.move_charge_at_immigrate (and it's not even specified
> to do anything beyond user pages in Documentation/cgroups/memory.txt),

Never intended to.

> but
> it would be consistent to charge the same memcg for a process's slab
> allocations as the process's user allocations.
>
> My response was why we shouldn't be charging user pages to
> mem_cgroup_from_task(current) rather than
> mem_cgroup_from_task(current->mm->owner) which is what is currently
> implemented.

Ah, all right. Well, for user memory I agree with you. My point was
exactly that user memory can always be pinpointed to a specific address
space, while kernel memory can't.

>
> If that can't be changed so that we can still migrate user memory amongst
> memcgs for memory.move_charge_at_immigrate, then it seems consistent to
> have all allocations done by a task to be charged to the same memcg.
> Hence, I suggested current->mm->owner for slab charging as well.

All right. This can be done. Although I don't see this as a must for
slab as already explained, I certainly don't oppose doing so as well.