Subject: Re: [Fwd: [PATCH -RSS 1/1] Fix reclaim failure]
Posted by Balbir Singh on Tue, 05 Jun 2007 08:12:21 GMT
View Forum Message <> Reply to Message

Pavel Emelianov wrote:
>>
>> static unsigned long isolate_container_pages(unsigned long nr_to_scan,
>> - struct list_head *src, struct list_head *dst,
>> - unsigned long *scanned, struct zone *zone, int mode)
>> + struct rss_container *rss, struct list_head *dst,
>> + unsigned long *scanned, struct zone *zone, int mode,
>> + int active)
>> {
>>   unsigned long nr_taken = 0;
>>   struct page *page;
>>   struct page_container *pc;
>>   unsigned long scan;
>>   LIST_HEAD(pc_list);
>> + struct list_head *src;
>> +
>> + src = active ? &rss->active_list : &rss->inactive_list;
>>
>>   for (scan = 0; scan < nr_to_scan && !list_empty(src); scan++) {
>>   pc = list_entry(src->prev, struct page_container, list);
>>   page = pc->page;
>> +
>> + /*
>> +   * We might have got our active, inactive lists
>> +   * incorrect, fix it here
>> +   */
>> + if (active && !PageActive(page)) {
>> +   list_move(&pc->list, &rss->inactive_list);
>> +   scan--;
>> +   continue;
>> + } else if (!active && PageActive(page)) {
>> +   list_move(&pc->list, &rss->active_list);
>> +   scan--;
>> +   continue;
>> + }
>> +
>
> Actually the plan was to keep these lists consistent, i.e. when page
> drops the active bit and moves to the inactive global LRU list, the
> according page_container should be migrated as well. Where's the place
> that messes the lists? I thought I found all the places when the page
> migrates across the lists...
>

Yes, we do that. This fix is required for the situation occurs when a
page is brought in initially. A file backed page does not have it's
PG_active bit. Alternatively, we could modify the call sites to put
the page in the correct list (active/inactive), but that can easily
lead to complexity in the case the page is already on the LRU.

>>   /*
>>   * TODO: now we hold all the pages in one... ok, two lists
>>   * and skip the pages from another zones with the check
>> @@ -249,12 +268,8 @@ unsigned long isolate_pages_in_container
>>
>>   /* we are called with zone->lru_lock held with irqs disabled */
>>   spin_lock(&rss->res.lock);
>> - if (active)
>> - ret = isolate_container_pages(nr_to_scan, &rss->active_list,
>> -   dst, scanned, zone, mode);
>> - else
>> - ret = isolate_container_pages(nr_to_scan, &rss->inactive_list,
>> -   dst, scanned, zone, mode);
>> + ret = isolate_container_pages(nr_to_scan, rss, dst, scanned, zone,
>> +   mode, active);
>
> I wanted to keep the solution of what list to select here to make it
> easier to switch to per-zone containers lists. With this check moved
> to the actual isolation function we won't be able to isolate pages from
> arbitrary list if we need such, but I believe we will need.
>

Hmm.. if we change adding back the pages correctly in the call site, this
change can be avoided.

>>   spin_unlock(&rss->res.lock);
>>   return ret;
>> }
>> _
>>
>
> Thanks,
> Pavel

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

_____
Containers mailing list