

---

Subject: [PATCH 1/2] CGroups \_s64 files: Add cgroups read\_s64/write\_s64 file methods

Posted by [Paul Menage](#) on Wed, 05 Mar 2008 10:20:26 GMT

[View Forum Message](#) <> [Reply to Message](#)

---

These are the signed equivalents of the read\_u64/write\_u64 methods

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

---

```
include/linux/cgroup.h | 8 ++++++++
kernel/cgroup.c        | 38 +++++++++++++++++++++++++++++++++++++-----
2 files changed, 36 insertions(+), 10 deletions(-)
```

Index: read\_s64-2.6.25-rc3-mm1/include/linux/cgroup.h

```
=====
--- read_s64-2.6.25-rc3-mm1.orig/include/linux/cgroup.h
+++ read_s64-2.6.25-rc3-mm1/include/linux/cgroup.h
@@ -216,6 +216,10 @@ struct cftype {
    */
    u64 (*read_u64) (struct cgroup *cgrp, struct cftype *cft);
    /*
+ * read_s64() is a signed version of read_u64()
+ */
+ s64 (*read_s64) (struct cgroup *cgrp, struct cftype *cft);
+ /*
    * read_map() is used for defining a map of key/value
    * pairs. It should call cb->fill(cb, key, value) for each
    * entry. The key/value pairs (and their ordering) should not
@@ -234,6 +238,10 @@ struct cftype {
    * userspace. Use in place of write(); return 0 or error.
    */
    int (*write_u64) (struct cgroup *cgrp, struct cftype *cft, u64 val);
+ /*
+ * write_s64() is a signed version of write_u64()
+ */
+ int (*write_s64) (struct cgroup *cgrp, struct cftype *cft, s64 val);

    int (*release) (struct inode *inode, struct file *file);
};
```

Index: read\_s64-2.6.25-rc3-mm1/kernel/cgroup.c

```
=====
--- read_s64-2.6.25-rc3-mm1.orig/kernel/cgroup.c
+++ read_s64-2.6.25-rc3-mm1/kernel/cgroup.c
@@ -1291,14 +1291,13 @@ enum cgroup_filetype {
    FILE_RELEASE_AGENT,
};
```

```

-static ssize_t cgroup_write_u64(struct cgroup *cgrp, struct cftype *cft,
+static ssize_t cgroup_write_X64(struct cgroup *cgrp, struct cftype *cft,
    struct file *file,
    const char __user *userbuf,
    size_t nbytes, loff_t *unused_ppos)
{
    char buffer[64];
    int retval = 0;
- u64 val;
    char *end;

    if (!nbytes)
@@ -1310,12 +1309,17 @@ static ssize_t cgroup_write_u64(struct c

    buffer[nbytes] = 0;    /* nul-terminate */
    strstrip(buffer);
- val = simple_strtoull(buffer, &end, 0);
- if (*end)
- return -EINVAL;
-
- /* Pass to subsystem */
- retval = cft->write_u64(cgrp, cft, val);
+ if (cft->write_u64) {
+ u64 val = simple_strtoull(buffer, &end, 0);
+ if (*end)
+ return -EINVAL;
+ retval = cft->write_u64(cgrp, cft, val);
+ } else {
+ s64 val = simple_strtoll(buffer, &end, 0);
+ if (*end)
+ return -EINVAL;
+ retval = cft->write_s64(cgrp, cft, val);
+ }
    if (!retval)
        retval = nbytes;
    return retval;
@@ -1396,8 +1400,8 @@ static ssize_t cgroup_file_write(struct
    return -ENODEV;
    if (cft->write)
        return cft->write(cgrp, cft, file, buf, nbytes, ppos);
- if (cft->write_u64)
- return cgroup_write_u64(cgrp, cft, file, buf, nbytes, ppos);
+ if (cft->write_u64 || cft->write_s64)
+ return cgroup_write_X64(cgrp, cft, file, buf, nbytes, ppos);
    return -EINVAL;
}

@@ -1413,6 +1417,18 @@ static ssize_t cgroup_read_u64(struct cg

```

```

    return simple_read_from_buffer(buf, nbytes, ppos, tmp, len);
}

+static ssize_t cgroup_read_s64(struct cgroup *cgrp, struct cftype *cft,
+    struct file *file,
+    char __user *buf, size_t nbytes,
+    loff_t *ppos)
+{
+ char tmp[64];
+ s64 val = cft->read_s64(cgrp, cft);
+ int len = sprintf(tmp, "%lld\n", (long long) val);
+
+ return simple_read_from_buffer(buf, nbytes, ppos, tmp, len);
+}
+
static ssize_t cgroup_common_file_read(struct cgroup *cgrp,
    struct cftype *cft,
    struct file *file,
@@ -1469,6 +1485,8 @@ static ssize_t cgroup_file_read(struct f
    return cft->read(cgrp, cft, file, buf, nbytes, ppos);
    if (cft->read_u64)
        return cgroup_read_u64(cgrp, cft, file, buf, nbytes, ppos);
+ if (cft->read_s64)
+ return cgroup_read_s64(cgrp, cft, file, buf, nbytes, ppos);
    return -EINVAL;
}

--

```

---

Containers mailing list

Containers@lists.linux-foundation.org

<https://lists.linux-foundation.org/mailman/listinfo/containers>

---