Subject: Re: [PATCH 3/3][RFC] Containers: Pagecache controller reclaim
Posted by Vaidyanathan Srinivas on Tue, 27 Mar 2007 07:17:10 GMT
View Forum Message <> Reply to Message

Aubrey Li wrote:
> On 3/6/07, Vaidyanathan Srinivasan <svaidy@linux.vnet.ibm.com> wrote:
>> The reclaim code is similar to RSS memory controller. Scan control is
>> slightly different since we are targeting different type of pages.
>>
>> Additionally no mapped pages are touched when scanning for pagecache pages.
>>
>> RSS memory controller and pagecache controller share common code in reclaim
>> and hence pagecache controller patches are dependent on RSS memory controller
>> patch even though the features are independently configurable at compile time.
>>
>> --- linux-2.6.20.orig/mm/vmscan.c
>> +++ linux-2.6.20/mm/vmscan.c
>> @@ -43,6 +43,7 @@
>>
>> #include <linux/swapops.h>
>> #include <linux/memcontrol.h>
>> +#include <linux/pagecache_acct.h>
>>
>> #include "internal.h"
>>
>> @@ -70,6 +71,8 @@ struct scan_control {
>>
>> struct container *container; /* Used by containers for reclaiming */
>> /* pages when the limit is exceeded */
>> + int reclaim_pagecache_only; /* Set when called from
>> + pagecache controller */
>> };
>>
>> /*
>> @@ -474,6 +477,15 @@ static unsigned long shrink_page_list(st
>> goto keep;
>>
>> VM_BUG_ON(PageActive(page));
>> + /* Take it easy if we are doing only pagecache pages */
>> + if (sc->reclaim_pagecache_only) {
>> + /* Check if this is a pagecache page they are not mapped */
>> + if (page_mapped(page))
>> + goto keep_locked;
>> + /* Check if this container has exceeded pagecache limit */
>> + if (!pagecache_acct_page_overlimit(page))
>> + goto keep_locked;
>> + }
>>

```
>>              sc->nr_scanned++;
>>
>> @@ -522,7 +534,8 @@ static unsigned long shrink_page_list(st
>>              }
>>
>>              if (PageDirty(page)) {
>> -                    if (referenced)
>> +                    /* Reclaim even referenced pagecache pages if over limit */
>> +                    if (!pagecache_acct_page_overlimit(page) && referenced)
>>                              goto keep_locked;
>>                      if (!may_enter_fs)
>>                              goto keep_locked;
>> @@ -869,6 +882,13 @@ force_reclaim_mapped:
>>              cond_resched();
>>              page = lru_to_page(&l_hold);
>>              list_del(&page->lru);
>> +            /* While reclaiming pagecache make it easy */
>> +            if (sc->reclaim_pagecache_only) {
>> +                    if (page_mapped(page) || !pagecache_acct_page_overlimit(page)) {
>> +                            list_add(&page->lru, &l_active);
>> +                            continue;
>> +                    }
>> +            }
>
> Please correct me if I'm wrong.
> Here, if page type is mapped or not overlimit, why add it back to active list?
> Did  shrink_page_list() is called by shrink_inactive_list()?
```

Correct, shrink_page_list() is called from shrink_inactive_list() but
the above code is patched in shrink_active_list(). The
'force_reclaim_mapped' label is from function shrink_active_list() and
not in shrink_page_list() as it may seem in the patch file.

While removing pages from active_list, we want to select only
pagecache pages and leave the remaining in the active_list.
page_mapped() pages are _not_ of interest to pagecache controller
(they will be taken care by rss controller) and hence we put it back.
 Also if the pagecache controller is below limit, no need to reclaim
so we put back all pages and come out.

--Vaidy