Subject: Re: [PATCH 1/2] signal checkpoint: define /proc/pid/sig/
Posted by serue on Mon, 02 Jul 2007 21:40:40 GMT
View Forum Message <> Reply to Message

Quoting Daniel Lezcano (dlezcano@fr.ibm.com):
> Serge E. Hallyn wrote:
> >Quoting Carl-Daniel Hailfinger (c-d.hailfinger.devel.2006@gmx.net):
> >>On 11.06.2007 19:05, Serge E. Hallyn wrote:
> >>>Quoting Cedric Le Goater (clg@fr.ibm.com):
> >>>
> >>>>should we continue to use /proc ? or switch to some other mechanisms
> >>>>like getnetlink (taskstats) to map kernel structures.
> >>>We want to avoid 'map'ping kernel structures, though, right? We can
> >>>dump the data in a more generic fashion through netlink, dunno what we
> >>>prefer. But this is very definately process information :), so /proc
> >>>does seem appropriate.
> >>While I agree that /proc seems appropriate, I see a few benefits of
> >>dumping the data through netlink:
> >
> >Good points, thanks.
> >
> >>* Speed. IIRC there were benchmarks showing an advantage of netlink
> >> over /proc when communicating with userspace. Sorry, no idea where
> >> I read that.
> >
> >I don't think we're dumping large amounts of data (the largest amounts,
> >process memory, we're looking at doing just by forcing dump to swap), so
> >I'm not sure how much it matters.
> >
> >Still,
> >
> >>* Versioning. While we strive to have the perfect interface on the
> >> first try, changes might be necessary. I see no way to handle
> >> multiple versions of an interface in /proc without big headaches.
> >
> >Good point, this kind of offsets my major point against netlink, that
> >we'd likely inherently end up versioning the interface by being tempted
> >to dump kernel structures verbatim. I doubt anyone would claim that
> >we'll never need to update the /proc interface, so that may make using
> >/proc a nonstarter.
> >
> >>* Conformity. With /proc, people often see a file, take a look at
> >> it and try to infer the structure of the file from what they see.
> >> This has led to multiple problems in the past when the content of
> >> some files in /proc changed slightly and tools broke. With
> >> netlink, implementers have to look at the spec to achieve anything
> >> useful.
> >

> >Ok, so presumably we'd want some 'start a checkpoint' or 'start a
> >restore' command (through syscall or whatever) to create the netlink
> >socket and pass that to the various kernel dump/restore pieces?
> >
> >Is there some better alternative people prefer to a syscall? If not,
> >Daniel, would you mind adding that to the front of your patchset, and
> >having your udp socket checkpoint/restore use that socket?
>
> Sorry, resend with the right <from> email.
>
> I am not sure I understand what you want. Knowing I talk english like a
> french cow, perhaps I missed something. Just let me know ...

Mooooo.

> The udp socket c/r is:
> - you provide a fd, you get a raw data (for statefile).
> - you provide a raw data, you get a fd.
>
> The generic netlink are on top of the netlinks. You subscribe to a
> family (in this case AF_INET_CR), you send a message with the DUMP
> command and the fd parameter of the socket you want to checkpoint and
> you read the dump data. If you want to restore it, you send the message
> with the RESTORE command and the attributes you received from the
> previous dump message and you read the answer which contains the fd of
> the newly created socket.
>
> What you are proposing is to create a syscall for restore and
> checkpoint. If the generic netlink is used, this is useless, because you
> can just create a CHECKPOINT/RESTORE command message and use
> socket/write/read/close to get all the statefile.

I was just thinking of consolidating all the genetlink stuff so we don't
have N genetlink families, one for each checkpointable subsystem.

But it doesn't matter. After I finally started trying to implement the
signal c/r on top of netlink instead of /proc, it occurred to me there
is the same problem as we had with audit on netlink, except without
as simple a possibility of resolution (afaics) bc of the lack of pidns
ids.

The problem is that a netlink send and it's handler are completely
asynchronous to each other. So let's say the user sends in the pid of
the process whose signal state should be checkpointed. It presumably
does an rtnetlink_send to send the pid, but if you trace what happens
to that data, it gets placed on the skb queue and then pulled off by a
handler, which could be in a bogus pid namespace. Since we have no
way of specifying pid namespaces, this means we can't actually

disambiguate tasks with the same pid number.

One workaround would be to use the ids which are introduced by
Cedric's bind_ns patchset.

Or, we can stick with /proc interface for signal checkpoint and
restart.

Or, use a container subsystem to walk the list of tasks in the container
and checkpoint signal info for all the tasks. Could be combined into
the freezer subsystem, into a separate checkpoint_restart subsystem
combined with all other things to be checkpointed, or just made into
it's own separate subsystem just for signal info.

Any other ideas?

thanks,
-serge

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