Subject: Re: [PATCH 1/1] Fix a panic while mouting containers on powerpc and some other small cleanups (Re: [
Posted by Paul Menage on Mon, 15 Jan 2007 09:22:16 GMT
View Forum Message <> Reply to Message

On 1/15/07, Balbir Singh <balbir@in.ibm.com> wrote:
>
> In sched.c, account_user_time() can be called with the task p set to rq->idle.
> Since idle tasks do not belong to any container, this was causing a panic in
> task_ca() in cpu_acct.c.

How come that didn't cause a problem on x86_64? If this is an inconsistency between architectures then perhaps it ought to be cleaned up.

Additionally, I think that we should make the idle tasks members of the root container(s), to remove this special case. (I'm a bit surprised that they're not already - I thought that the early container initialization was early enough that the idle tasks hadn't yet been forked. Is that different on PowerPC?

>
> Multiplying the time by 1000 is not correct in cpuusage_read(). The code
> has been converted to use the correct cputime API.

Thanks.

>
> Add mount/umount callbacks.

I'm not sure I like the mount/unmount callbacks. What exactly are you trying to gain from them? My intention was that the

cont->subsys[i]->container = cont;

line in container_get_sb() was doing essentially this - i.e. the container_subsys_state for the root container in a subsystem is already kept up to date by the container system, and the subsystem can rely on the "container" field in the container_subsys_state.

Thanks,

Paul

>
> Signed-off-by: Balbir Singh <balbir@in.ibm.com>
> ---
>

```
>  kernel/cpu_acct.c |   29 ++++++++++++++++++++++++------
>  kernel/sched.c    |   17 ++++++++++------
>  2 files changed, 34 insertions(+), 12 deletions(-)
>
> diff -puN kernel/cpu_acct.c~fix-cpuacct-panic-on-mount kernel/cpu_acct.c
> ---  linux-2.6.20-rc3/kernel/cpu_acct.c~fix-cpuacct-panic-on-moun t       2007-01-15
> 14:23:20.000000000 +0530
> +++ linux-2.6.20-rc3-balbir/kernel/cpu_acct.c   2007-01-15 14:23:20.000000000 +0530
> @@ -22,6 +22,7 @@ struct cpuacct {
>  };
>
>  static struct container_subsys cpuacct_subsys;
> +static struct container *root;
>
>  static inline struct cpuacct *container_ca(struct container *cont)
>  {
> @@ -49,6 +50,16 @@ static void cpuacct_destroy(struct conta
>         kfree(container_ca(cont));
>  }
>
> +static void cpuacct_mount(struct container_subsys *ss, struct container *cont)
> +{
> +       root = cont;
> +}
> +
> +static void cpuacct_umount(struct container_subsys *ss, struct container *cont)
> +{
> +       root = NULL;
> +}
> +
>  static ssize_t cpuusage_read(struct container *cont,
>                              struct cftype *cft,
>                              struct file *file,
> @@ -57,6 +68,7 @@ static ssize_t cpuusage_read(struct cont
>  {
>         struct cpuacct *ca = container_ca(cont);
>         cputime64_t time;
> +       unsigned long time_in_jiffies;
>         char usagebuf[64];
>         char *s = usagebuf;
>
> @@ -64,9 +76,8 @@ static ssize_t cpuusage_read(struct cont
>         time = ca->time;
>         spin_unlock_irq(&ca->lock);
>
> -       time *= 1000;
> -       do_div(time, HZ);
> -       s += sprintf(s, "%llu", (unsigned long long) time);
```

```
> +       time_in_jiffies = cputime_to_jiffies(time);
> +       s += sprintf(s, "%llu\n", (unsigned long long) time_in_jiffies);
>
>         return simple_read_from_buffer(buf, nbytes, ppos, usagebuf, s - usagebuf);
>  }
> @@ -83,12 +94,13 @@ static int cpuacct_populate(struct conta
>  }
>
>
> -void cpuacct_charge(struct task_struct *task, cputime_t cputime) {
> +void cpuacct_charge(struct task_struct *task, cputime_t cputime)
> +{
>
>         struct cpuacct *ca;
>         unsigned long flags;
>
> -       if (cpuacct_subsys.subsys_id < 0) return;
> +       if (cpuacct_subsys.subsys_id < 0 || !root) return;
>         rcu_read_lock();
>         ca = task_ca(task);
>         if (ca) {
> @@ -104,13 +116,18 @@ static struct container_subsys cpuacct_s
>         .create = cpuacct_create,
>         .destroy = cpuacct_destroy,
>         .populate = cpuacct_populate,
> +       .mount = cpuacct_mount,
> +       .umount = cpuacct_umount,
>         .subsys_id = -1,
>  };
>
>
>  int __init init_cpuacct(void)
>  {
> -       int id = container_register_subsys(&cpuacct_subsys);
> +       int id;
> +
> +       root = NULL;
> +       id = container_register_subsys(&cpuacct_subsys);
>         return id < 0 ? id : 0;
>  }
>
> diff -puN kernel/sched.c~fix-cpuacct-panic-on-mount kernel/sched.c
> --- linux-2.6.20-rc3/kernel/sched.c~fix-cpuacct-panic-on-mount  2007-01-15
> 14:23:20.000000000 +0530
> +++ linux-2.6.20-rc3-balbir/kernel/sched.c      2007-01-15 14:23:20.000000000 +0530
> @@ -3067,10 +3067,17 @@ void account_user_time(struct task_struc
>  {
>         struct cpu_usage_stat *cpustat = &kstat_this_cpu.cpustat;
```

```
>         cputime64_t tmp;
> +       struct rq *rq = this_rq();
>
>         p->utime = cputime_add(p->utime, cputime);
>
> -       cpuacct_charge(p, cputime);
> +       /*
> +        * On powerpc this routine can be called with p set to the idle
> +        * task of the cpu. idle tasks don't really belong to any
> +        * container.
> +        */
> +       if (p != rq->idle)
> +               cpuacct_charge(p, cputime);
>
>         /* Add user time to cpustat. */
>         tmp = cputime_to_cputime64(cputime);
> @@ -3095,18 +3102,16 @@ void account_system_time(struct task_str
>
>         p->stime = cputime_add(p->stime, cputime);
>
> -       if (p != rq->idle)
> -               cpuacct_charge(p, cputime);
> -
>         /* Add system time to cpustat. */
>         tmp = cputime_to_cputime64(cputime);
>         if (hardirq_count() - 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);
> -       else if (atomic_read(&rq->nr_iowait) > 0)
> +               cpuacct_charge(p, cputime);
> +       } else if (atomic_read(&rq->nr_iowait) > 0)
>                 cpustat->iowait = cputime64_add(cpustat->iowait, tmp);
>         else
>                 cpustat->idle = cputime64_add(cpustat->idle, tmp);
> _
>
>         Balbir Singh
>         Linux Technology Center
>         Bangalore, IBM ISTL
>
```