
Subject: [PATCH] Add a 'trigger' callback on struct cftype.
Posted by [Pavel Emelianov](#) on Tue, 11 Mar 2008 15:21:19 GMT
[View Forum Message](#) <> [Reply to Message](#)

If the patch with max_usage for res_counter will be accepted we'll have two :) files, that a event-triggers essentially, i.e. they don't care what the user actually write to then, but are interested in the writing by its own.

So the proposal is to make cgroups infrastructure handle this case.

Signed-off-by: Pavel Emelyanov <xemul@openvz.org>

```
---
diff --git a/include/linux/cgroup.h b/include/linux/cgroup.h
index 785a01c..df579e3 100644
--- a/include/linux/cgroup.h
+++ b/include/linux/cgroup.h
@@ -243,6 +243,7 @@ struct cftype {
    */
    int (*write_s64) (struct cgroup *cgrp, struct cftype *cft, s64 val);

+ void (*trigger) (struct cgroup *cgrp, unsigned int event);
    int (*release) (struct inode *inode, struct file *file);
};

diff --git a/kernel/cgroup.c b/kernel/cgroup.c
index e8e8ec4..7d73c2b 100644
--- a/kernel/cgroup.c
+++ b/kernel/cgroup.c
@@ -1410,6 +1410,10 @@ static ssize_t cgroup_file_write(struct file *file, const char __user *buf,
    return cft->write(cgrp, cft, file, buf, nbytes, ppos);
    if (cft->write_u64 || cft->write_s64)
        return cgroup_write_X64(cgrp, cft, file, buf, nbytes, ppos);
+ if (cft->trigger) {
+ cft->trigger(cgrp, (unsigned int)cft->private);
+ return nbytes;
+ }
    return -EINVAL;
}

diff --git a/mm/memcontrol.c b/mm/memcontrol.c
index c27141d..4c1d24c 100644
--- a/mm/memcontrol.c
+++ b/mm/memcontrol.c
@@ -868,27 +868,14 @@ static ssize_t mem_cgroup_write(struct cgroup *cont, struct cftype *cft,
    mem_cgroup_write_strategy);
```

```

}

-static ssize_t mem_cgroup_max_reset(struct cgroup *cont, struct cftype *cft,
- struct file *file, const char __user *userbuf,
- size_t nbytes, loff_t *ppos)
+static void mem_cgroup_max_reset(struct cgroup *cont, unsigned int event)
{
- struct mem_cgroup *mem;
-
- mem = mem_cgroup_from_cont(cont);
- res_counter_reset_max(&mem->res);
- return nbytes;
+ res_counter_reset_max(&mem_cgroup_from_cont(cont)->res);
}

-static ssize_t mem_force_empty_write(struct cgroup *cont,
- struct cftype *cft, struct file *file,
- const char __user *userbuf,
- size_t nbytes, loff_t *ppos)
+static void mem_force_empty_write(struct cgroup *cont, unsigned int event)
{
- struct mem_cgroup *mem = mem_cgroup_from_cont(cont);
- int ret = mem_cgroup_force_empty(mem);
- if (!ret)
- ret = nbytes;
- return ret;
+ mem_cgroup_force_empty(mem_cgroup_from_cont(cont));
}

static const struct mem_cgroup_stat_desc {
@@ -936,7 +923,7 @@ static struct cftype mem_cgroup_files[] = {
{
.name = "max_usage_in_bytes",
.private = RES_MAX_USAGE,
- .write = mem_cgroup_max_reset,
+ .trigger = mem_cgroup_max_reset,
.read_u64 = mem_cgroup_read,
},
{
@@ -952,7 +939,7 @@ static struct cftype mem_cgroup_files[] = {
},
{
.name = "force_empty",
- .write = mem_force_empty_write,
+ .trigger = mem_force_empty_write,
},
{
.name = "stat",

```

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