Subject: [PATCH] fix bad behavior in use_hierarchy file
Posted by Glauber Costa on Mon, 25 Jun 2012 09:21:01 GMT
View Forum Message <> Reply to Message

I have an application that does the following:

* copy the state of all controllers attached to a hierarchy
* replicate it as a child of the current level.

I would expect writes to the files to mostly succeed, since they
are inheriting sane values from parents.

But that is not the case for use_hierarchy. If it is set to 0, we
succeed ok. If we're set to 1, the value of the file is automatically
set to 1 in the children, but if userspace tries to write the
very same 1, it will fail. That same situation happens if we
set use_hierarchy, create a child, and then try to write 1 again.

Now, there is no reason whatsoever for failing to write a value
that is already there. It doesn't even match the comments, that
states:

```
 /* If parent's use_hierarchy is set, we can't make any modifications
  * in the child subtrees...
```

since we are not changing anything.

The following patch tests the new value against the one we're storing,
and automatically return 0 if we're not proposing a change.

Signed-off-by: Glauber Costa <glommer@parallels.com>
CC: Dhaval Giani <dhaval.giani@gmail.com>
CC: Michal Hocko <mhocko@suse.cz>
CC: Kamezawa Hiroyuki <kamezawa.hiroyu@jp.fujitsu.com>
CC: Johannes Weiner <hannes@cmpxchg.org>

```
---
 mm/memcontrol.c |    6 ++++++
 1 file changed, 6 insertions(+)

diff --git a/mm/memcontrol.c b/mm/memcontrol.c
index ac35bcc..cccebbc 100644
--- a/mm/memcontrol.c
+++ b/mm/memcontrol.c
@@ -3779,6 +3779,10 @@ static int mem_cgroup_hierarchy_write(struct cgroup *cont, struct cftype *cft,
   parent_memcg = mem_cgroup_from_cont(parent);

  cgroup_lock();
```

```
+
+ if (memcg->use_hierarchy == val)
+  goto out;
+
  /*
   * If parent's use_hierarchy is set, we can't make any modifications
   * in the child subtrees. If it is unset, then the change can
@@ -3795,6 +3799,8 @@ static int mem_cgroup_hierarchy_write(struct cgroup *cont, struct cftype *cft,
    retval = -EBUSY;
  } else
   retval = -EINVAL;
+
+out:
  cgroup_unlock();

  return retval;
--
1.7.10.2
```