Subject: Re: [PATCH v5 06/18] consider a memcg parameter in kmem_create_cache
Posted by Glauber Costa on Thu, 25 Oct 2012 13:42:55 GMT
View Forum Message <> Reply to Message

On 10/23/2012 09:50 PM, JoonSoo Kim wrote:
>> -struct kmem_cache *__kmem_cache_alias(const char *name, size_t size,
>> > - size_t align, unsigned long flags, void (*ctor)(void *))
>> > +struct kmem_cache *
>> > +__kmem_cache_alias(struct mem_cgroup *memcg, const char *name, size_t size,
>> > + size_t align, unsigned long flags, void (*ctor)(void *))
>> > {
>> > struct kmem_cache *s;
>> >
>> > - s = find_mergeable(size, align, flags, name, ctor);
>> > + s = find_mergeable(memcg, size, align, flags, name, ctor);
>> > if (s) {
>> > s->refcount++;
>> > /*
> If your intention is that find_mergeable() works for memcg-slab-caches properly,
> it cannot works properly with this code.
> When memcg is not NULL, slab cache is only added to memcg's slab cache list.
> find_mergeable() only interate on original-slab-cache list.
> So memcg slab cache never be mergeable.

Actually, recent results made me reconsider this.

I split this in multiple lists so we could transverse the lists faster
for /proc/slabinfo.

Turns out, there are many places that will rely on the ability to scan
through *all* caches in the system (root or not). This is one (easily
fixable) example, but there are others, like the hotplug handlers.

That said, I don't think that /proc/slabinfo is *that* performance
sensitive, so it is better to just skip the non-root caches, and just
keep all caches in the global list.

Maybe we would still benefit from a memcg-side list, for example, when
we're destructing memcg, so I'll consider keeping that (with a list
field in memcg_params). But even for that one, is still doable to
transverse the whole list...