Subject: [PATCH] [RFC][patch 4/12][CFQ-cgroup] Add ioprio entry
Posted by Satoshi UCHIDA on Thu, 03 Apr 2008 07:13:21 GMT
View Forum Message <> Reply to Message

This patch add "ioprio" entry in cfq_cgroup.
The "ioprio" entry shows I/O priority of group.
When you would change priority, write this entry.

Signed-off-by: Satoshi UCHIDA <uchida@ap.jp.nec.com>

---
block/cfq-cgroup.c | 96 ++
1 files changed, 96 insertions(+), 0 deletions(-)

```
diff --git a/block/cfq-cgroup.c b/block/cfq-cgroup.c
index de00a0d..378a23d 100644
--- a/block/cfq-cgroup.c
+++ b/block/cfq-cgroup.c
@@ -15,8 +15,12 @@
 #include <linux/cgroup.h>
 #include <linux/cfq-iosched.h>

+#define CFQ_CGROUP_SLICE_SCALE  (5)
+#define CFQ_CGROUP_MAX_IOPRIO  (8)
+
 struct cfq_cgroup {
  struct cgroup_subsys_state css;
+ unsigned int ioprio;
 };


@@ -41,6 +45,8 @@ cfq_cgroup_create(struct cgroup_subsys *ss, struct cgroup *cont)
  if (unlikely(!cfqc))
   return ERR_PTR(-ENOMEM);

+ cfqc->ioprio = 3;
+
  return &cfqc->css;
 }

@@ -49,9 +55,99 @@ static void cfq_cgroup_destroy(struct cgroup_subsys *ss, struct cgroup
*cont)
  kfree(cgroup_to_cfq_cgroup(cont));
 }

+static ssize_t cfq_cgroup_read(struct cgroup *cont, struct cftype *cft,
+         struct file *file, char __user *userbuf,
+         size_t nbytes, loff_t *ppos)
```

```
+{
+ struct cfq_cgroup *cfqc;
+ char *page;
+ ssize_t ret;
+
+ page = (char *)__get_free_page(GFP_TEMPORARY);
+ if (!page)
+  return -ENOMEM;
+
+ cgroup_lock();
+ if (cgroup_is_removed(cont)) {
+  cgroup_unlock();
+  ret = -ENODEV;
+  goto out;
+ }
+
+ cfqc = cgroup_to_cfq_cgroup(cont);
+
+ cgroup_unlock();
+
+ /* print priority */
+ ret = sprintf(page, " priority: %d \n", cfqc->ioprio);
+
+ ret = simple_read_from_buffer(userbuf, nbytes, ppos, page, ret);
+
+out:
+ free_page((unsigned long)page);
+ return ret;
+}
+
+static ssize_t cfq_cgroup_write(struct cgroup *cont, struct cftype *cft,
+    struct file *file, const char __user *userbuf,
+    size_t nbytes, loff_t *ppos)
+{
+ struct cfq_cgroup *cfqc;
+ ssize_t ret;
+ int new_prio;
+ char *buffer = NULL;
+
+ cgroup_lock();
+ if (cgroup_is_removed(cont)) {
+  cgroup_unlock();
+  ret = -ENODEV;
+  goto out;
+ }
+
+ cfqc = cgroup_to_cfq_cgroup(cont);
+ cgroup_unlock();
```

```
+
+ /* set priority */
+ if ((buffer = kmalloc(nbytes + 1, GFP_KERNEL)) == 0)
+  return -ENOMEM;
+
+ if (copy_from_user(buffer, userbuf, nbytes)) {
+  ret = -EFAULT;
+  goto out;
+ }
+ buffer[nbytes]=0;
+
+ new_prio = simple_strtoul(buffer, NULL, 10);
+ if ((new_prio < 0) || (new_prio > CFQ_CGROUP_MAX_IOPRIO)) {
+  ret = -EINVAL;
+  goto out;
+ }
+
+ cfqc->ioprio = new_prio;
+ ret = nbytes;
+
+out:
+ kfree(buffer);
+ return ret;
+}
+
+static struct cftype files[] = {
+ {
+  .name = "ioprio",
+  .read = cfq_cgroup_read,
+  .write = cfq_cgroup_write,
+ },
+};
+
+static int cfq_cgroup_populate(struct cgroup_subsys *ss, struct cgroup *cont)
+{
+ return cgroup_add_files(cont, ss, files, ARRAY_SIZE(files));
+}
+
 struct cgroup_subsys cfq_subsys = {
   .name = "cfq",
   .create = cfq_cgroup_create,
   .destroy = cfq_cgroup_destroy,
   .subsys_id = cfq_subsys_id,
+  .populate = cfq_cgroup_populate,
 };
--
1.5.4.1
```