Subject: Re: [RFC][PATCH 2/7] VPIDs: pid/vpid conversions
Posted by ebiederm on Thu, 09 Feb 2006 00:37:31 GMT
View Forum Message <> Reply to Message

Alexey Kuznetsov <kuznet@ms2.inr.ac.ru> writes:

> Hello!
>
>> Do you know how incomplete this patch is?
>
> The question is for me. It handles all the subsystems which are allowed
> to be used inside openvz containers. And _nothing_ more, it would be pure S&M.

I agree and this is why I don't like VPIDS I don't see a way for them
to be anything but pure S&M.

>> Is there a plan to catch all of the in-kernel use of pids
>
> grep for ->pid,->tgid,->pgid,->session and look. What could be better? :-)

Ouch. I know there are cases that the above test fails for. Which
is why I prefer an interface that takes a global reference and gives
you a compile error if you don't. You are much more likely to catch
all of the users that way.

>> You missed cap_set_all.
>
> No doubts, something is missing. Please, could you show how to fix it
> or to point directly at the place. Thank you.

In capability.c it does for_each_thread or something like that. It is
very similar to cap_set_pg. But in a virtual context all != all :)

The current OpenVZ patch appears to at least catch cap_set_all.

> Actually, you cycled on this pid problem. If you think private pid spaces
> are really necessary, it is prefectly OK. openvz (and, maybe, all VPS-oriented
> solutions) do _not_ need this (well, look, virtuozzo is a mature product
> for 5 years already, and vpids were added very recently for one specific
> purpose), but can live within private spaces or just in peace with them.
> We can even apply vpids on top on pid spaces to preserve global process tree.
> Provided you leave a chance not to enforce use of private pid spaces
> inside containers, of course.

I think for people doing migration a private pid space in some form is
necessary, I agree it is generally overkill for the VPS case but if it
is efficient it should be usable. And certainly having facilities
like this be optional seems very important.

My problem with the vpid case and it's translate at the kernel boundary is that boundary is huge, and there is no compile time checking to help you find the problem users. So I don't think vpids make a solution that can be maintained, and thus merging them looks like a very bad idea.

Eric
