Subject: Re: [PATCH v3 13/28] slub: create duplicate cache
Posted by Glauber Costa on Tue, 29 May 2012 20:25:36 GMT
View Forum Message <> Reply to Message

On 05/30/2012 12:21 AM, Christoph Lameter wrote:
> On Wed, 30 May 2012, Glauber Costa wrote:
>
>> Well, I'd have to dive in the code a bit more, but that the impression that
>> the documentation gives me, by saying:
>>
>> "Cpusets constrain the CPU and Memory placement of tasks to only
>> the resources within a task's current cpuset."
>>
>> is that you can't allocate from a node outside that set. Is this correct?
>
> Basically yes but there are exceptions (like slab queues etc). Look at the
> hardwall stuff too that allows more exceptions for kernel allocations to
> use memory from other nodes.
>
>> So extrapolating this to memcg, the situation is as follows:
>>
>> * You can't use more memory than what you are assigned to.
>> * In order to do that, you need to account the memory you are using
>> * and to account the memory you are using, all objects in the page
>>   must belong to you.
>
> Cpusets work at the page boundary and they do not have the requirement you
> are mentioning of all objects in the page having to belong to a certain
> cpusets. Let that go and things become much easier.
>
>> With a predictable enough workload, this is a recipe for working around the
>> very protection we need to establish: one can DoS a physical box full of
>> containers, by always allocating in someone else's pages, and pinning kernel
>> memory down. Never releasing it, so the shrinkers are useless.
>
> Sure you can construct hyperthetical cases like that. But then that is
> true already of other container like logic in the kernel already.
>
>> So I still believe that if a page is allocated to a cgroup, all the objects in
>> there belong to it - unless of course the sharing actually means something -
>> and identifying this is just too complicated.
>
> We have never worked container like logic like that in the kernel due to
> the complicated logic you would have to put in. The requirement that all
> objects in a page come from the same container is not necessary. If you
> drop this notion then things become very easy and the patches will become
> simple.

I promise to look at that in more detail and get back to it. In the meantime, I think it would be enlightening to hear from other parties as well, specially the ones also directly interested in using the technology.