Subject: Re: [PATCH v3 06/13] memcg: kmem controller infrastructure
Posted by Glauber Costa on Fri, 21 Sep 2012 08:41:58 GMT
View Forum Message <> Reply to Message

On 09/20/2012 08:05 PM, JoonSoo Kim wrote:
> Hi, Glauber.
>
> 2012/9/18 Glauber Costa <glommer@parallels.com>:
>> +/*
>> + * We need to verify if the allocation against current->mm->owner's memcg is
>> + * possible for the given order. But the page is not allocated yet, so we'll
>> + * need a further commit step to do the final arrangements.
>> + *
>> + * It is possible for the task to switch cgroups in this mean time, so at
>> + * commit time, we can't rely on task conversion any longer. We'll then use
>> + * the handle argument to return to the caller which cgroup we should commit
>> + * against. We could also return the memcg directly and avoid the pointer
>> + * passing, but a boolean return value gives better semantics considering
>> + * the compiled-out case as well.
>> + *
>> + * Returning true means the allocation is possible.
>> + */
>> +bool
>> +__memcg_kmem_newpage_charge(gfp_t gfp, struct mem_cgroup **_memcg, int order)
>> +{
>> +	struct mem_cgroup *memcg;
>> +	bool ret;
>> +	struct task_struct *p;
>> +
>> +	*_memcg = NULL;
>> +	rcu_read_lock();
>> +	p = rcu_dereference(current->mm->owner);
>> +	memcg = mem_cgroup_from_task(p);
>> +	rcu_read_unlock();
>> +
>> +	if (!memcg_can_account_kmem(memcg))
>> +		return true;
>> +
>> +	mem_cgroup_get(memcg);
>> +
>> +	ret = memcg_charge_kmem(memcg, gfp, PAGE_SIZE << order) == 0;
>> +	if (ret)
>> +		*_memcg = memcg;
>> +	else
>> +		mem_cgroup_put(memcg);
>> +
>> +	return ret;
>> +}

>
> "*_memcg = memcg" should be executed when "memcg_charge_kmem" is success.
> "memcg_charge_kmem" return 0 if success in charging.
> Therefore, I think this code is wrong.
> If I am right, it is a serious bug that affect behavior of all the patchset.

Which is precisely what it does. ret is a boolean, that will be true
when charge succeeded (== 0 test)

>
>> +void __memcg_kmem_commit_charge(struct page *page, struct mem_cgroup *memcg,
>> +   int order)
>> +{
>> +   struct page_cgroup *pc;
>> +
>> +   WARN_ON(mem_cgroup_is_root(memcg));
>> +
>> +   /* The page allocation failed. Revert */
>> +   if (!page) {
>> +   memcg_uncharge_kmem(memcg, PAGE_SIZE << order);
>> +   return;
>> +   }
>
> In case of "!page ", mem_cgroup_put(memcg) is needed,
> because we already call "mem_cgroup_get(memcg)" in
> __memcg_kmem_newpage_charge().
> I know that mem_cgroup_put()/get() will be removed in later patch, but
> it is important that every patch works fine.

Okay, I'll add the put here. It is indeed missing.
