Subject: Re: [PATCH v2 4/5] expose fine-grained per-cpu data for cpuacct stats
Posted by Sha Zhengju on Wed, 18 Apr 2012 12:30:39 GMT
View Forum Message <> Reply to Message

_

On Mon, Apr 9, 2012 at 6:25 PM, Glauber Costa <glommer@parallels.com> wrote:
> The cpuacct cgroup already exposes user and system numbers in a per-cgroup
> fashion. But they are a summation along the whole group, not a per-cpu figure.
> Also, they are coarse-grained version of the stats usually shown at places
> like /proc/stat.
>
> I want to have enough cgroup data to emulate the /proc/stat interface. To
> achieve that, I am creating a new file "stat_percpu" that displays the
> fine-grained per-cpu data. The original data is left alone.
>
> The format of this file resembles the one found in the usual cgroup's stat
> files. But of course, the fields will be repeated, one per cpu, and prefixed
> with the cpu number.
>
> Therefore, we'll have something like:
>
> cpu0.user X
> cpu0.system Y
> ...
> cpu1.user X1
> cpu1.system Y1
> ...
>

Why not show the all-cpu data together with the per-cpu one? I think
the total one
is an usual concern in most cases.

> Signed-off-by: Glauber Costa <glommer@parallels.com>
>
> kernel/sched/core.c | 34 ++++++++++++++++++++++++++++++++++
> 1 files changed, 34 insertions(+), 0 deletions(-)
>
> diff --git a/kernel/sched/core.c b/kernel/sched/core.c
> index 1ee3772..52bae67 100644
> --- a/kernel/sched/core.c
> +++ b/kernel/sched/core.c
> @@ -8186,6 +8186,35 @@ static int cpuacct_stats_show(struct cgroup *cgrp, struct cftype *cft,
>   return 0;
> }
>
> +static inline void do_fill_cb(struct cgroup_map_cb *cb, struct cpuacct *ca,

```
> +                        char *str, int cpu, int index)
> +{
> +       char name[24];
> +       struct kernel_cpustat *kcpustat = per_cpu_ptr(ca->cpustat, cpu);
> +
> +       snprintf(name, sizeof(name), "cpu%d.%s", cpu, str);
> +       cb->fill(cb, name, cputime64_to_clock_t(kcpustat->cpustat[index]));
> +}
> +
> +static int cpuacct_stats_percpu_show(struct cgroup *cgrp, struct cftype *cft,
> +                                    struct cgroup_map_cb *cb)
> +{
> +       struct cpuacct *ca = cgroup_ca(cgrp);
> +       int cpu;
> +
> +       for_each_online_cpu(cpu) {
> +               do_fill_cb(cb, ca, "user", cpu, CPUTIME_USER);
> +               do_fill_cb(cb, ca, "nice", cpu, CPUTIME_NICE);
> +               do_fill_cb(cb, ca, "system", cpu, CPUTIME_SYSTEM);
> +               do_fill_cb(cb, ca, "irq", cpu, CPUTIME_IRQ);
> +               do_fill_cb(cb, ca, "softirq", cpu, CPUTIME_SOFTIRQ);
> +               do_fill_cb(cb, ca, "guest", cpu, CPUTIME_GUEST);
> +               do_fill_cb(cb, ca, "guest_nice", cpu, CPUTIME_GUEST_NICE);
> +       }
> +
> +       return 0;
> +}
> +
>  static struct cftype files[] = {
>       {
>               .name = "usage",
> @@ -8200,6 +8229,11 @@ static struct cftype files[] = {
>               .name = "stat",
>               .read_map = cpuacct_stats_show,
>       },
> +     {
> +              .name = "stat_percpu",
> +              .read_map = cpuacct_stats_percpu_show,
> +       },
> +
>  };
>
>  static int cpuacct_populate(struct cgroup_subsys *ss, struct cgroup *cgrp)
> --
> 1.7.7.6
>
```