Subject: Re: [RFC] [-mm PATCH] Memory controller fix swap charging context in unuse_pte()
Posted by Balbir Singh on Sun, 28 Oct 2007 20:32:19 GMT
View Forum Message <> Reply to Message

On Mon, Oct 29, 2007 at 01:57:40AM +0530, Balbir Singh wrote:
Hugh Dickins wrote:

[snip]

> Without your mem_cgroup mods in mm/swap_state.c, unuse_pte makes
> the right assignments (I believe). But I find that swapout (using
> 600M in a 512M machine) from a 200M cgroup quickly OOMs, whereas
> it behaves correctly with your mm/swap_state.c.
>

On my UML setup, I booted the UML instance with 512M of memory and
used the swapout program that you shared. I tried two things

1. Ran swapout without any changes. The program ran well without
   any OOM condition occuring, lot of reclaim occured.
2. Ran swapout with the changes to mm/swap_state.c removed (diff below)
   and I still did not see any OOM. The reclaim count was much lesser
   since swap cache did not get accounted back to the cgroup from
   which pages were being evicted.

I am not sure why I don't see the OOM that you see, still trying. May be
I missing something obvious at this late hour in the night :-)

Output of the tests

```
balbir@ubuntu:/container/swapout$ cat memory.limit_in_bytes
209715200
balbir@ubuntu:/container/swapout$ cat memory.usage_in_bytes
65536
balbir@ubuntu:/container/swapout$ cat tasks
1815
1847
balbir@ubuntu:/container/swapout$ ps
  PID TTY          TIME CMD
 1815 pts/0    00:00:00 bash
 1848 pts/0    00:00:00 ps
balbir@ubuntu:/container/swapout$ ~/swapout
balbir@ubuntu:/container/swapout$ echo $?
0
balbir@ubuntu:/container/swapout$ cat memory.failcnt
```

18

Diff to remove mods from swap_state.c (for testing only)

```
--- mm/swap_state.c.org 2007-10-29 01:42:14.000000000 +0530
+++ mm/swap_state.c 2007-10-29 01:52:48.000000000 +0530
@@ -79,10 +79,6 @@ static int __add_to_swap_cache(struct pa
  BUG_ON(PageSwapCache(page));
  BUG_ON(PagePrivate(page));

- error = mem_cgroup_cache_charge(page, current->mm, gfp_mask);
- if (error)
-  goto out;
-
  error = radix_tree_preload(gfp_mask);
  if (!error) {
   write_lock_irq(&swapper_space.tree_lock);
@@ -94,14 +90,11 @@ static int __add_to_swap_cache(struct pa
    set_page_private(page, entry.val);
    total_swapcache_pages++;
    __inc_zone_page_state(page, NR_FILE_PAGES);
-  } else
-   mem_cgroup_uncharge_page(page);
+  }

   write_unlock_irq(&swapper_space.tree_lock);
   radix_tree_preload_end();
- } else
-  mem_cgroup_uncharge_page(page);
-out:
+ }
  return error;
 }

@@ -141,7 +134,6 @@ void __delete_from_swap_cache(struct pag
  BUG_ON(PageWriteback(page));
  BUG_ON(PagePrivate(page));

- mem_cgroup_uncharge_page(page);
  radix_tree_delete(&swapper_space.page_tree, page_private(page));
  set_page_private(page, 0);
  ClearPageSwapCache(page);
```

--

Warm Regards,
Balbir Singh
Linux Technology Center
IBM, ISTL

Containers mailing list
Containers@lists.linux-foundation.org
https://lists.linux-foundation.org/mailman/listinfo/containers