Subject: Re: Question : memrlimit cgroup's task_move (2.6.26-rc5-mm3)
Posted by Balbir Singh on Thu, 19 Jun 2008 18:25:56 GMT
View Forum Message <> Reply to Message

* KAMEZAWA Hiroyuki <kamezawa.hiroyu@jp.fujitsu.com> [2008-06-19 12:14:35]:

> I used memrlimit cgroup at the first time.
>
> May I ask a question about memrlimit cgroup ?
>

Hi, Kamezawa-San,

Could you please review/test the patch below to see if it solves your
problem? If it does, I'll push it up to Andrew

Description

This patch fixes a task migration problem reported by Kamezawa-San. This
patch should fix all issues with migraiton, except for a rare condition
documented in memrlimit_cgroup_move_task(). To fix that problem, we
would need to add transaction properties to cgroups.

The problem reported was that migrating to a group that did not have
sufficient limits to accept an incoming task caused a kernel warning.

Steps to reproduce

% mkdir /dev/cgroup/memrlimit/group_01
% mkdir /dev/cgroup/memrlimit/group_02
% echo 1G > /dev/cgroup/memrlimit/group_01/memrlimit.limit_in_bytes
% echo 0 > /dev/cgroup/memrlimit/group_02/memrlimit.limit_in_bytes
% echo $$ > /dev/cgroup/memrlimit/group_01/tasks
% echo $$ > /dev/cgroup/memrlimit/group_02/tasks
% exit

memrlimit does the right thing by not moving the charges to group_02,
but the task is still put into g2 (since we did not use can_attach to
fail migration). Once in g2, when we echo the task to the root cgroup,
it tries to uncharge the cost of the task from g2. g2 does not have
any charge associated with the task, hence we get a warning.

Reported-by: kamezawa.hiroyu@jp.fujitsu.com
Signed-off-by: Balbir Singh <balbir@linux.vnet.ibm.com>
---

include/linux/res_counter.h | 18 ++++++++++++++++++

```
 mm/memrlimitcgroup.c        |   40 ++++++++++++++++++++++++++++++++++++++++
 2 files changed, 58 insertions(+)

diff -puN mm/memrlimitcgroup.c~memrlimit-cgroup-fix-attach-task mm/memrlimitcgroup.c
--- linux-2.6.26-rc5/mm/memrlimitcgroup.c~memrlimit-cgroup-fix-attach-task 2008-06-19
22:17:46.000000000 +0530
+++ linux-2.6.26-rc5-balbir/mm/memrlimitcgroup.c 2008-06-19 23:48:27.000000000 +0530
@@ -166,6 +166,39 @@ static int memrlimit_cgroup_populate(str
     ARRAY_SIZE(memrlimit_cgroup_files));
 }

+static int memrlimit_cgroup_can_move_task(struct cgroup_subsys *ss,
+     struct cgroup *cgrp,
+     struct task_struct *p)
+{
+ struct mm_struct *mm;
+ struct memrlimit_cgroup *memrcg;
+ int ret = 0;
+
+ mm = get_task_mm(p);
+ if (mm == NULL)
+  return -EINVAL;
+
+ /*
+  * Hold mmap_sem, so that total_vm does not change underneath us
+  */
+ down_read(&mm->mmap_sem);
+
+ rcu_read_lock();
+ if (p != rcu_dereference(mm->owner))
+  goto out;
+
+ memrcg = memrlimit_cgroup_from_cgrp(cgrp);
+
+ if (!res_counter_add_check(&memrcg->as_res,
+    (mm->total_vm << PAGE_SHIFT)))
+  ret = -ENOMEM;
+out:
+ rcu_read_unlock();
+ up_read(&mm->mmap_sem);
+ mmput(mm);
+ return ret;
+}
+
 static void memrlimit_cgroup_move_task(struct cgroup_subsys *ss,
      struct cgroup *cgrp,
      struct cgroup *old_cgrp,
@@ -193,6 +226,12 @@ static void memrlimit_cgroup_move_task(s
```

```
  if (memrcg == old_memrcg)
   goto out;

+ /*
+  * NOTE: Even though we do the necessary checks in can_attach(),
+  * by the time we come here, there is a chance that we still
+  * fail (the memrlimit cgroup has grown its usage, and the
+  * addition of total_vm will no longer fit into its limit)
+  */
  if (res_counter_charge(&memrcg->as_res, (mm->total_vm << PAGE_SHIFT)))
   goto out;
  res_counter_uncharge(&old_memrcg->as_res, (mm->total_vm << PAGE_SHIFT));
@@ -231,6 +270,7 @@ struct cgroup_subsys memrlimit_cgroup_su
  .destroy = memrlimit_cgroup_destroy,
  .populate = memrlimit_cgroup_populate,
  .attach = memrlimit_cgroup_move_task,
+ .can_attach = memrlimit_cgroup_can_move_task,
  .mm_owner_changed = memrlimit_cgroup_mm_owner_changed,
  .early_init = 0,
 };
diff -puN kernel/res_counter.c~memrlimit-cgroup-fix-attach-task kernel/res_counter.c
diff -puN include/linux/res_counter.h~memrlimit-cgroup-fix-attach-task include/linux/res_counter.h
--- linux-2.6.26-rc5/include/linux/res_counter.h~memrlimit-cgroup-fix-attach-task 2008-06-19
22:52:17.000000000 +0530
+++ linux-2.6.26-rc5-balbir/include/linux/res_counter.h 2008-06-19 23:05:05.000000000 +0530
@@ -153,4 +153,22 @@ static inline void res_counter_reset_fai
  cnt->failcnt = 0;
  spin_unlock_irqrestore(&cnt->lock, flags);
 }
+
+/*
+ * Add the value val to the resource counter and check if we are
+ * still under the limit.
+ */
+static inline bool res_counter_add_check(struct res_counter *cnt,
+      unsigned long val)
+{
+ bool ret = false;
+ unsigned long flags;
+
+ spin_lock_irqsave(&cnt->lock, flags);
+ if (cnt->usage + val < cnt->limit)
+  ret = true;
+ spin_unlock_irqrestore(&cnt->lock, flags);
+ return ret;
+}
+
 #endif
```

–

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

_____

Containers mailing list
Containers@lists.linux-foundation.org
https://lists.linux-foundation.org/mailman/listinfo/containers