Subject: Re: Namespaces exhausted CLONE_XXX bits problem
Posted by serue on Mon, 14 Jan 2008 18:07:48 GMT
View Forum Message <> Reply to Message

Quoting Pavel Emelyanov (xemul@openvz.org):
> Serge E. Hallyn wrote:
> > Quoting Cedric Le Goater (clg@fr.ibm.com):
> >> to be more precise :
> >>
> >> long sys_clone_something(struct clone_something_args args)
> >>
> >> and
> >>
> >> long sys_unshare_something(struct unshare_something_args args)
> >>
> >> The arg passing will be slower bc of the copy_from_user() but we will
> >> still have the sys_clone syscall for the fast path.
> >>
> >> C.
> >
> > I'm fine with the direction you're going, but just as one more option,
> > we could follow more of the selinux/lsm approach of first requesting
> > clone/unshare options, then doing the actual clone/unshare. So
> > something like
> >
> > sys_clone_request(extended_64bit_clone_flags)
>
> What if we someday hit the 64-bit limit? :)
>
> > sys_clone(usual args)
> >
> > or
> >
> > echo pid,mqueue,user,ipc,uts,net > /proc/self/clone_unshare
> > clone()
>
> Well, this is how sys_indirect() was intended to work. Nobody
> liked it, so I'm afraid this will also not be accepted.

I would have thought sys_indirect would be disliked because
it looks like an ioctl type multiplexor. Whereas sys_clone_request()
or /proc/self/clone_unshare simply sets arguments in advance, the
way /proc/self/attr/current does.

-serge

_____
Containers mailing list
Containers@lists.linux-foundation.org