
Subject: [RFC][PATCH 4/9] cgroups: block: cfq: I/O bandwidth controlling subsystem for CGroups based on CFQ

Posted by [Vasily Tarasov](#) on Fri, 15 Feb 2008 06:59:47 GMT

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

From: Vasily Tarasov <vtaras@openvz.org>

Adds hooks to CFQ code to form proper data structures interconnection.

Signed-off-by: Vasily Tarasov <vtaras@openvz.org>

--- linux-2.6.25-rc5-mm1/include/linux/cfqio-cgroup.h.creat-elim 2008-02-15 01:07:29.000000000 -0500

+++ linux-2.6.25-rc5-mm1/include/linux/cfqio-cgroup.h 2008-02-15 01:08:25.000000000 -0500

```
@@ -36,6 +36,12 @@ struct cfqio_ss_css {
extern struct cfqio_cgroup_data *
cfqio_cgrp_findcreate(struct cfqio_ss_css *, struct cfq_data *, gfp_t gfp_mask);
extern void cfqio_ss_exit_queue(struct cfq_data *);
```

+

```
+static inline struct cfqio_ss_css *cfqio_css_by_tsk(struct task_struct *tsk)
```

```
+{
```

```
+ return container_of(tsk->cgroups->subsys[cfqio_subsys_id],
```

```
+ struct cfqio_ss_css, css);
```

```
+}
```

```
#else
```

```
static inline struct cfqio_cgroup_data *
```

```
cfqio_cgrp_findcreate(struct cfqio_ss_css *cfqio_ss,
```

```
@@ -45,6 +51,11 @@ cfqio_cgrp_findcreate(struct cfqio_ss_cs
```

```
}
```

```
extern void cfqio_ss_exit_queue(struct cfq_data *cfqd) { ; }
```

+

```
+static inline struct cfqio_ss_css *cfqio_css_by_tsk(struct task_struct *tsk)
```

```
+{
```

```
+ return NULL;
```

```
+}
```

```
#endif /* CONFIG_CGROUP_CFQIO */
```

```
static inline void cfqio_init_cfqio_cgroup(struct cfqio_cgroup_data *cfqio_cgrp)
```

--- linux-2.6.25-rc5-mm1/block/cfq-iosched.c.creat-elim 2008-02-15 01:03:38.000000000 -0500

+++ linux-2.6.25-rc5-mm1/block/cfq-iosched.c 2008-02-15 01:08:25.000000000 -0500

```
@@ -12,6 +12,8 @@
```

```
#include <linux/rbtree.h>
```

```
#include <linux/ioprio.h>
```

```
#include <linux/cfq-iosched.h>
```

```
+#include <linux/cgroup.h>
```

```

+#include <linux/cfqio-cgroup.h>

/*
 * tunables
@@ -1256,11 +1258,14 @@ cfq_find_alloc_queue(struct cfq_data *cf
{
    struct cfq_queue *cfqq, *new_cfqq = NULL;
    struct cfq_io_context *cic;
+ struct cfqio_cgroup_data *cfqio_cgrp = NULL;
+ struct cfqio_ss_css *cfqio_css;

    retry:
    cic = cfq_cic_lookup(cfqd, ioc);
    /* cic always exists here */
    cfqq = cic_to_cfqq(cic, is_sync);
+ cfqio_css = cfqio_css_by_tsk(current);

    if (!cfqq) {
        if (new_cfqq) {
@@ -1277,6 +1282,14 @@ retry:
            new_cfqq = kmem_cache_alloc_node(cfq_pool,
                gfp_mask | __GFP_NOFAIL | __GFP_ZERO,
                cfqd->queue->node);
+         if (new_cfqq) {
+             cfqio_cgrp = cfqio_cgrp_findcreate(cfqio_css,
+                 cfqd, gfp_mask);
+             if (!cfqio_cgrp) {
+                 kmem_cache_free(cfq_pool, new_cfqq);
+                 new_cfqq = NULL;
+             }
+         }
        spin_lock_irq(cfqd->queue->queue_lock);
        goto retry;
    } else {
@@ -1285,6 +1298,13 @@ retry:
        cfqd->queue->node);
        if (!cfqq)
            goto out;
+
+         cfqio_cgrp = cfqio_cgrp_findcreate(cfqio_css,
+             cfqd, gfp_mask);
+         if (!cfqio_cgrp) {
+             kmem_cache_free(cfq_pool, cfqq);
+             cfqq = NULL;
+         }
    }

    RB_CLEAR_NODE(&cfqq->rb_node);

```

@@ -1298,6 +1318,8 @@ retry:

```
    cfq_init_prio_data(cfqq, ioc);
```

```
+ cfqq->cfqio_cgrp = cfqio_cgrp;
```

```
+
```

```
    if (is_sync) {
```

```
        if (!cfq_class_idle(cfqq))
```

```
            cfq_mark_cfqq_idle_window(cfqq);
```

@@ -1990,6 +2012,8 @@ static void cfq_exit_queue(elevator_t *e

```
    cfq_shutdown_timer_wq(cfqd);
```

```
+ cfqio_ss_exit_queue(cfqd);
```

```
+
```

```
    kfree(cfqd);
```

```
}
```

@@ -2002,6 +2026,9 @@ static void *cfq_init_queue(struct reque
 return NULL;

```
    cfqd->service_tree = CFQ_RB_ROOT;
```

```
+ #ifndef CONFIG_CGROUP_CFQIO
```

```
+ cfqio_init_cfqio_cgroup(&cfqd->cfqio_cgroup);
```

```
+ #endif
```

```
    INIT_LIST_HEAD(&cfqd->cic_list);
```

```
    cfqd->queue = q;
```

@@ -2022,6 +2049,7 @@ static void *cfq_init_queue(struct reque

```
    cfqd->cfq_slice[1] = cfq_slice_sync;
```

```
    cfqd->cfq_slice_async_rq = cfq_slice_async_rq;
```

```
    cfqd->cfq_slice_idle = cfq_slice_idle;
```

```
+ INIT_LIST_HEAD(&cfqd->act_cfqio_cgrp_head);
```

```
    return cfqd;
```

```
}
```