Subject: Re: [PATCH v7 1/2] Always free struct memcg through schedule_work()
Posted by Glauber Costa on Fri, 25 May 2012 09:51:22 GMT
View Forum Message <> Reply to Message

On 05/25/2012 01:50 PM, Michal Hocko wrote:
> On Fri 25-05-12 13:32:07, Glauber Costa wrote:
>> Right now we free struct memcg with kfree right after a
>> rcu grace period, but defer it if we need to use vfree() to get
>> rid of that memory area. We do that by need, because we need vfree
>> to be called in a process context.
>>
>> This patch unifies this behavior, by ensuring that even kfree will
>> happen in a separate thread. The goal is to have a stable place to
>> call the upcoming jump label destruction function outside the realm
>> of the complicated and quite far-reaching cgroup lock (that can't be
>> held when calling neither the cpu_hotplug.lock nor the jump_label_mutex)
>>
>> Signed-off-by: Glauber Costa<glommer@parallels.com>
>> Acked-by: Kamezawa Hiroyuki<kamezawa.hiroyu@jp.fujitsu.com>
>
> Acked-by: Michal Hocko<mhocko@suse.cz>
>
> Just one comment below
>
>> CC: Tejun Heo<tj@kernel.org>
>> CC: Li Zefan<lizefan@huawei.com>
>> CC: Johannes Weiner<hannes@cmpxchg.org>
>> CC: Michal Hocko<mhocko@suse.cz>
>> CC: Andrew Morton<akpm@linux-foundation.org>
>> ---
>> mm/memcontrol.c | 24 ++++++++++++++++++-----------
>> 1 files changed, 13 insertions(+), 11 deletions(-)
>>
>> diff --git a/mm/memcontrol.c b/mm/memcontrol.c
>> index 932a734..0b4b4c8 100644
>> --- a/mm/memcontrol.c
>> +++ b/mm/memcontrol.c
> [...]
>> @@ -4826,23 +4826,28 @@ out_free:
>> }
>>
>> /*
>> - * Helpers for freeing a vzalloc()ed mem_cgroup by RCU,
>> + * Helpers for freeing a kmalloc()ed/vzalloc()ed mem_cgroup by RCU,
>>   * but in process context. The work_freeing structure is overlaid
>>   * on the rcu_freeing structure, which itself is overlaid on memsw.
>>   */
>> -static void vfree_work(struct work_struct *work)

>> +static void free_work(struct work_struct *work)
>> {
>>   struct mem_cgroup *memcg;
>> + int size = sizeof(struct mem_cgroup);
>>
>>   memcg = container_of(work, struct mem_cgroup, work_freeing);
>> - vfree(memcg);
>> + if (size< PAGE_SIZE)
>
> What about
> if (is_vmalloc_addr(memcg))
>> + kfree(memcg);
>> + else
>> + vfree(memcg);
>> }
>
Could be, but I believe this one is already in Andrew's tree from last
submission (might be wrong)
