
Subject: [RFC 2/2] memcg: hierarchy support interface (yet another one)
Posted by [KAMEZAWA Hiroyuki](#) on Wed, 28 May 2008 07:57:27 GMT
[View Forum Message](#) <> [Reply to Message](#)

Hierarchy support for memcg.

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

Index: hie-2.6.26-rc2-mm1/mm/memcontrol.c

```
=====
--- hie-2.6.26-rc2-mm1.orig/mm/memcontrol.c
+++ hie-2.6.26-rc2-mm1/mm/memcontrol.c
@@ -792,6 +792,81 @@ int mem_cgroup_shrink_usage(struct mm_st
 }

/*
+ * Memory Controller hierarchy support.
+ */
+
+int memcg_shrink_callback(struct res_counter *cnt, unsigned long long val)
+{
+ struct mem_cgroup *memcg = container_of(cnt, struct mem_cgroup, res);
+ unsigned long flags;
+ int ret = 1;
+ int progress = 1;
+
+retry:
+ spin_lock_irqsave(&cnt->lock, flags);
+ /* Need to shrink ? */
+ if (cnt->usage + val <= cnt->limit)
+ ret = 0;
+ /* we can never get enough resource ? */
+ if (cnt->security + val > cnt->limit)
+ ret = -1;
+ spin_unlock_irqrestore(&cnt->lock, flags);
+ if (!ret)
+ return 0;
+ if (ret == -1)
+ return 1;
+ if (!progress)
+ return 1;
+ progress = try_to_free_mem_cgroup_pages(memcg, GFP_KERNEL);
+
+ goto retry;
+}
+
+int mem_cgroup_resize_callback(struct res_counter *cnt, unsigned long long val)
```

```

+{
+ struct mem_cgroup *child = container_of(cnt, struct mem_cgroup, res);
+ struct mem_cgroup *parent;
+ struct cgroup *my_cg;
+ unsigned long flags, borrow;
+ unsigned long long diffs;
+ int ret = 0;
+
+ my_cg = child->css.cgroup;
+ /* Is this root group ? */
+ if (!my_cg->parent) {
+ spin_lock_irqsave(&cnt->lock, flags);
+ cnt->limit = val;
+ spin_unlock_irqrestore(&cnt->lock, flags);
+ return 0;
+ }
+ spin_lock_irqsave(&cnt->lock, flags);
+ if (val > cnt->limit) {
+ diffs = val - cnt->limit;
+ borrow = 1;
+ } else {
+ diffs = cnt->limit - val;
+ borrow = 0;
+ }
+ spin_unlock_irqrestore(&cnt->lock, flags);
+
+ parent = mem_cgroup_from_cont(my_cg->parent);
+ /* When we increase resource, call borrow. When decrease, call repay*/
+ if (borrow)
+ ret = res_counter_borrow_resource(cnt, &parent->res, diffs,
+ memcg_shrink_callback, 5);
+ else
+ ret = res_counter_repay_resource(cnt, &parent->res, diffs,
+ memcg_shrink_callback, 5);
+ 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.
+ */
@@ -898,7 +973,8 @@ static ssize_t mem_cgroup_write(struct c
{

```

```

return res_counter_write(&mem_cgroup_from_cont(cont)->res,
    cft->private, userbuf, nbytes, ppos,
- mem_cgroup_write_strategy);
+ mem_cgroup_write_strategy,
+ mem_cgroup_resize_callback);
}

static int mem_cgroup_reset(struct cgroup *cont, unsigned int event)
@@ -992,6 +1068,11 @@ static struct cftype mem_cgroup_files[]
    .name = "stat",
    .read_map = mem_control_stat_show,
},
+ {
+ .name = "assigned_to_child",
+ .private = RES_SECURITY,
+ .read_u64 = mem_cgroup_read,
+ },
};

static int alloc_mem_cgroup_per_zone_info(struct mem_cgroup *mem, int node)
@@ -1069,6 +1150,8 @@ mem_cgroup_create(struct cgroup_subsys *
}

res_counter_init(&mem->res);
+ if (cont->parent)
+ res_counter_zero_limit(&mem->res);

for_each_node_state(node, N_POSSIBLE)
    if (alloc_mem_cgroup_per_zone_info(mem, node))
@@ -1095,6 +1178,14 @@ static void mem_cgroup_destroy(struct cg
{
    int node;
    struct mem_cgroup *mem = mem_cgroup_from_cont(cont);
+ struct mem_cgroup *parent;
+
+ if (cont->parent) {
+ parent = mem_cgroup_from_cont(cont->parent);
+ /* we did what we can...just returns what we borrow */
+ res_counter_repay_resource(&mem->res,
+ &parent->res, -1, NULL, 0);
+ }

for_each_node_state(node, N_POSSIBLE)
    free_mem_cgroup_per_zone_info(mem, node);

```

Containers mailing list
Containers@lists.linux-foundation.org

