Subject: Re: [RFC][PATCH] fork: Don't special case CLONE_NEWPID for process or sessions
Posted by Pavel Emelianov on Thu, 01 Nov 2007 15:37:51 GMT
View Forum Message <> Reply to Message

Eric W. Biederman wrote:
> Pavel Emelyanov <xemul@openvz.org> writes:
>
>> Eric W. Biederman wrote:
>>
>> Sorry for the late answer, I have just noticed that I forgot to
>> answer on this patch.
>
> Thanks for answering.
>
>>> Given that the kernel supports sys_setsid we don't need a special case
>>> in fork if we want to set: session == pgrp == pid.
>>>
>>> The historical (although not 2.6) linux behavior has been to start the
>>> init with session == pgrp == 0 which is effectively what removing this
>>> special case will do.
>> Hm... I overlooked this fact. Looks like the namespace's init will
>> have them set to 1.
>
> Yes. It is not a big difference as init can handle being exec'd by
> something else, thus is expected to be able to handle the case where
> setsid has already been called.
>
> So we are good but your current code makes it impossible to set
> tsk->signal->leader and become a proper session leader which is
> painful.
>
>>> can we remove it and save some code, make copy_process easier to read
>>> easier to maintain, and possibly a little faster?
>>>
>>> I know it is a little weird belong to a process groups that isn't
>>> visible in your pid namespace, but it there are no good reasons
>>> why it shouldn't work.
>> This is not good to have such a situation as the init will have
>> the ability to kill the tasks from the namespace he can't see,
>> e.g. his parent and the processes in that group.
>
> Yes. sys_kill(0, SIGXXX) will allow this.
>
> As this is the main reason for this I don't see any reason to keep
> the current clone behavior.

Are you talking about keeping the ability to kill the outer processes?

> Sending signals to our process group and our parent is an ability that
> we allow even the most untrusted processes normally, and it is an
> ability we can easily remove simply by calling setsid.

You mix two things together - letting tasks send signals to their
groups is good, but letting tasks send signals outside the namespace
is bad.

> Not doing magic with the session and the process group allows init
> to properly become a session leader when setsid is called.
>
> Starting with a shared session and process group makes it more likely
> kernel implementors will look closely to ensure they handle strange
> cases like this properly and that developers using CLONE_NEWPID will
> look closely to ensure there are not other pid gotchas the need to
> deal with.
>
> Sharing the process group, session and controlling tty of our parent
> can be an advantage in small scenarios where using an existing
> controlling tty is an advantage. Think of a chroot build root or a
> chroot rpm install. Not letting processes escape and become deaemons
> is an advantage, but it really doesn't matter if they send signals to
> their parent.

Well, we allow a tiny possibility to have shared pids, but do we
really want to support this possibility in the rest of the code?

> When isolation is important we do not want the ability to send signals
> to outside of the pid namespace. Currently except for the child death
> signal of init it appears that simply calling setsid is enough.
>
> So short of any other objections I think I will brush up this patch and
> send it along to Andrew.

Hm... Could you please send it for pre-rfc before then?

> Eric
>

_____