Subject: Re: [RFC][-mm] [1/2] Simple stats for cpu resource controller
Posted by Peter Zijlstra on Fri, 28 Mar 2008 10:17:16 GMT
View Forum Message <> Reply to Message

On Fri, 2008-03-28 at 15:32 +0530, Balaji Rao wrote:
> On Thursday 27 March 2008 01:28:10 am Peter Zijlstra wrote:
> > On Wed, 2008-03-26 at 23:48 +0530, Balaji Rao wrote:
> <snip>
> > > +/* Called under irq disable. */
> > > +static void __cpu_cgroup_stat_add_safe(struct cpu_cgroup_stat *stat,
> > > + enum cpu_cgroup_stat_index idx, int val)
> >
> > What is safe about this function?
> >
> That it can be called only from an interrupt context.

It can be called from any context that has hardirqs disabled. And the __
prefix suggests as much, no need to tag _safe to the end as well, we
never do that.

> > > +{
> > > + 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
> > > }
> >
> > So both of these are tick based? The normal CFS [us]time stats are not.
> >
> Hmmm.. Yea, right. So I should use the approach used by task_utime and task_stime when
reporting it, right ?

Not sure what you want to use this for, but yeah, that makes most sense.

That is, I do know _what_ you want to use it for, just not sure which
requirements you put on it. The pure tick based thing might be good
enough for most purposes, the runtime proportion thing is just more
accurate.

> > > /*
> > > @@ -7939,6 +7986,40 @@ static u64 cpu_shares_read_u64(struct cgroup *cgrp, struct
cftype *cft)
> > >
> > >   return (u64) tg->shares;
> > > }
> > > +
> >
> Thanks for the review.