
Subject: Re: [RFC][mm] Simple stats for cpu resource controller v4
Posted by [Balaji Rao](#) on Sun, 11 May 2008 20:18:37 GMT
[View Forum Message](#) <> [Reply to Message](#)

Hi Andrew,

Here's a version that uses percpu_counters and which actually works. The only evil it contains is the check,

```
if (percpu_counter_ready) {  
    ..  
}
```

I've also added a callback called 'inititalize' to a cgroup_subsys.

The alternate idea I suggested was to add a hook at a point where the first task has not yet started and where kmalloc is usable. Its hacky and not a nice thing to do IMHO.

How does this look ?

Regards,
Balaji Rao

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

```
diff --git a/include/linux/cgroup.h b/include/linux/cgroup.h  
index e155aa7..60a25cb 100644  
--- a/include/linux/cgroup.h  
+++ b/include/linux/cgroup.h  
@@ -293,6 +293,7 @@ int cgroup_is_descendant(const struct cgroup *cgrp);  
struct cgroup_subsys {  
    struct cgroup_subsys_state *(*create)(struct cgroup_subsys *ss,  
        struct cgroup *cgrp);  
+ void (*initialize)(int early);  
    void (*pre_destroy)(struct cgroup_subsys *ss, struct cgroup *cgrp);  
    void (*destroy)(struct cgroup_subsys *ss, struct cgroup *cgrp);  
    int (*can_attach)(struct cgroup_subsys *ss,  
diff --git a/kernel/cgroup.c b/kernel/cgroup.c  
index 8877a08..be37395 100644  
--- a/kernel/cgroup.c  
+++ b/kernel/cgroup.c  
@@ -2553,6 +2553,9 @@ int __init cgroup_init_early(void)  
  
    if (ss->early_init)  
        cgroup_init_subsys(ss);
```

```

+
+ if (ss->initialize)
+ ss->initialize(1);
+ }
+ return 0;
+ }
@@ -2577,6 +2580,9 @@ int __init cgroup_init(void)
+ struct cgroup_subsys *ss = subsys[i];
+ if (!ss->early_init)
+ cgroup_init_subsys(ss);
+
+
+ if (ss->initialize)
+ ss->initialize(0);
+ }

/* Add init_css_set to the hash table */
diff --git a/kernel/sched.c b/kernel/sched.c
index bbdc32a..50f737e 100644
--- a/kernel/sched.c
+++ b/kernel/sched.c
@@ -247,11 +247,32 @@ static void destroy_rt_bandwidth(struct rt_bandwidth *rt_b)
+ struct cfs_rq;

+
+ static LIST_HEAD(task_groups);
+ #ifdef CONFIG_CGROUP_SCHED
+ #define CPU_CGROUP_STAT_THRESHOLD (1 << 30)
+ 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 {
+ struct percpu_counter cpustat[CPU_CGROUP_STAT_NSTATS];
+ };
+
+ static void __cpu_cgroup_stat_add(struct cpu_cgroup_stat *stat,
+ enum cpu_cgroup_stat_index idx, int val)
+ {
+ if (stat)
+ percpu_counter_add(&stat->cpustat[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
@@ -3837,6 +3858,16 @@ 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(tg->stat, CPU_CGROUP_STAT_UTIME,
+ cputime_to_msecs(cputime));
+ }
+endif
+ }

/*
@@ -3892,8 +3923,17 @@ void account_system_time(struct task_struct *p, int hardirq_offset,
    cpustat->irq = cputime64_add(cpustat->irq, tmp);
    else if (softirq_count())
    cpustat->softirq = cputime64_add(cpustat->softirq, tmp);
- else if (p != rq->idle)
+ else if (p != rq->idle) {
    cpustat->system = cputime64_add(cpustat->system, tmp);
+ifdef CONFIG_CGROUP_SCHED
+ {
+ struct task_group *tg;
+ tg = task_group(p);
+ __cpu_cgroup_stat_add(tg->stat, CPU_CGROUP_STAT_STIME,
+ cputime_to_msecs(cputime));
+ }
+ }
+endif
    else if (atomic_read(&rq->nr_iowait) > 0)
    cpustat->iowait = cputime64_add(cpustat->iowait, tmp);
    else
@@ -8179,10 +8219,26 @@ static inline struct task_group *cgroup_tg(struct cgroup *cgrp)
    struct task_group, css);
}

+static void cpu_cgroup_initialize(int early)
+{
+ int i;

```

```

+ struct cpu_cgroup_stat *stat;
+
+ if (!early) {
+   stat = kmalloc(sizeof(struct cpu_cgroup_stat)
+     , GFP_KERNEL);
+   for (i = 0; i < CPU_CGROUP_STAT_NSTATS; i++)
+     percpu_counter_init(
+       &stat->cpustat[i], 0);
+   init_task_group.stat = stat;
+ }
+}
+
+ static struct cgroup_subsys_state *
+ cpu_cgroup_create(struct cgroup_subsys *ss, struct cgroup *cgrp)
+ {
+   struct task_group *tg;
+   int i;

+   if (!cgrp->parent) {
+     /* This is early initialization for the top cgroup */
@@ -8198,6 +8254,10 @@ cpu_cgroup_create(struct cgroup_subsys *ss, struct cgroup *cgrp)
+     if (IS_ERR(tg))
+       return ERR_PTR(-ENOMEM);

+   tg->stat = kmalloc(sizeof(struct cpu_cgroup_stat), GFP_KERNEL);
+   for (i = 0; i < CPU_CGROUP_STAT_NSTATS; i++)
+     percpu_counter_init(&tg->stat->cpustat[i], 0);
+
+   /* Bind the cgroup to task_group object we just created */
+   tg->css.cgroup = cgrp;

@@ -8251,6 +8311,38 @@ static u64 cpu_shares_read_u64(struct cgroup *cgrp, struct cftype
+ *cft)
+ }
+ #endif

+ static s64 cpu_cgroup_read_stat(struct cpu_cgroup_stat *stat,
+   enum cpu_cgroup_stat_index idx)
+ {
+   if (stat)
+     return percpu_counter_read(&stat->cpustat[idx]);
+
+   return 0;
+ }
+
+ 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 < CPU_CGROUP_STAT_NSTATS; 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;
+}
+
+#ifdef CONFIG_RT_GROUP_SCHED
static ssize_t cpu_rt_runtime_write(struct cgroup *cgrp, struct cftype *cft,
s64 val)
@@ -8295,6 +8387,10 @@ static struct cftype cpu_files[] = {
.write_u64 = cpu_rt_period_write_uint,
},
#endif
+ {
+ .name = "stat",
+ .read_map = cpu_cgroup_stats_show,
+ },
+ };

static int cpu_cgroup_populate(struct cgroup_subsys *ss, struct cgroup *cont)
@@ -8304,6 +8400,7 @@ static int cpu_cgroup_populate(struct cgroup_subsys *ss, struct
cgroup *cont)

struct cgroup_subsys cpu_cgroup_subsys = {
.name = "cpu",
.initialize = cpu_cgroup_initialize,
.create = cpu_cgroup_create,
.destroy = cpu_cgroup_destroy,
.can_attach = cpu_cgroup_can_attach,

```

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