Subject: Re: [RFC}[PATCH] forced uncharge for successful rmdir.
Posted by Balbir Singh on Mon, 01 Oct 2007 06:11:22 GMT
View Forum Message <> Reply to Message

KAMEZAWA Hiroyuki wrote:
> Hi, thank you for review.
>

Your always welcome, thanks for helping with the controller.

> On Mon, 01 Oct 2007 09:46:02 +0530
> Balbir Singh <balbir@linux.vnet.ibm.com> wrote:
>
```
>>> @@ -424,17 +424,80 @@ void mem_cgroup_uncharge(struct page_cgr
>>>   if (atomic_dec_and_test(&pc->ref_cnt)) {
>>>    page = pc->page;
>>>    lock_page_cgroup(page);
>>> -  mem = pc->mem_cgroup;
>>> -  css_put(&mem->css);
>>> -  page_assign_page_cgroup(page, NULL);
>>> -  unlock_page_cgroup(page);
>>> -  res_counter_uncharge(&mem->res, PAGE_SIZE);
>>> +  pc = page_get_page_cgroup(page);
>>> +  if (pc) {
>>> +   mem = pc->mem_cgroup;
>>> +   css_put(&mem->css);
>>> +   page_assign_page_cgroup(page, NULL);
>>> +   unlock_page_cgroup(page);
>>> +   res_counter_uncharge(&mem->res, PAGE_SIZE);
>>> +    spin_lock_irqsave(&mem->lru_lock, flags);
>>> +    list_del_init(&pc->lru);
>>> +    spin_unlock_irqrestore(&mem->lru_lock, flags);
>>> +   kfree(pc);
>>> +  } else
>>> +   unlock_page_cgroup(page);
>>> + }
>>> +}
```
>> This looks like a bug fix in mem_cgroup_uncharge(). Did you hit a
>> condition of simultaneous free? Could we split this up into a separate
>> patch please.
> No, but forced-uncharge and usual unchage will have race.
> "page" is linked to zone's LRU while unchage is going.
>
>

OK

```
>>> +  page = pc->page;
>>> +  lock_page_cgroup(page);
>>> +  pc = page_get_page_cgroup(page);
>>> +  /* check race */
>>> +  if (pc) {
>>> +   css_put(&mem->css);
>>> +   page_assign_page_cgroup(page, NULL);
>>> +   unlock_page_cgroup(page);
>>> +   res_counter_uncharge(&mem->res, PAGE_SIZE);
>>> +   list_del_init(&pc->lru);
>>> +   kfree(pc);
>>> +  } else
>>> +   unlock_page_cgroup(page);
>>> +  if (--count == 0) {
>>> +   spin_unlock(&mem->lru_lock);
>>> +   cond_resched();
>>> +   spin_lock(&mem->lru_lock);
>>> +   count = SWAP_CLUSTER_MAX;
>>> +  }
>>> + }
>>> + spin_unlock(&mem->lru_lock);
>>> +}
```
>> The forced_uncharge_list is one way of doing it, the other
>> way is to use a shrink_all_memory() logic. For now, I think
>> this should be fine.
> I have both versions. I myself think forced-unchage is better.
>

OK, I think we can try and see how forced uncharge works.

```
>>> - if (tmp <= MEM_CGROUP_TYPE_UNSPEC || tmp >= MEM_CGROUP_TYPE_MAX)
>>> + if (tmp == MEM_CGROUP_TYPE_UNSPEC) {
>>> +  if (atomic_read(&mem->css.cgroup->count) == 0)  /* uncharge all */
>>> +   ret = mem_cgroup_forced_uncharge_all(mem);
>>> +  else
>>> +   ret = -EBUSY;
>>> +  if (!ret)
>>> +   ret = nbytes;
>>> +  goto out_free;
>>> + }
>>> +
```
>> Can we use a different file for this? Something like
>> memory.force_reclaim or memory.force_out_memory?
> Yes, okay. How about drop_charge ?
> (This uncharge doesn't drop memory...)
>

drop_charge is a technical term, I was hoping to find something that
the administrators can easily understand.

>>> + if (tmp < MEM_CGROUP_TYPE_UNSPEC || tmp >= MEM_CGROUP_TYPE_MAX)
>>>   goto out_free;
>>>
>>>   mem->control_type = tmp;
>>>
>> Overall, the patch looks good. I am going to stress test this patch.
>>
> Thanks. I'll post again when the next -mm comes.
>

Thanks! I'll test the current changes.

--
 Warm Regards,
 Balbir Singh
 Linux Technology Center
 IBM, ISTL