Subject: Re: [RFC][PATCH 5/5] Add a Signal Control Group Subsystem
Posted by Matt Helsley on Wed, 30 Apr 2008 18:44:57 GMT
View Forum Message <> Reply to Message

```
On Fri, 2008-04-25 at 13:41 +0200, Cedric Le Goater wrote:
> Matt Helsley wrote:
> > Add a signal control group subsystem that allows us to send signals to all tasks
> > in the control group by writing the desired signal(7) number to the kill file.
> >
> > NOTE: We don't really need per-cgroup state, but control groups doesn't support
> > stateless subsystems yet.
> >
> > Signed-off-by: Matt Helsley <matthltc@us.ibm.com>
> > ---
> >  include/linux/cgroup_signal.h |   28 +++++++++
> >  include/linux/cgroup_subsys.h |    6 +
> >  init/Kconfig                  |    6 +
> >  kernel/Makefile               |    1
> >  kernel/cgroup_signal.c        |  129 ++++++++++++++++++++++++++++++++++++++++++
> >  5 files changed, 170 insertions(+)
> >
> > Index: linux-2.6.25-mm1/include/linux/cgroup_signal.h
> > ===================================================================
> > --- /dev/null
> > +++ linux-2.6.25-mm1/include/linux/cgroup_signal.h
> > @@ -0,0 +1,28 @@
> > +#ifndef _LINUX_CGROUP_SIGNAL_H
> > +#define _LINUX_CGROUP_SIGNAL_H
> > +/*
> > + * cgroup_signal.h -  control group freezer subsystem interface
>
> s/freezer/signal/
>
> > + *
> > + * Copyright IBM Corp. 2007
> > + *
> > + * Author : Cedric Le Goater <clg@fr.ibm.com>
> > + * Author : Matt Helsley <matthltc@us.ibm.com>
> > + */
> > +
> > +#include <linux/cgroup.h>
> > +
> > +#ifdef CONFIG_CGROUP_SIGNAL
> > +
> > +struct stateless {
> > + struct cgroup_subsys_state css;
> > +};
>
```

> I'm not sure this is correct to say so. Imagine you want to send
> a SIGKILL to a cgroup, you would expect all tasks to die and the
> cgroup to become empty. right ?
>
> but if a task is doing clone() while it's being killed by this cgroup
> signal subsystem, we can miss the child. This is because there's a
> small window in copy_process() where the child is in the cgroup and
> not visible yet.
>
>   copy_process()
> cgroup_fork()
>   do stuff
> cgroup_fork_callbacks()
>
> cgroup_post_fork()
>   put new task in the list.
>
> (I didn't dig too much the code, though. So I might be missing
>   something)
>
> So if we want to send the signal to all tasks in the cgroup, we need
> to track the new tasks with a fork callback, just like the freezer :
>
> static void signal_fork(struct cgroup_subsys *ss, struct task_struct *task)
> {
>
> }
>
> and, of course, we need to keep somewhere the signal number we need to
> send.
>
>
> All this depends on how we want the cgroup signal subsystem to behave.
> It could be brainless of course, but it seems to me that the biggest
> benefit of such a subsystem is to use the cgroup capability to track
> new tasks coming in.
>
> Cheers,
>
> C.

Assuming we did this, isn't it still possible to send SIGSTOP to every
task in the cgroup yet still appear to have not stopped every task in
the cgroup:

```
 Task A      Task B
echo 19 > signal.send
 record signal
 return -EBUSY from can_attach
 send signals to all the tasks
 return 0 from write syscall
      echo newpid > tasks
cat tasks
<Uh oh, not all tasks are stopped...>
```

Cheers,
 -Matt Helsley

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