Subject: Re: problem with ZONE_MOVABLE.
Posted by Balbir Singh on Thu, 13 Sep 2007 10:30:06 GMT
View Forum Message <> Reply to Message

KAMEZAWA Hiroyuki wrote:
> Hi,
>
> While I'm playing with memory controller of 2.6.23-rc4-mm1, I met following.
>
> ==
> [root@drpq test-2.6.23-rc4-mm1]# echo $$ > /opt/mem_control/group_1/tasks
> [root@drpq test-2.6.23-rc4-mm1]# cat /opt/mem_control/group_1/memory.limit
> 32768
> [root@drpq test-2.6.23-rc4-mm1]# cat /opt/mem_control/group_1/memory.usage
> 286
> // Memory is limited to 512 GiB. try "dd" 1GiB (page size is 16KB)
>
> [root@drpq test-2.6.23-rc4-mm1]# dd if=/dev/zero of=/tmp/tmpfile bs=1024 count=1048576
> Killed
> [root@drpq test-2.6.23-rc4-mm1]# ls
> Killed
> //above are caused by OOM.
> [root@drpq test-2.6.23-rc4-mm1]# cat /opt/mem_control/group_1/memory.usage
> 32763
> [root@drpq test-2.6.23-rc4-mm1]# cat /opt/mem_control/group_1/memory.limit
> 32768
> // fully filled by page cache. no reclaim run.
> ==
>
> The reason this happens is because I used kernelcore= boot option, i.e
> ZONE_MOVABLE. Seems try_to_free_mem_container_pages() ignores ZONE_MOVABLE.
>
> Quick fix is attached, but Mel's one-zonelist-pernode patch may change this.
> I'll continue to watch.
>
> Thanks,
> -Kame
> ==
> Now, there is ZONE_MOVABLE...
>
> page cache and user pages are allocated from gfp_zone(GFP_HIGHUSER_MOVABLE)
>
> Signed-off-by: KAMEZAWA Hiroyuki <kamezawa.hiroyu@jp.fujitsu.com>
> ---
> mm/vmscan.c | 9 ++-------
> 1 file changed, 2 insertions(+), 7 deletions(-)
>
> Index: linux-2.6.23-rc4-mm1.bak/mm/vmscan.c

> ===
> --- linux-2.6.23-rc4-mm1.bak.orig/mm/vmscan.c
> +++ linux-2.6.23-rc4-mm1.bak/mm/vmscan.c
> @@ -1351,12 +1351,6 @@ unsigned long try_to_free_pages(struct z
>
> #ifdef CONFIG_CONTAINER_MEM_CONT
>
> -#ifdef CONFIG_HIGHMEM
> -#define ZONE_USERPAGES ZONE_HIGHMEM
> -#else
> -#define ZONE_USERPAGES ZONE_NORMAL
> -#endif
> -
> unsigned long try_to_free_mem_container_pages(struct mem_container *mem_cont)
> {
>   struct scan_control sc = {
> @@ -1371,9 +1365,10 @@ unsigned long try_to_free_mem_container_
>   };
>   int node;
>   struct zone **zones;
> + int target_zone = gfp_zone(GFP_HIGHUSER_MOVABLE);
>
>   for_each_online_node(node) {
> - zones = NODE_DATA(node)->node_zonelists[ZONE_USERPAGES].zones;
> + zones = NODE_DATA(node)->node_zonelists[target_zone].zones;
>   if (do_try_to_free_pages(zones, sc.gfp_mask, &sc))
>   return 1;
>   }

Mel, has sent out a fix (for the single zonelist) that conflicts with
this one. Your fix looks correct to me, but it will be over ridden
by Mel's fix (once those patches are in -mm).

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