Subject: Re: [PATCH 19/23] slab: per-memcg accounting of slab caches
Posted by Glauber Costa on Wed, 02 May 2012 15:40:37 GMT
View Forum Message <> Reply to Message

>> @@ -3834,11 +3866,15 @@ static inline void __cache_free(struct kmem_cache *cachep,
void *objp,
>> */
>> void *kmem_cache_alloc(struct kmem_cache *cachep, gfp_t flags)
>> {
>> -	void *ret = __cache_alloc(cachep, flags, __builtin_return_address(0));
>> +	void *ret;
>> +
>> +	rcu_read_lock();
>> +	cachep = mem_cgroup_get_kmem_cache(cachep, flags);
>> +	rcu_read_unlock();
>
> Don't we need to check in_interrupt(), current, __GFP_NOFAIL every
> time we call mem_cgroup_cgroup_get_kmem_cache()?
>
> I would personally prefer if those checks were put inside
> mem_cgroup_get_kmem_cache() instead of having to check for every
> caller.
>

in_interrupt() yes, __GFP_NOFAIL I don't think so.

__GFP_NOFAIL should lead to a res_counter_charge_nofail() in the end.
The name similarity is no coincidence...

 From a code style PoV, it makes sense to bundle an in_interrupt() check
here, but from a performance PoV, putting it in the callers can help us
avoid the price of a function call.

But well, looking at the code, I see it is not there as well... =(

I plan to change memcontrol.h to look like this:

```
static __always_inline struct kmem_cache *
mem_cgroup_get_kmem_cache(struct kmem_cache *cachep, gfp_t gfp)
{
	if (mem_cgroup_kmem_on && current->mm && !in_interrupt())
		return __mem_cgroup_get_kmem_cache(cachep, gfp);
	return cachep;
}
```