Subject: [RFC][PATCH 6/9] cgroups: block: cfq: I/O bandwidth controlling subsystem for CGroups based on CFQ
Posted by Vasily Tarasov on Fri, 15 Feb 2008 06:59:51 GMT
View Forum Message <> Reply to Message

From: Vasily Tarasov <vtaras@openvz.org>

Introduces cgroups scheduling. Each time when I/O request is placed on
per-process request queue and there were no other requests from this
cgroup, this cgroup is added to the end of active cgroups list.
This list is service in round-robin fashion. Switching between
cgroups happens eiher when cgroup expires its time slice, either if there
are no more requests from it. Each time I/O request is completed,
we check if it was the last request from cgroup, and in this case remove
cgroup from the active list.

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

```
--- linux-2.6.25-rc5-mm1/include/linux/cfqio-cgroup.h.cgrpsched 2008-02-15 01:08:25.000000000
-0500
+++ linux-2.6.25-rc5-mm1/include/linux/cfqio-cgroup.h 2008-02-15 01:10:42.000000000 -0500
@@ -42,6 +42,10 @@ static inline struct cfqio_ss_css *cfqio
  return container_of(tsk->cgroups->subsys[cfqio_subsys_id],
      struct cfqio_ss_css, css);
 }
+extern int cfqio_cgrp_expired(struct cfq_data *);
+extern void cfqio_cgrp_schedule_active(struct cfq_data *);
+extern void  cfqio_cgrp_inc_rqnum(struct cfq_queue *);
+extern void cfqio_cgrp_dec_rqnum(struct cfq_queue *);
 #else
 static inline struct cfqio_cgroup_data *
 cfqio_cgrp_findcreate(struct cfqio_ss_css *cfqio_ss,
@@ -56,6 +60,16 @@ static inline struct cfqio_ss_css *cfqio
 {
  return NULL;
 }
+
+static inline int cfqio_cgrp_expired(struct cfq_data *cfqd) { return 0; }
+
+static inline void cfqio_cgrp_schedule_active(struct cfq_data *cfqd)
+{
+ cfqd->active_cfqio_cgroup = &cfqd->cfqio_cgroup;
+}
+
+static inline void cfqio_cgrp_inc_rqnum(struct cfq_queue *cfqq) { ; }
+static inline void cfqio_cgrp_dec_rqnum(struct cfq_queue *cfqq) { ; }
```

```
 #endif /* CONFIG_CGROUP_CFQIO */

 static inline void cfqio_init_cfqio_cgroup(struct cfqio_cgroup_data *cfqio_cgrp)
--- linux-2.6.25-rc5-mm1/include/linux/cfq-iosched.h.cgrpsched 2008-02-15 01:09:09.000000000
-0500
+++ linux-2.6.25-rc5-mm1/include/linux/cfq-iosched.h 2008-02-15 01:10:42.000000000 -0500
@@ -56,6 +56,7 @@ struct cfqio_cgroup_data {
  struct cfqio_ss_css *cfqio_css;
  /* rr list of queues with requests */
  struct cfq_rb_root service_tree;
+ unsigned long  rqnum;
 };

 /*
@@ -113,6 +114,8 @@ struct cfq_data {
  struct list_head act_cfqio_cgrp_head;
  /* cgroup that owns a timeslice at the moment */
  struct cfqio_cgroup_data *active_cfqio_cgroup;
+ unsigned int cfqio_cgrp_slice;
+ unsigned long cfqio_slice_end;
 };

 /*
--- linux-2.6.25-rc5-mm1/block/cfqio-cgroup.c.cgrpsched 2008-02-15 01:07:29.000000000 -0500
+++ linux-2.6.25-rc5-mm1/block/cfqio-cgroup.c 2008-02-15 01:10:42.000000000 -0500
@@ -24,6 +24,65 @@ LIST_HEAD(cfqio_ss_css_head);
  */
 DEFINE_SPINLOCK(cfqio_ss_css_lock);

+int cfqio_cgrp_expired(struct cfq_data *cfqd)
+{
+ return time_after(jiffies, cfqd->cfqio_slice_end) ?  1 : 0;
+}
+
+static inline unsigned long time_slice_by_ioprio(unsigned int ioprio,
+       unsigned int base_slice)
+{
+ return  base_slice +
+  (base_slice * (ioprio - CFQIO_SS_IOPRIO_MIN))
+   / (CFQIO_SS_IOPRIO_MAX - CFQIO_SS_IOPRIO_MIN);
+}
+
+static inline void set_active_cgrp(struct cfq_data *cfqd)
+{
+ if (list_empty(&cfqd->act_cfqio_cgrp_head))
+  return;
+
+ cfqd->active_cfqio_cgroup = list_first_entry(&cfqd->act_cfqio_cgrp_head,
```

```
+    struct cfqio_cgroup_data, act_cfqio_cgrp_list);
+ list_move_tail(&cfqd->active_cfqio_cgroup->act_cfqio_cgrp_list,
+     &cfqd->act_cfqio_cgrp_head);
+ cfqd->cfqio_slice_end = jiffies +
+ time_slice_by_ioprio(cfqd->active_cfqio_cgroup->cfqio_css->ioprio,
+       cfqd->cfqio_cgrp_slice);
+}
+
+void cfqio_cgrp_schedule_active(struct cfq_data *cfqd)
+{
+ if (cfqio_cgrp_expired(cfqd) || !cfqd->active_cfqio_cgroup ||
+    !cfqd->active_cfqio_cgroup->rqnum)
+  set_active_cgrp(cfqd);
+}
+
+void cfqio_cgrp_inc_rqnum(struct cfq_queue *cfqq)
+{
+ struct cfqio_cgroup_data *cfqio_cgrp;
+
+ cfqio_cgrp = cfqq->cfqio_cgrp;
+
+ if (!cfqio_cgrp->rqnum)
+  list_add_tail(&cfqio_cgrp->act_cfqio_cgrp_list,
+    &cfqq->cfqd->act_cfqio_cgrp_head);
+
+ cfqio_cgrp->rqnum++;
+}
+
+void cfqio_cgrp_dec_rqnum(struct cfq_queue *cfqq)
+{
+ struct cfqio_cgroup_data *cfqio_cgrp;
+
+ cfqio_cgrp = cfqq->cfqio_cgrp;
+
+ cfqio_cgrp->rqnum--;
+
+ if (!cfqio_cgrp->rqnum)
+  list_del(&cfqio_cgrp->act_cfqio_cgrp_list);
+}
+
 static struct cfqio_cgroup_data *
 __find_cfqio_cgrp(struct cfqio_ss_css *cfqio_css, struct cfq_data *cfqd)
 {
--- linux-2.6.25-rc5-mm1/block/cfq-iosched.c.cgrpsched 2008-02-15 01:09:09.000000000 -0500
+++ linux-2.6.25-rc5-mm1/block/cfq-iosched.c 2008-02-15 01:10:42.000000000 -0500
@@ -29,6 +29,7 @@ static const int cfq_slice_sync = HZ / 1
 static int cfq_slice_async = HZ / 25;
 static const int cfq_slice_async_rq = 2;
```

```
 static int cfq_slice_idle = HZ / 125;
+static int cfqio_cgrp_slice = HZ / 2;

 /*
  * offset from end of service tree
@@ -185,6 +186,8 @@ static inline int cfq_slice_used(struct
 {
  if (cfq_cfqq_slice_new(cfqq))
   return 0;
+ if (cfqio_cgrp_expired(cfqq->cfqd))
+  return 1;
  if (time_before(jiffies, cfqq->slice_end))
   return 0;

@@ -447,6 +450,7 @@ static void cfq_add_cfqq_rr(struct cfq_d
  BUG_ON(cfq_cfqq_on_rr(cfqq));
  cfq_mark_cfqq_on_rr(cfqq);
  cfqd->busy_queues++;
+ cfqio_cgrp_inc_rqnum(cfqq);

  cfq_resort_rr_list(cfqd, cfqq);
 }
@@ -466,6 +470,7 @@ static void cfq_del_cfqq_rr(struct cfq_d

  BUG_ON(!cfqd->busy_queues);
  cfqd->busy_queues--;
+ cfqio_cgrp_dec_rqnum(cfqq);
 }

 /*
@@ -708,6 +713,8 @@ static struct cfq_queue *cfq_get_next_qu
 {
  struct cfqio_cgroup_data *cfqio_cgrp;

+ cfqio_cgrp_schedule_active(cfqd);
+
  cfqio_cgrp = cfqd->active_cfqio_cgroup;
  if (!cfqio_cgrp)
   return NULL;
@@ -2076,6 +2083,7 @@ static void *cfq_init_queue(struct reque
  cfqd->cfq_slice_async_rq = cfq_slice_async_rq;
  cfqd->cfq_slice_idle = cfq_slice_idle;
  INIT_LIST_HEAD(&cfqd->act_cfqio_cgrp_head);
+ cfqd->cfqio_cgrp_slice = cfqio_cgrp_slice;

  return cfqd;
 }
```