Subject: Re: [PATCH v3 00/28] kmem limitation for memcg
Posted by Christoph Lameter on Tue, 29 May 2012 16:01:03 GMT

On Tue, 29 May 2012, Glauber Costa wrote:

> > I think it may be simplest to only account for the pages used by a slab in
> > a memcg. That code could be added to the functions in the slab allocators
> > that interface with the page allocators. Those are not that performance
> > critical and would do not much harm.
>
> No, I don't think so. Well, accounting the page is easy, but when we do a new
> allocation, we need to match a process to its correspondent page. This will
> likely lead to flushing the internal cpu caches of the slub, for instance,
> hurting performance. That is because once we allocate a page, all objects on
> that page need to belong to the same cgroup.

Matching a process to its page is a complex thing even for pages used by
userspace.

How can you make sure that all objects on a page belong to the same
cgroup? There are various kernel allocations that have uses far beyond a
single context. There is already a certain degree of fuzziness there and
we tolerate that in other contexts as well.

> Also, you talk about intrusiveness, accounting pages is a lot more intrusive,
> since then you need to know a lot about the internal structure of each cache.
> Having the cache replicated has exactly the effect of isolating it better.

Why would you need to know about the internal structure? Just get the
current process context and use the cgroup that is readily available there
to account for the pages.

> > If you need per object accounting then the cleanest solution would be to
> > duplicate the per node arrays per memcg (or only the statistics) and have
> > the kmem_cache structure only once in memory.
>
> No, it's all per-page. Nothing here is per-object, maybe you misunderstood
> something?

There are free/used object counters in each page. You could account for
objects in the l3 lists or kmem_cache_node strcut and thereby avoid
having to deal with the individual objects at the per cpu level.