
Subject: [RFC/PATCH 6/8]: CGroup Files: Remove cpuset_common_file_write()
Posted by [Paul Menage](#) on Tue, 13 May 2008 06:37:13 GMT
[View Forum Message](#) <> [Reply to Message](#)

This patch tweaks the signatures of the update_cpumask() and update_nodemask() functions so that they can be called directly as handlers for the new cgroups write_string() method.

This allows cpuset_common_file_write() to be removed.

Signed-off-by: Paul Menage <menage@google.com>

kernel/cpuset.c | 73 ++++++-----
1 file changed, 10 insertions(+), 63 deletions(-)

Index: cgroup-2.6.25-mm1/kernel/cpuset.c

=====

--- cgroup-2.6.25-mm1.orig/kernel/cpuset.c

+++ cgroup-2.6.25-mm1/kernel/cpuset.c

@@ -224,10 +224,6 @@ static struct cpuset top_cpuset = {
 * The task_struct fields mems_allowed and mems_generation may only
 * be accessed in the context of that task, so require no locks.
 *

- * The cpuset_common_file_write handler for operations that modify
- * the cpuset hierarchy holds cgroup_mutex across the entire operation,
- * single threading all such cpuset modifications across the system.
_ *

* The cpuset_common_file_read() handlers only hold callback_mutex across
* small pieces of code, such as when reading out possibly multi-word
* cpumasks and nodemasks.

@@ -747,8 +743,9 @@ static void cpuset_change_cpumask(struct
 * @cs: the cpuset to consider
 * @buf: buffer of cpu numbers written to this cpuset
 */

-static int update_cpumask(struct cpuset *cs, char *buf)
+static int update_cpumask(struct cgroup *cgrp, struct cftype *cft, char *buf)

{
+ struct cpuset *cs = cgroup_cs(cgrp);
 struct cpuset trialcs;
 struct cgroup_scanner scan;
 struct ptr_heap heap;

@@ -767,7 +764,6 @@ static int update_cpumask(struct cpuset
 * that parsing. The validate_change() call ensures that cpusets
 * with tasks have cpus.
 */

- buf = strstr(buf);
 if (!*buf) {

```

    cpus_clear(trialcs.cpus_allowed);
} else {
@@ -875,8 +871,9 @@ static void cpuset_migrate_mm(struct mm_

static void *cpuset_being_rebound;

-static int update_nodemask(struct cpuset *cs, char *buf)
+static int update_nodemask(struct cgroup *cgrp, struct cftype *cft, char *buf)
{
+ struct cpuset *cs = cgroup_cs(cgrp);
  struct cpuset trialcs;
  nodemask_t oldmem;
  struct task_struct *p;
@@ -902,7 +899,6 @@ static int update_nodemask(struct cpuset
  * that parsing. The validate_change() call ensures that cpusets
  * with tasks have memory.
  */
- buf = strstr(buf);
  if (!*buf) {
    nodes_clear(trialcs.mems_allowed);
  } else {
@@ -1201,59 +1197,6 @@ typedef enum {
  FILE_SPREAD_SLAB,
} cpuset_filetype_t;

-static ssize_t cpuset_common_file_write(struct cgroup *cont,
-    struct cftype *cft,
-    struct file *file,
-    const char __user *userbuf,
-    size_t nbytes, loff_t *unused_ppos)
-{
- struct cpuset *cs = cgroup_cs(cont);
- cpuset_filetype_t type = cft->private;
- char *buffer;
- int retval = 0;
-
- /* Crude upper limit on largest legitimate cpulist user might write. */
- if (nbytes > 100U + 6 * max(NR_CPUS, MAX_NUMNODES))
- return -E2BIG;
-
- /* +1 for nul-terminator */
- if ((buffer = kmalloc(nbytes + 1, GFP_KERNEL)) == 0)
- return -ENOMEM;
-
- if (copy_from_user(buffer, userbuf, nbytes)) {
- retval = -EFAULT;
- goto out1;
- }

```

```

- buffer[nbytes] = 0; /* nul-terminate */
-
- cgroup_lock();
-
- if (cgroup_is_removed(cont)) {
-     retval = -ENODEV;
-     goto out2;
- }
-
- switch (type) {
- case FILE_CPULIST:
-     retval = update_cpumask(cs, buffer);
-     break;
- case FILE_MEMLIST:
-     retval = update_nodemask(cs, buffer);
-     break;
- default:
-     retval = -EINVAL;
-     goto out2;
- }
-
- if (retval == 0)
-     retval = nbytes;
-out2:
- cgroup_unlock();
-out1:
- kfree(buffer);
- return retval;
-}
-
static int cpuset_write_u64(struct cgroup *cgrp, struct cftype *cft, u64 val)
{
    int retval = 0;
@@ -1412,14 +1355,18 @@ static struct cftype files[] = {
    {
        .name = "cpus",
        .read = cpuset_common_file_read,
-     .write = cpuset_common_file_write,
+     .write_string = update_cpumask,
+     .max_write_len = (100U + 6 * NR_CPUS),
+     .lockmode = CFT_LOCK_CGL_WRITE,
        .private = FILE_CPULIST,
    },

    {
        .name = "mems",
        .read = cpuset_common_file_read,
-     .write = cpuset_common_file_write,

```

```
+ .write_string = update_nodemask,  
+ .max_write_len = (100U + 6 * MAX_NUMNODES),  
+ .lockmode = CFT_LOCK_CGL_WRITE,  
  .private = FILE_MEMLIST,  
},
```

--

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