Subject: Re: [RFC PATCH 0/4] Container Freezer: Reuse Suspend Freezer
Posted by Matt Helsley on Fri, 04 Apr 2008 22:27:40 GMT

View Forum Message <> Reply to Message

On Fri, 2008-04-04 at 11:56 -0400, Oren Laadan wrote:
>
> Matt Helsley wrote:
> > On Thu, 2008-04-03 at 16:49 -0700, Paul Menage wrote:
> >> On Thu, Apr 3, 2008 at 2:03 PM, <matthltc@us.ibm.com> wrote:
> >>>   * "freezer.kill"
> >>>
> >>>   writing <n> will send signal number <n> to all tasks
> >>>
> >> My first thought (not having looked at the code yet) is that sending a
> >> signal doesn't really have anything to do with freezing, so it
> >> shouldn't be in the same subsystem. Maybe a separate subsystem called
> >> "signal"?
> >>
> >> And more than that, it's not something that requires any particular
> >> per-process state, so there's no reason that the subsystem that
> >> provides the "kill" functionality shouldn't be able to be mounted in
> >> multiple hierarchies.
> >>
> >> How about if I added support for stateless subsystems, that could
> >> potentially be mounted in multiple hierarchies at once? They wouldn't
> >> need an entry in the css set, since they have no state.
> >
> > This seems reasonable to me. A quick look at Cedric's patches suggests
> > there's no need for such cgroup subsystems to be tied together -- the
> > signalling is all done internally to the freeze_task(), refrigerator(),
> > and thaw_process() functions from what I recall.
> >
> >>> * Usage :
> >>>
> >>>   # mkdir /containers/freezer
> >>>   # mount -t container -ofreezer freezer /containers/freezer
> >>>   # mkdir /containers/freezer/0
> >>>   # echo $some_pid > /containers/freezer/0/tasks
> >>>
> >>> to get status of the freezer subsystem :
> >>>
> >>>   # cat /containers/freezer/0/freezer.freeze
> >>>   RUNNING
> >>>
> >>> to freeze all tasks in the container :
> >>>
> >>>   # echo 1 > /containers/freezer/0/freezer.freeze
> >>>   # cat /containers/freezer/0/freezer.freeze

> >>>   FREEZING
> >>>   # cat /containers/freezer/0/freezer.freeze
> >>>   FROZEN
> >> Could we separate this out into two files? One called "freeze" that's
> >> a 0/1 for whether we're intending to freeze the subsystem, and one
> >> called "frozen" that indicates whether it is frozen? And maybe a
> >> "state" file to report the RUNNING/FREEZING/FROZEN distinction in a
> >> human-readable way?
> >
> > 3 files seems like overkill. I think making them human-readable is good
> > and can be done with two files: "state" (read-only) and
> > "state-next" (read/write). Transitions between RUNNING and FROZEN are
> > obvious when state-next != state. I think the advantages are it's pretty
> > human-readable, you don't need separate strings and files for the
> > transitions, it's clear what's about to happen (IMHO), and it only
> > requires 2 files. Some examples:
> >
> > To initiate freezing:
> >
> > # cat /containers/freezer/0/freezer.state
> > RUNNING
> > # echo "FROZEN" > /containers/freezer/0/freezer.state-next
> > # cat /containers/freezer/0/freezer.state
> > RUNNING
> > # cat /containers/freezer/0/freezer.state-next
> > FROZEN
> > # sleep N
> > # cat /containers/freezer/0/freezer.state
> > FROZEN
> > # cat /containers/freezer/0/freezer.state-next
> > FROZEN
> >
> > So to cancel freezing you might see something like:
> >
> > # cat /containers/freezer/0/freezer.state
> > RUNNING
> > # cat /containers/freezer/0/freezer.state-next
> > FROZEN
> > # echo "RUNNING" > /containers/freezer/0/freezer.state-next
> > # cat /containers/freezer/0/freezer.state-next
> > RUNNING
> >
> > If you wanted to know if a group was transitioning:
> >
> > # diff /containers/freezer/0/freezer.state /containers/freezer/0/freezer.state-next
> >
> > Or:
> > # current=`cat /containers/freezer/0/freezer.state`

```
> > # next=`cat /containers/freezer/0/freezer.state-next`
> > # [ "$current" != "$next" ] && echo "Transitioning"
> > # [ "$current" == "RUNNING" -a "$next" == "FROZEN" ] && echo "Freezing"
> > # [ "$current" == "FROZEN" -a "$next" == "RUNNING" ] && echo "Thawing"
> > # [ "$current" == "RUNNING" -a "$next" == "RUNNING" ] && echo "No-op"
> > # [ "$current" == "FROZEN" -a "$next" == "FROZEN" ] && echo "No-op"
>
```

> First, I totally agree with Serge's comment (oh well, it's about my
> own suggestion, so I must) - for checkpoint/restart we'll need more
> states if we are to use the same subsystem.

 I don't have an upper limit on how many more states we will need and I
think that number impacts the interface significantly. Can you give us
an estimate?

> Second, my gut feeling is that a single, atomic operation to get the
> status is preferred over multiple (non-atomic) operations. In other
> words, I suggest a single state file instead of two. You can encode
> every possible transition in a single state. It's not that the kernel

 If the transitions are to be human-readable and there are more than a
small number of states it may not be desirable to encode transitions as
states. Paul's reason for suggesting the additional file(s), as best I
could tell, was to keep the interface human-readable.

Cheers,
 -Matt Helsley

_____