

---

Subject: [RFC][mm] [1/2] Simple stats for cpu resource controller

Posted by [Balaji Rao](#) on Wed, 26 Mar 2008 18:18:45 GMT

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

---

This patch implements trivial statistics for the cpu controller.

Signed-off-by: Balaji Rao <balajirao@gmail.com>

CC: Balbir Singh <balbir@linux.vnet.ibm.com>

CC: Dhaval Giani <dhaval@linux.vnet.ibm.com>

diff --git a/kernel/sched.c b/kernel/sched.c

index 9fbfa05..eac9333 100644

--- a/kernel/sched.c

+++ b/kernel/sched.c

@@ -164,10 +164,38 @@ struct cfs\_rq;

static LIST\_HEAD(task\_groups);

+#ifdef CONFIG\_CGROUP\_SCHED

+enum cpu\_cgroup\_stat\_index {

+ CPU\_CGROUP\_STAT\_UTIME, /\* Usertime of the task group \*/

+ CPU\_CGROUP\_STAT\_STIME, /\* Kerntime of the task group \*/

+

+ CPU\_CGROUP\_STAT\_NSTATS,

+};

+

+struct cpu\_cgroup\_stat\_cpu {

+ s64 count[CPU\_CGROUP\_STAT\_NSTATS];

+} \_\_\_\_cacheline\_aligned\_in\_smp;

+

+struct cpu\_cgroup\_stat {

+ struct cpu\_cgroup\_stat\_cpu cpustat[NR\_CPUS];

+};

+

+/\* Called under irq disable. \*/

+static void \_\_cpu\_cgroup\_stat\_add\_safe(struct cpu\_cgroup\_stat \*stat,

+ enum cpu\_cgroup\_stat\_index idx, int val)

+{

+ int cpu = smp\_processor\_id();

+

+ BUG\_ON(!irqs\_disabled());

+ stat->cpustat[cpu].count[idx] += val;

+}

+#endif

+

/\* task group related information \*/

struct task\_group {

#ifdef CONFIG\_CGROUP\_SCHED

```

    struct cgroup_subsys_state css;
+ struct cpu_cgroup_stat stat;
#endif

#ifdef CONFIG_FAIR_GROUP_SCHED
@@ -3670,6 +3698,16 @@ void account_user_time(struct task_struct *p, cputime_t cputime)
    cpustat->nice = cputime64_add(cpustat->nice, tmp);
    else
    cpustat->user = cputime64_add(cpustat->user, tmp);
+
+ /* Charge the task's group */
+ #ifdef CONFIG_CGROUP_SCHED
+ {
+ struct task_group *tg;
+ tg = task_group(p);
+ __cpu_cgroup_stat_add_safe(&tg->stat, CPU_CGROUP_STAT_UTIME,
+ cputime_to_msecs(cputime));
+ }
+ #endif
+ }

/*
@@ -3733,6 +3771,15 @@ void account_system_time(struct task_struct *p, int hardirq_offset,
    cpustat->idle = cputime64_add(cpustat->idle, tmp);
    /* Account for system time used */
    acct_update_integrals(p);
+
+ #ifdef CONFIG_CGROUP_SCHED
+ {
+ struct task_group *tg;
+ tg = task_group(p);
+ __cpu_cgroup_stat_add_safe(&tg->stat, CPU_CGROUP_STAT_STIME,
+ cputime_to_msecs(cputime));
+ }
+ #endif
+ }

/*
@@ -7939,6 +7986,40 @@ static u64 cpu_shares_read_u64(struct cgroup *cgrp, struct cftype
*cft)

    return (u64) tg->shares;
+ }
+
+ static s64 cpu_cgroup_read_stat(struct cpu_cgroup_stat *stat,
+ enum cpu_cgroup_stat_index idx)
+ {
+ int cpu;

```

```

+ s64 ret = 0;
+ for_each_possible_cpu(cpu)
+ ret += stat->cpustat[cpu].count[idx];
+ return ret;
+}
+
+static const struct cpu_cgroup_stat_desc {
+ const char *msg;
+ u64 unit;
+} cpu_cgroup_stat_desc[] = {
+ [CPU_CGROUP_STAT_UTIME] = { "utime", 1, },
+ [CPU_CGROUP_STAT_STIME] = { "stime", 1, },
+};
+
+static int cpu_cgroup_stats_show(struct cgroup *cgrp, struct cftype *cft,
+ struct cgroup_map_cb *cb)
+{
+ struct task_group *tg = cgroup_tg(cgrp);
+ struct cpu_cgroup_stat *stat = &tg->stat;
+ int i;
+
+ for (i = 0; i < ARRAY_SIZE(stat->cpustat[0].count); i++) {
+ s64 val;
+ val = cpu_cgroup_read_stat(stat, i);
+ val *= cpu_cgroup_stat_desc[i].unit;
+ cb->fill(cb, cpu_cgroup_stat_desc[i].msg, val);
+ }
+ return 0;
+}
+
+endif

#ifdef CONFIG_RT_GROUP_SCHED
@@ -7961,6 +8042,11 @@ static struct cftype cpu_files[] = {
    .read_u64 = cpu_shares_read_u64,
    .write_u64 = cpu_shares_write_u64,
    },
+
+ {
+ .name = "stat",
+ .read_map = cpu_cgroup_stats_show,
+ },
+
+endif
#ifdef CONFIG_RT_GROUP_SCHED
{

```

---

Containers mailing list  
Containers@lists.linux-foundation.org

