Subject: Re: [patch -mm 1/5] mqueue namespace : add struct mq_namespace
Posted by ebiederm on Tue, 02 Oct 2007 17:16:46 GMT
View Forum Message <> Reply to Message

sukadev@us.ibm.com writes:

> Cedric Le Goater [clg@fr.ibm.com] wrote:
> |
> | >> however, we have an issue with the signal notification in __do_notify()
> | >> we could kill a process in a different pid namespace.
> | >
> | > So I took a quick look at the code as it is (before this patchset)
> | > and the taking a reference to a socket and the taking a reference to
> | > a struct pid should do the right thing when we intersect with other
> | > namespaces. It certainly does not look like a fundamental issue.
> >
> |
> | right. this should be covered when the pid namespace signal handling is
> | complete. kill_pid_info() should fail to send a signal to a sibling or
> | a parent pid namespace.
> |
> | I guess we should add a WARNING() to say that we're attempting to do so.
> >
> Just want to clarify how a signal is sent to a parent ns.
>
> A process P1 sets itself up to be notified when a message arrives
> on a queue.
>
> P1 then clones P2 with CLONE_NEWPID.
>
> P2 writes to the message queue and thus signals P1
>
> What should the semantics be here ?
>
> I guess it makes less sense for two namespaces to be dependent on the same
> message queue this way. But, if P2 writes to the queue, technically, the
> queue is not empty, so P1 should be notified, no ?

Sounds right to me.

> This sounds similar to the SIGIO signal case (F_SETOWN). My understanding
> was that we would notify whoever was set to receive the notification, even
> if they were in a parent ns (again my reasoning was its based on the state
> of a file).

Yep.

> IOW, should we change kill_pid_info() ? If the caller can 'see' the

> 'struct pid' they can signal it. The expectation was that callers would
> call find_vpid() and thus only see processes in their namespace.

Ok. Now I'm concerned.

I deliberately designed the initial pid namespace infrastructure to allow
mixing like this. Because it is the right thing to do.

The expectation is that in general namespaces provide isolation simply
because you cannot see and thus cannot interact with other processes.
However isolation is not the purpose in life of namespaces and if you
use them in more creative ways mixing should work just fine. But
you have to use all of the namespaces together, and you have
to carefully set things up to guarantee isolation.

The really challenging case to handle here is what happens if we are
signaling to someone in a sibling pid namespace. What do we set the
parent pid in the siginfo struct to. I think we agreed that 0 (blame
the kernel) is the appropriate pid last time we talked about this.

I'm worried now that the concept of vpid has confused someone. It
still doesn't feel right to me to call one pid value more or less
virtual then any other so the concept of a virtual pid doesn't make
sense to me. The way I have always thought of it is:
- pid_nr(struct pid *)
  The pid in the current pid namespace.
- __pid_nr(struct pid_namespace, struct pid *)
  The pid in some specified pid namespace.

With struct pid being defined to be global and doing something
appropriate in all pid namespaces.

Thinking about this concern that Cedric raises is actually independent
of the mqueue namespace and seems to be totally a pid namespace thing.
Because the only way this happens if we happen to share the mqueue
namespace. (i.e. what we are doing now).

Eric

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