

---

Subject: Re: [PATCH, v3 2/2] cgroups: introduce timer slack subsystem  
Posted by [Kirill A. Shutemov](#) on Fri, 04 Feb 2011 13:34:39 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

On Thu, Feb 03, 2011 at 11:57:43AM -0800, Jacob Pan wrote:

> On Thu, 3 Feb 2011 10:12:51 -0800  
> Paul Menage <menage@google.com> wrote:  
>  
> > On Thu, Feb 3, 2011 at 9:51 AM, Jacob Pan  
> > <jacob.jun.pan@linux.intel.com> wrote:  
> > >  
> > > I think this logic defeats the purpose of having timer\_slack  
> > > subsystem in the first place. IMHO, the original intention was to  
> > > have grouping effect of tasks in the cgroup.  
> >  
> > You can get the semantics you want by just setting min\_slack\_ns =  
> > max\_slack\_ns.  
> >  
> true. it will just make set fail when min = max. it is awkward and  
> counter intuitive when you want to change the group timer\_slack. you  
> will have to move both min and max to clamp the value, where set  
> function can not be used.

Interface is very similar to /sys/devices/system/cpu/cpuX/cpufreq.  
I think it's sane. If you want some extension, you can do it with  
userspace helper.

> In addition, when a parent changes min = max, I don't see the current  
> code enforce new settings on the children. Am i missing something?

I've missed it. I'll fix.

> In my use case, i want to put some apps into a managed group where  
> relaxed slack value is used, but when time comes to move the app out of  
> that cgroup, we would like to restore the original timer slack. I see  
> having a current value per cgroup can be useful if we let timer code  
> pick whether to use task slack value or the cgroup slack value.  
> Or we have to cache the old value per task

What's mean "original timer slack" if you are free to move a task  
between a lot of cgroups and process itself free to change it anytime?

--

Kirill A. Shutemov

---

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

---

Subject: Re: [PATCH, v3 2/2] cgroups: introduce timer slack subsystem  
Posted by [jacob.jun.pan](#) on Fri, 04 Feb 2011 17:27:55 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

On Fri, 4 Feb 2011 15:34:39 +0200

"Kirill A. Shutemov" <[kirill@shutemov.name](mailto:kirill@shutemov.name)> wrote:

> On Thu, Feb 03, 2011 at 11:57:43AM -0800, Jacob Pan wrote:

> > On Thu, 3 Feb 2011 10:12:51 -0800

> > Paul Menage <[menage@google.com](mailto:menage@google.com)> wrote:

> >

> > > On Thu, Feb 3, 2011 at 9:51 AM, Jacob Pan

> > > <[jacob.jun.pan@linux.intel.com](mailto:jacob.jun.pan@linux.intel.com)> wrote:

> > > >

> > > > I think this logic defeats the purpose of having timer\_slack

> > > > subsystem in the first place. IMHO, the original intention was

> > > > to have grouping effect of tasks in the cgroup.

> > >

> > > You can get the semantics you want by just setting min\_slack\_ns =

> > > max\_slack\_ns.

> > >

> > true. it will just make set fail when min = max. it is awkward and

> > counter intuitive when you want to change the group timer\_slack. you

> > will have to move both min and max to clamp the value, where set

> > function can not be used.

>

> Interface is very similar to /sys/devices/system/cpu/cpuX/cpufreq.

> I think it's sane. If you want some extention, you can do it with

> userspace helper.

>

I don't disagree the current interface is usable. Just less intuitive.

The situation is different for cpufreq, where you

don't the situation of adding new entries to be adjusted in the  
existing max-min range.

> > In addition, when a parent changes min = max, I don't see the

> > current code enforce new settings on the children. Am i missing

> > something?

>

> I've missed it. I'll fix.

>

> > In my use case, i want to put some apps into a managed group where

> > relaxed slack value is used, but when time comes to move the app

> > out of that cgroup, we would like to restore the original timer

> > slack. I see having a current value per cgroup can be useful if we

> > let timer code pick whether to use task slack value or the cgroup

> > slack value. Or we have to cache the old value per task

>

> What's mean "original timer slack" if you are free to move a task

> between a lot of cgroups and process itself free to change it anytime?  
>

I need to manage tasks by a management software instead of letting the task change timer\_slack by itself. The goal is to make management transparent and no modifications to the existing apps. Therefore, it is desirable to automatically enforce timer\_slack when the apps are in the cgroup while automatically restore it when it is no longer under cgroup management.

So the "original timer slack" can be the default 50us or whatever value chosen by the task itself. But the app itself should not care or even be aware of which cgroup it is in.

So here are two options I can think of

1. add a new variable called cg\_timer\_slack\_ns to struct task\_struct{}  
cg\_timer\_slack\_ns will be set by cgroup timer\_slack subsystem, then we can retain the original per task value in timer\_slack\_ns.  
timer code will pick max(cg\_timer\_slack\_ns, timer\_slack\_ns) if cg\_timer\_slack\_ns is set.

2. leave task\_struct unchanged, add a current\_timer\_slack to the cgroup. timer\_slack cgroup does not modify per task timer\_slack\_ns. similar to option #1, let timer code pick the timer\_slack to use based on whether the task is in timer\_slack cgroup.

Any thoughts?

Thanks,  
Jacob

---

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

---