Subject: [RFC][PATCH] allow "unlimited" limit value.
Posted by KAMEZAWA Hiroyuki on Tue, 25 Sep 2007 10:39:00 GMT
View Forum Message <> Reply to Message

just for a RFC.

When I use memory controller, I notice that memory.limit_in_bytes shows
just very big number, if unlimited.

A user(or tool) has to know that the big number(LLONG_MAX) means "unlimted".
IMHO, some interface which allows users to specify "unlimited" value is helpful.

This patch tries to define value RES_COUTNER_UNLIMITED (== LLONG_MAX) and
modifies an interface to support "unlimted" value.

Because this patch breaks limit_in_bytes to some extent,
I'm glad if someone has a better idea to show unlimited value.
(if some easy value means "unlimited", it's helpful. LLONG_MAX is not easy
 to be recognized.)

==after this patch ==
[root@aworks kamezawa]# echo -n 400000000 > /opt/cgroup/memory.limit_in_bytes
[root@aworks kamezawa]# cat /opt/cgroup/memory.limit_in_bytes
400003072
[root@aworks kamezawa]# echo -n unlimited > /opt/cgroup/memory.limit_in_bytes
[root@aworks kamezawa]# cat /opt/cgroup/memory.limit_in_bytes
unlimited

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

 include/linux/res_counter.h | 1 +
 kernel/res_counter.c | 11 ++++++++---
 2 files changed, 9 insertions(+), 3 deletions(-)

Index: linux-2.6.23-rc8-mm1/include/linux/res_counter.h
===
--- linux-2.6.23-rc8-mm1.orig/include/linux/res_counter.h
+++ linux-2.6.23-rc8-mm1/include/linux/res_counter.h
@@ -28,6 +28,7 @@ struct res_counter {
   * the limit that usage cannot exceed
   */
  unsigned long long limit;
+#define RES_COUNTER_UNLIMITED ((unsigned long long)LLONG_MAX)
  /*
   * the number of unsuccessful attempts to consume the resource
   */

```
Index: linux-2.6.23-rc8-mm1/kernel/res_counter.c
===================================================================
--- linux-2.6.23-rc8-mm1.orig/kernel/res_counter.c
+++ linux-2.6.23-rc8-mm1/kernel/res_counter.c
@@ -16,7 +16,7 @@
 void res_counter_init(struct res_counter *counter)
 {
  spin_lock_init(&counter->lock);
- counter->limit = (unsigned long long)LLONG_MAX;
+ counter->limit = RES_COUNTER_UNLIMITED;
 }

 int res_counter_charge_locked(struct res_counter *counter, unsigned long val)
@@ -84,7 +84,9 @@ ssize_t res_counter_read(struct res_coun

  s = buf;
  val = res_counter_member(counter, member);
- if (read_strategy)
+ if (*val == RES_COUNTER_UNLIMITED) {
+  s += sprintf(s, "unlimited\n", *val);
+ } else if (read_strategy)
   s += read_strategy(*val, s);
  else
   s += sprintf(s, "%llu\n", *val);
@@ -112,7 +114,10 @@ ssize_t res_counter_write(struct res_cou

  ret = -EINVAL;

- if (write_strategy) {
+ if ((strcmp(buf, "-1") == 0) ||
+     (strcmp(buf,"unlimited") == 0)) {
+  tmp = RES_COUNTER_UNLIMITED;
+ } else if(write_strategy) {
   if (write_strategy(buf, &tmp)) {
    goto out_free;
   }
```

_____

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