
Subject: [RFC][PATCH 3/4] memcg shrink usage at limit change
Posted by [KAMEZAWA Hiroyuki](#) on Fri, 18 Jul 2008 10:36:34 GMT
[View Forum Message](#) <> [Reply to Message](#)

Shrinking memory usage at limit change.

This is an enhancement. based on res_counter-limit-change-ebusy.patch

Changelog: v3 -> v4

- core logic is separated into two parts. (reuse the function later.)
- added cond_resched() again.
 1. I was pointed out that alloc_pages() does cond_resched() after try_to_free_pages().
 2. This routine is called by an user. A user can set 'nice' in general.

Changelog: v2 -> v3

- supported interrupt by signal. (A user can stop limit change by Ctrl-C.)

Changelog: v1 -> v2

- adjusted to be based on write_string() patch set
- removed backward goto.
- removed unnecessary cond_resched().

Signed-off-by: KAMEZAWA Hiroyuki <kamezawa.hiroyu@jp.fujitsu.com>

Documentation/controllers/memory.txt | 3 -
mm/memcontrol.c | 73 ++++++++-----
2 files changed, 70 insertions(+), 6 deletions(-)

Index: mmtom-stamp-2008-07-15-15-39/mm/memcontrol.c

=====

--- mmtom-stamp-2008-07-15-15-39.orig/mm/memcontrol.c

+++ mmtom-stamp-2008-07-15-15-39/mm/memcontrol.c

@@ -804,6 +804,55 @@ int mem_cgroup_shrink_usage(struct mm_st
{

/*

+ * Shrink routine works cooperatively with users rather than the kernel.

+ * So,

+ * - we have to check signals

+ * - we pass GFP_HIGHUSERMOVABLE.

+ * TODO?:

+ * - allow timeout

+ */

+static int mem_cgroup_shrink_usage_to(struct mem_cgroup *memcg,
+ unsigned long long val)

+{

```

+ int retry_count = MEM_CGROUP_RECLAIM_RETRIES;
+ int progress;
+ int ret;
+
+ while (!res_counter_check_under_val(&memcg->res, val)) {
+   if (signal_pending(current)) {
+     ret = -EINTR;
+     break;
+   }
+   if (!retry_count) {
+     ret = -EBUSY;
+     break;
+   }
+   progress = try_to_free_mem_cgroup_pages(memcg,
+     GFP_HIGHUSER_MOVABLE);
+   if (!progress)
+     retry_count--;
+   cond_resched();
+ }
+ return 0;
+}
+
+int mem_cgroup_resize_limit(struct mem_cgroup *memcg, unsigned long long val)
+{
+ int ret = 0;
+
+ do {
+   ret = mem_cgroup_shrink_usage_to(memcg, val);
+   if (ret < 0)
+     break;
+   /* memory usage shrinking succeed. */
+   if (res_counter_set_limit(&memcg->res, val))
+     break;
+ } while (1);
+ return ret;
+}
+
+/*
+ * This routine traverse page_cgroup in given list and drop them all.
+ * *And* this routine doesn't reclaim page itself, just removes page_cgroup.
+ */
@@ -883,13 +932,29 @@ static u64 mem_cgroup_read(struct cgroup
+   return res_counter_read_u64(&mem_cgroup_from_cont(cont)->res,
+     cft->private);
+ }
+
+ /*

```

```

+ * The user of this function is...
+ * RES_LIMIT.
+ */
static int mem_cgroup_write(struct cgroup *cont, struct cftype *cft,
    const char *buffer)
{
- return res_counter_write(&mem_cgroup_from_cont(cont)->res,
-     cft->private, buffer,
-     res_counter_memparse_write_strategy);
+ struct mem_cgroup *memcg = mem_cgroup_from_cont(cont);
+ unsigned long long val;
+ int ret;
+
+ switch (cft->private) {
+ case RES_LIMIT:
+ /* This function does all necessary parse...reuse it */
+ ret = res_counter_memparse_write_strategy(buffer, &val);
+ if (!ret)
+     ret = mem_cgroup_resize_limit(memcg, val);
+ break;
+ default:
+ ret = -EINVAL; /* should be BUG() ? */
+ break;
+ }
+ return ret;
}

```

```
static int mem_cgroup_reset(struct cgroup *cont, unsigned int event)
```

Index: mmtom-stamp-2008-07-15-15-39/Documentation/controllers/memory.txt

=====

--- mmtom-stamp-2008-07-15-15-39.orig/Documentation/controllers/memory.txt

+++ mmtom-stamp-2008-07-15-15-39/Documentation/controllers/memory.txt

@@ -242,8 +242,7 @@ rmdir() if there are no tasks.

1. Add support for accounting huge pages (as a separate controller)
2. Make per-cgroup scanner reclaim not-shared pages first
3. Teach controller to account for shared-pages
- 4. Start reclamation when the limit is lowered
- 5. Start reclamation in the background when the limit is
- +4. Start reclamation in the background when the limit is
not yet hit but the usage is getting closer

Summary

Containers mailing list

Containers@lists.linux-foundation.org

<https://lists.linux-foundation.org/mailman/listinfo/containers>