
Subject: Re: A question about group CFS scheduling
Posted by [Peter Zijlstra](#) on Thu, 26 Jun 2008 08:15:12 GMT
[View Forum Message](#) <> [Reply to Message](#)

On Thu, 2008-06-26 at 15:19 +0800, Zhao Forrest wrote:

> Hi experts,
>
> In Documentation/sched-design-CFS.txt it reads:
> Group scheduler tunables:
>
> When CONFIG_FAIR_USER_SCHED is defined, a directory is created in sysfs for
> each new user and a "cpu_share" file is added in that directory.
> # cd /sys/kernel/uids
> # cat 512/cpu_share # Display user 512's CPU share
> 1024
> # echo 2048 > 512/cpu_share # Modify user 512's CPU share
> # cat 512/cpu_share # Display user 512's CPU share
> 2048
> #
> CPU bandwidth between two users are divided in the ratio of their CPU shares.
> For ex: if you would like user "root" to get twice the bandwidth of user
> "guest", then set the cpu_share for both the users such that "root"'s
> cpu_share is twice "guest"'s cpu_share.
>
> My question is: how is CPU bandwidth divided between cgroup and
> regular processes?
> For example,
> 1 the cpu_share of user "root" is set to 2048
> 2 the cpu_share of user "guest" is set to 1024
> 3 there're many processes owned by other users, which don't belong to any cgroup

A process always belongs to a (c)group.

> if the relative CPU bandwidth allocated to cgroup of "root" is 2,
> allocated to cgroup of "guest" is 1, then what's the relative CPU
> bandwidth allocated to other regular processes? 2 or 1?

Are you interested in UID based group scheduling or cgroup scheduling?

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 weight 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 task 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 practice 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.

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

Subject: Re: A question about group CFS scheduling
Posted by [Zhao Forrest](#) on Thu, 26 Jun 2008 08:33:11 GMT
[View Forum Message](#) <> [Reply to Message](#)

>
> 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 weight 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 task 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 practice 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 mutually exclusive, am I right? That is,
when enabling cgrouping, USER grouping needs to be disabled, vice
versa.

Thanks,
Forrest

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