Subject: Re: Network namespaces a path to mergable code.
Posted by Andrey Savochkin on Wed, 28 Jun 2006 17:22:40 GMT
View Forum Message <> Reply to Message

Hi Eric,

On Wed, Jun 28, 2006 at 10:51:26AM -0600, Eric W. Biederman wrote:
> Andrey Savochkin <saw@swsoft.com> writes:
>
> > One possible option to resolve this question is to show 2 relatively short
> > patches just introducing namespaces for sockets in 2 ways: with explicit
> > function parameters and using implicit current context.
> > Then people can compare them and vote.
> > Do you think it's worth the effort?
>
> Given that we have two strong opinions in different directions I think it
> is worth the effort to resolve this.

Do you have time to extract necessary parts of your old patch?
Or you aren't afraid of letting me draft an alternative version of socket
namespaces basing on your code? :)

>
> In a slightly different vein your second patch introduced a lot
> of #ifdef CONFIG_NET_NS in C files. That is something we need to look closely
> at.
>
> So I think the abstraction that we use to access per network namespace
> variables needs some work if we are going to allow the ability to compile
> out all of the namespace code. The explicit versus implicit lookup is just
> one dimension of that problem.

This is a good comment.

Those ifdef's mostly correspond to places where we walk over lists
and need to filter-out entities not belonging to a specific namespace.
Those places about the same in your and my implementation.
We can think what we can do with them.
One trick that I used on several occasions is net_ns_same macro
which doesn't evalute its arguments if CONFIG_NET_NS not defined,
and thus can be used without ifdef's.

Returning to implicit vs explicit function arguments, I belive that implicit
arguments are more promising in having zero impact on the code when
CONFIG_NET_NS is disabled.
Functions like inet_addr_type will translate into exactly the same code as
they did without net namespace patches.

>
> >> I'm still curious why many of those chunks can't use existing helper
> >> functions, to be cleaned up.
> >
> > What helper functions are you referring to?
>
> Basically most of the device list walker functions live in.
> net/core/dev.c
>
> I don't know if the cases you fixed could have used any of those
> helper functions but it certainly has me asking that question.
>
> A general pattern that happens in cleanups is the discovery
> that code using an old interface in a problematic way really
> could be done much better another way. I didn't dig enough
> to see if that was the case in any of the code that you changed.

Well, there is obvious improvement of this kind: many protocols walk over
device list to find devices with non-NULL protocol specific pointers.
For example, IPv6, decnet and others do it on module unloading to clean up.
Those places just ask for some simpler standard way of doing it, but I wasn't
bold enough for such radical change.
Do you think I should try?

Best regards

Andrey
