Subject: Re: A question about group CFS scheduling
Posted by Peter Zijlstra on Thu, 26 Jun 2008 11:08:27 GMT
View Forum Message <> Reply to Message

On Thu, 2008-06-26 at 16:33 +0800, Zhao Forrest wrote:
> >
> > Let me explain the cgroup case (the sanest option IMHO):
> >
> > initially all your tasks will belong to the root cgroup, eg:
> >
> > assuming:
> > mkdir -p /cgroup/cpu
> > mount none /cgroup/cpu -t cgroup -o cpu
> >
> > Then the root cgroup (cgroup:/) is /cgroup/cpu/ and all tasks will be
> > found in /cgroup/cpu/tasks.
> >
> > You can then create new groups as sibling from this root group, eg:
> >
> > cgroup:/foo
> > cgroup:/bar
> >
> > They will get a weigth of 1024 by default, exactly as heavy as a nice 0
> > task.
> >
> > That means that no matter how many tasks you stuff into foo, their
> > combined cpu time will be as much as a single tasks in cgroup:/ would
> > get.
> >
> > This is fully recursive, so you can also create:
> >
> > cgroup:/foo/bar and its tasks in turn will get as much combined cpu time
> > as a single task in cgroup:/foo would get.
> >
> > In theory this should go on indefinitely, in practise we'll run into
> > serious numerical issues quite quickly.
> >
> >
> > The USER grouping basically creates a fake root and all uids (including
> > 0) are its siblings. The only special case is that uid-0 (aka root) will
> > get twice the weight of the others.
> >
>
> Thank you for your detailed explanation! I have one more question:
> cgrouping and USER grouping is mutual exclusive, am I right? That is,
> when enabling cgrouping, USER grouping need to be disabled, vice
> versa.

Yes indeed. That is set at kernel build time. So the kernel either supports cgroup scheduling or uid stuff.