Subject: Re: [patch -mm 1/5] mqueue namespace : add struct mq_namespace
Posted by ebiederm on Wed, 03 Oct 2007 14:32:19 GMT
View Forum Message <> Reply to Message

Cedric Le Goater <clg@fr.ibm.com> writes:

> [
> I have big fingers this morning and I managed to send this email
> while typing it ... see below for the end. I should be awake now :)
>]
>
>> 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.
>
> 0 seems appropriate for a signal coming from a parent namespace, yes. but
> here, we could be sending a signal from a child or sibling namespace.

Hmm. I finally see the part of this that is problematic and
that we aren't currently handling.

> sig_i.si_signo = info->notify.sigev_signo;
> sig_i.si_errno = 0;
> sig_i.si_code = SI_MESGQ;
> sig_i.si_value = info->notify.sigev_value;
> sig_i.si_pid = current->tgid;
> sig_i.si_uid = current->uid;
>
> kill_pid_info(info->notify.sigev_signo, &sig_i, info->notify_owner);

The filling in of the signal info structure. My gut feel says that
should be a struct pid reference.   We need to be very careful with
the siginfo cases, and si_pid. Unless someone has built a mechanism
for dealing with this I haven't seen.

I still think the right approach here is that if the pid doesn't map
into the target processes pid namespace we should use 0.

I also strongly suspect that si_pid should be a struct pid and that we
should translate it in the receiving process if possible.

> Is there a way to catch this general issue (we have the same in sigio)
> in the kill*(struct pid) routines ? spit a big warning when the
> current->nsproxy->pid_ns is not a parent ?

The SIGIO case is even trickier, although that may count as always
coming from the kernel so this doesn't come up.

In both cases what is really happening a process in a sibling
pid namespace is doing something and the kernel is telling us about
it. So sending the signal is legitimate.

The difficulty is that the kernel can't express the process that did
something. From my perspective that appears to be a fundamental
limitation, and our only real choice is to send 0.

We have a similar limitation with the uid namespace as well in this
case.

We can implement this easily by passing a struct pid. And translating
just before we cross the kernel/user boundary. And if it doesn't
translate use 0. Basically this is just your check for seeing
if the destination pid namespace is the same or a parent of
the sending pid namespace. That is what I always envisioned
being in __pid_nr(). As I don't think it makes any sense to map
pids from sibling pid namespaces.

The reason we want in general to do the translation at the
destination process is because we may be sending a signal
to a process group which could have processes in multiple
pid namespaces.

Eric

_____