
Subject: [PATCH v3 11/13] memcg: allow a memcg with kmem charges to be destroyed.

Posted by [Glauber Costa](#) on Tue, 18 Sep 2012 14:04:08 GMT

[View Forum Message](#) <> [Reply to Message](#)

Because the ultimate goal of the kmem tracking in memcg is to track slab pages as well, we can't guarantee that we'll always be able to point a page to a particular process, and migrate the charges along with it - since in the common case, a page will contain data belonging to multiple processes.

Because of that, when we destroy a memcg, we only make sure the destruction will succeed by discounting the kmem charges from the user charges when we try to empty the cgroup.

Signed-off-by: Glauber Costa <glommer@parallels.com>

Acked-by: Kamezawa Hiroyuki <kamezawa.hiroyu@jp.fujitsu.com>

CC: Christoph Lameter <cl@linux.com>

CC: Pekka Enberg <penberg@cs.helsinki.fi>

CC: Michal Hocko <mhocko@suse.cz>

CC: Johannes Weiner <hannes@cmpxchg.org>

CC: Suleiman Souhlal <suleiman@google.com>

mm/memcontrol.c | 17 ++++++

1 file changed, 16 insertions(+), 1 deletion(-)

diff --git a/mm/memcontrol.c b/mm/memcontrol.c

index aada601..b05ecac 100644

--- a/mm/memcontrol.c

+++ b/mm/memcontrol.c

@@ -631,6 +631,11 @@ static void disarm_kmem_keys(struct mem_cgroup *memcg)

```
{
    if (memcg_kmem_is_accounted(memcg))
        static_key_slow_dec(&memcg_kmem_enabled_key);
+ /*
```

```
+ * This check can't live in kmem destruction function,
```

```
+ * since the charges will outlive the cgroup
```

```
+ */
```

```
+ WARN_ON(res_counter_read_u64(&memcg->kmem, RES_USAGE) != 0);
```

```
}
```

```
#else
```

```
static void disarm_kmem_keys(struct mem_cgroup *memcg)
```

```
@@ -3933,6 +3938,7 @@ static int mem_cgroup_force_empty(struct mem_cgroup *memcg, bool
free_all)
```

```
int node, zid, shrink;
```

```
int nr_retries = MEM_CGROUP_RECLAIM_RETRIES;
```

```
struct cgroup *cgrp = memcg->css.cgroup;
```

```
+ u64 usage;
```

```

css_get(&memcg->css);

@@ -3966,8 +3972,17 @@ move_account:
    mem_cgroup_end_move(memcg);
    memcg_oom_recover(memcg);
    cond_resched();
+ /*
+  * Kernel memory may not necessarily be trackable to a specific
+  * process. So they are not migrated, and therefore we can't
+  * expect their value to drop to 0 here.
+  *
+  * having res filled up with kmem only is enough
+  */
+ usage = res_counter_read_u64(&memcg->res, RES_USAGE) -
+ res_counter_read_u64(&memcg->kmem, RES_USAGE);
+ /* "ret" should also be checked to ensure all lists are empty. */
- } while (res_counter_read_u64(&memcg->res, RES_USAGE) > 0 || ret);
+ } while (usage > 0 || ret);
out:
    css_put(&memcg->css);
    return ret;
--
1.7.11.4

```
