Subject: Re: namespace support requires network modules to say "GPL"
Posted by ebiederm on Tue, 04 Dec 2007 19:17:57 GMT
View Forum Message <> Reply to Message

Ben Greear <greearb@candelatech.com> writes:

> Eric W. Biederman wrote:
>> However there also seem to be simpler cases like Ben's bridge module,
>> that don't appear to have any global state.
>>
> Well, my module has some global state, but I don't think it needs to care about
> namespaces. My first impression is that my module should be able to bridge
> namespaces...not be contained within one. I can have user-space make sure that
> I don't bridge between
> devices in different name-spaces, or perhaps bridging between namespaces
> wouldn't be a problem anyway.

Bridging between namespaces should not be a problem, but it could be
a bit of a challenge to setup (in finding the network devices).
Probably the easy way is to setup the bridging and then move one of the
network devices to the other network namespace.

Essentially bridging between two network devices in two network
namespaces looks like bridging between two network devices on two
separate network stacks. Although internally things look a little
better.

> If I *do* need to add some sort of namespace
> awareness to just achieve today's functionality, I don't mind making the
> changes,
> so long as I don't need to change to GPL licensing. Perhaps at the least you
> can export enough symbols w/out GPL tag to achieve backwards compat with .23
> and previous kernels, or rework dev_get_by_* etc to not need GPL'd namespace
> symbols and just return the device in the default namespace?

IANAL but to me your code sounds like a derivative work of the linux
kernel. Which implies that if you are distributing your module you
need to change to GPL licensing. The _GPL tag on EXPORT_SYMBOL does
not change those rules.

>> Ben I don't have a clue how your user space interface works. My gut
>> feel is that you can likely use sk->sk_net (if your configuration is
>> through a socket), or failing that current->nsproxy->net_ns. To get
>> the network namespace to look up "eth0" and "eth1".
>>
> Currently I use procfs and ioctls bound to a procfs file descriptor.

Which is where it gets tricky You are defining new userspace ABIs.

I can see where they occasionally make sense during development
and prototyping but long term out of tree userspace interfaces appear
to me to be a real maintenance problem.

> For namespaces in general, will there be a way to just do a dev_get_by_* and
> find the
> device in *any* namespace and query the device to see what namespace it is in?
> Then my module or some other more clever piece of code can determine the
> namespaces
> (by comparing pointers if nothing else) and make proper decision. For instance,
> maybe
> we want to bridge two namespaces, or maybe we want to forbid that ever
> happening...

The issue is that fundamentally all userspace device identifiers can
be duped between namespaces. So since there is no unique identifier
we can not implement a function to do that.

>> This however still begs the question how do we want to handle this
>> so there is a minimum of pain.
>>
>> Since using register_pernet_subsys implies you need your own member
>> in struct net. I am inclined to leave that with the GPL hint on
>> the EXPORT as you need to be really tight with the system to use that.
>>
> I certainly don't want to have to muck with struct net unless you have some way
> to
> register anonymous (and non GPL) subsystems. I'm guessing you do not want to
> support that....

Well I don't see a license compatible way to have any GPL incompatible
licensed linux kernel code. Off hand that means code needs to be
licensed under the GPL or BSD without advertising clause.

Does EXPORT_SYMBOL_GPL complain if you have a BSD licensed module?

Eric