Subject: Re: namespace support requires network modules to say "GPL"
Posted by Stephen Hemminger on Sun, 02 Dec 2007 04:23:54 GMT
View Forum Message <> Reply to Message

On Sat, 01 Dec 2007 22:34:09 -0500
Mark Lord <lkml@rtr.ca> wrote:

> Stephen Hemminger wrote:
> > On Sat, 1 Dec 2007 11:17:36 -0800
> > Stephen Hemminger <shemminger@linux-foundation.org> wrote:
> >
> >> Then init_net needs to be not GPL limited. Sorry, we need to allow
> >> non GPL network drivers. There is a fine line between keeping the
> >> binary seething masses from accessing random kernel functions, and allowing
> >> reasonable (but still non GPL) things like ndiswrapper to use network
> >> device interface.
> >>
> > I spoke too soon earlier, ndiswrapper builds and loads against current
> > 2.6.24-rc3. Vmware and proprietary VPN software probably do not. Once again I don't
> > give a damn, but the enterprise distro vendors certainly care.
> ...
>
> Naw, enterprise (or any other) distro vendors shouldn't have any issues here,
> since they can just patch their kernels around any issues.
>
> But it looks like Eric has this one thought out well enough.

So you are saying all this is not a problem, fine.
Any affected parties can certainly lobby for themselves. But I suspect
they all think the kernel community is a bunch of ... and will just ignore
the problem.

--
Stephen Hemminger <shemminger@linux-foundation.org>

____
Containers mailing list
Containers@lists.linux-foundation.org
https://lists.linux-foundation.org/mailman/listinfo/containers

Subject: Re: namespace support requires network modules to say "GPL"
Posted by Ben Greear on Sun, 02 Dec 2007 19:28:56 GMT
View Forum Message <> Reply to Message

Stephen Hemminger wrote:
>>
>> Naw, enterprise (or any other) distro vendors shouldn't have any issues here,
>> since they can just patch their kernels around any issues.

>>
>> But it looks like Eric has this one thought out well enough.
>>
>
> So you are saying all this is not a problem, fine.
> Any affected parties can certainly lobby for themselves. But I suspect
> they all think the kernel community is a bunch of ... and will just ignore
> the problem.
>
I have a binary module that uses dev_get_by_name...it's sort of a
bridge-like thing and
needs user-space to tell it which device to listen for packets on...

This code doesn't need or care about name-spaces, so I don't see how it
could really
be infringing on the author's code (any worse than loading a binary
driver into the kernel
ever does).

I would certainly prefer to not have to patch around any problems with
calling dev_get_by_name
from a non-gpl module, but if required, I can probably figure something
out...

Thanks,
Ben

--
Ben Greear <greearb@candelatech.com>
Candela Technologies Inc http://www.candelatech.com

____
Containers mailing list
Containers@lists.linux-foundation.org
https://lists.linux-foundation.org/mailman/listinfo/containers

---

Subject: Re: namespace support requires network modules to say "GPL"
Posted by Patrick McHardy on Sun, 02 Dec 2007 20:03:56 GMT
View Forum Message <> Reply to Message

Ben Greear wrote:
> Stephen Hemminger wrote:
>>>
>>> Naw, enterprise (or any other) distro vendors shouldn't have any
>>> issues here,
>>> since they can just patch their kernels around any issues.

>>>
>>> But it looks like Eric has this one thought out well enough.
>>>
>>
>> So you are saying all this is not a problem, fine.
>> Any affected parties can certainly lobby for themselves. But I suspect
>> they all think the kernel community is a bunch of ... and will just
>> ignore
>> the problem.
>
> I have a binary module that uses dev_get_by_name...it's sort of a
> bridge-like thing and
> needs user-space to tell it which device to listen for packets on...
>
> This code doesn't need or care about name-spaces, so I don't see how it
> could really
> be infringing on the author's code (any worse than loading a binary
> driver into the kernel
> ever does).
>
> I would certainly prefer to not have to patch around any problems with
> calling dev_get_by_name
> from a non-gpl module, but if required, I can probably figure something
> out...

For all I care binary modules can break, but frankly I don't see
how encapsulating a couple of structures and pointers in a new
structure and adding a new argument to existing functions shifts
the decision about how a function should be usable to the namespace
guys. IMO all functions should continue to be usable as before,
as decided by whoever actually wrote them. The only exception
might be stuff where an existing EXPORT_SYMBOL is clearly wrong,
but that would be a seperate discussion.

____
Containers mailing list
Containers@lists.linux-foundation.org
https://lists.linux-foundation.org/mailman/listinfo/containers

---

Subject: Re: namespace support requires network modules to say "GPL"
Posted by Adrian Bunk on Sun, 02 Dec 2007 20:43:56 GMT
View Forum Message <> Reply to Message

On Sun, Dec 02, 2007 at 09:03:56PM +0100, Patrick McHardy wrote:
> Ben Greear wrote:
>> Stephen Hemminger wrote:

>>>>
>>>> Naw, enterprise (or any other) distro vendors shouldn't have any issues
>>>> here,
>>>> since they can just patch their kernels around any issues.
>>>>
>>>> But it looks like Eric has this one thought out well enough.
>>>>
>>>
>>> So you are saying all this is not a problem, fine.
>>> Any affected parties can certainly lobby for themselves. But I suspect
>>> they all think the kernel community is a bunch of ... and will just
>>> ignore
>>> the problem.
> >
>> I have a binary module that uses dev_get_by_name...it's sort of a
>> bridge-like thing and
>> needs user-space to tell it which device to listen for packets on...
>>
>> This code doesn't need or care about name-spaces, so I don't see how it
>> could really
>> be infringing on the author's code (any worse than loading a binary driver
>> into the kernel
>> ever does).
>>
>> I would certainly prefer to not have to patch around any problems with
>> calling dev_get_by_name
>> from a non-gpl module, but if required, I can probably figure something
>> out...
>
>
> For all I care binary modules can break, but frankly I don't see
> how encapsulating a couple of structures and pointers in a new
> structure and adding a new argument to existing functions shifts
> the decision about how a function should be usable to the namespace
> guys. IMO all functions should continue to be usable as before,
> as decided by whoever actually wrote them.
>...

Even ignoring the fact that it's unclear whether distributing modules
with not GPLv2 compatible licences is legal at all or might bring you in
jail, your statement has an interesting implication:

Stuff like e.g. the EXPORT_SYMBOL(sk_alloc) predates the
EXPORT_SYMBOL_GPL stuff.

Who is considered the author of this code?

And when should he state whether he prefers to use EXPORT_SYMBOL_GPL

but wasn't able to use it at that when he wrote it since his code
predates it and is glad to be able to decide this now?

cu
Adrian

--

"Is there not promise of rain?" Ling Tan asked suddenly out
of the darkness. There had been need of rain for many days.
"Only a promise," Lao Er said.
Pearl S. Buck - Dragon Seed

____

Subject: Re: namespace support requires network modules to say "GPL"
Posted by Patrick McHardy on Sun, 02 Dec 2007 21:59:46 GMT
View Forum Message <> Reply to Message

Adrian Bunk wrote:
> On Sun, Dec 02, 2007 at 09:03:56PM +0100, Patrick McHardy wrote:
>
>> For all I care binary modules can break, but frankly I don't see
>> how encapsulating a couple of structures and pointers in a new
>> structure and adding a new argument to existing functions shifts
>> the decision about how a function should be usable to the namespace
>> guys. IMO all functions should continue to be usable as before,
>> as decided by whoever actually wrote them.
>> ...
>
> Even ignoring the fact that it's unclear whether distributing modules
> with not GPLv2 compatible licences is legal at all or might bring you in
> jail,

Agreed, lets ignore that :)

> your statement has an interesting implication:
>
> Stuff like e.g. the EXPORT_SYMBOL(sk_alloc) predates the
> EXPORT_SYMBOL_GPL stuff.
>
> Who is considered the author of this code?
>
> And when should he state whether he prefers to use EXPORT_SYMBOL_GPL

> but wasn't able to use it at that when he wrote it since his code
> predates it and is glad to be able to decide this now?

He can state it when he feels like it, I don't see the point.
Authors generally get to decide whether they use EXPORT_SYMBOL
or EXPORT_SYMBOL_GPL unless in cases where its really clear-cut
that EXPORT_SYMBOL is inapproriate. But thats a different matter.

If a symbol was OK to be used previously and something using it
would not automatically be considered a derived work, how does
passing &init_net to the function just to make the compiler
happy, avoid BUG_ONs and generally keep things working as before
make it more of a derived work?

____
Containers mailing list
Containers@lists.linux-foundation.org
https://lists.linux-foundation.org/mailman/listinfo/containers

---

Subject: Re: namespace support requires network modules to say "GPL"
Posted by Adrian Bunk on Mon, 03 Dec 2007 01:14:37 GMT
View Forum Message <> Reply to Message

On Sun, Dec 02, 2007 at 10:59:46PM +0100, Patrick McHardy wrote:
> Adrian Bunk wrote:
>> On Sun, Dec 02, 2007 at 09:03:56PM +0100, Patrick McHardy wrote:
>...
>> your statement has an interesting implication:
>>
>> Stuff like e.g. the EXPORT_SYMBOL(sk_alloc) predates the EXPORT_SYMBOL_GPL
>> stuff.
>>
>> Who is considered the author of this code?
>>
>> And when should he state whether he prefers to use EXPORT_SYMBOL_GPL but
>> wasn't able to use it at that when he wrote it since his code predates it
>> and is glad to be able to decide this now?
>
> He can state it when he feels like it, I don't see the point.
> Authors generally get to decide whether they use EXPORT_SYMBOL
> or EXPORT_SYMBOL_GPL unless in cases where its really clear-cut
> that EXPORT_SYMBOL is inapproriate. But thats a different matter.
>...

You miss my point.

Stuff like sk_alloc was exported to modules before EXPORT_SYMBOL_GPL existed (it was even exported to modules before EXPORT_SYMBOL existed).

We are talking about code and exports that are at about 12 years old,
which is at about twice as old as EXPORT_SYMBOL_GPL.

So what should happen in your opinion if e.g. Alan checks which of the
network code he had written when it was exported a dozen years ago,
stating that he never wanted it to be available to non-GPL modules?

cu
Adrian

--

"Is there not promise of rain?" Ling Tan asked suddenly out
of the darkness. There had been need of rain for many days.
"Only a promise," Lao Er said.
Pearl S. Buck - Dragon Seed

____
Containers mailing list
Containers@lists.linux-foundation.org
https://lists.linux-foundation.org/mailman/listinfo/containers

Subject: Re: namespace support requires network modules to say "GPL"
Posted by den on Mon, 03 Dec 2007 08:33:30 GMT
View Forum Message <> Reply to Message

Patrick McHardy wrote:
> Adrian Bunk wrote:
>> On Sun, Dec 02, 2007 at 09:03:56PM +0100, Patrick McHardy wrote:
>>
>>> For all I care binary modules can break, but frankly I don't see
>>> how encapsulating a couple of structures and pointers in a new
>>> structure and adding a new argument to existing functions shifts
>>> the decision about how a function should be usable to the namespace
>>> guys. IMO all functions should continue to be usable as before,
>>> as decided by whoever actually wrote them.
>>> ...
>>
>> Even ignoring the fact that it's unclear whether distributing modules
>> with not GPLv2 compatible licences is legal at all or might bring you
>> in jail,
>
> Agreed, lets ignore that :)
>
>> your statement has an interesting implication:

>>
>> Stuff like e.g. the EXPORT_SYMBOL(sk_alloc) predates the
>> EXPORT_SYMBOL_GPL stuff.
>>
>> Who is considered the author of this code?
>>
>> And when should he state whether he prefers to use EXPORT_SYMBOL_GPL
>> but wasn't able to use it at that when he wrote it since his code
>> predates it and is glad to be able to decide this now?
>
>
> He can state it when he feels like it, I don't see the point.
> Authors generally get to decide whether they use EXPORT_SYMBOL
> or EXPORT_SYMBOL_GPL unless in cases where its really clear-cut
> that EXPORT_SYMBOL is inapproriate. But thats a different matter.
>
> If a symbol was OK to be used previously and something using it
> would not automatically be considered a derived work, how does
> passing &init_net to the function just to make the compiler
> happy, avoid BUG_ONs and generally keep things working as before
> make it more of a derived work?

We, namely, Pavel Emelyanov and me, if we have some rights as a
committers to this staff :), do not mind against change
EXPORT_SYMBOL_GPL to EXPORT_SYMBOL.

Regards,
 Den

____
Containers mailing list
Containers@lists.linux-foundation.org
https://lists.linux-foundation.org/mailman/listinfo/containers

---

Subject: Re: namespace support requires network modules to say "GPL"
Posted by ebiederm on Mon, 03 Dec 2007 17:35:21 GMT
View Forum Message <> Reply to Message

Patrick McHardy <kaber@trash.net> writes:

> Ben Greear wrote:
>> I have a binary module that uses dev_get_by_name...it's sort of a bridge-like
>> thing and
>> needs user-space to tell it which device to listen for packets on...
>>
>> This code doesn't need or care about name-spaces, so I don't see how it could
>> really
>> be infringing on the author's code (any worse than loading a binary driver

>> into the kernel
>> ever does).

Regardless of infringement it is incompatible with a complete network
namespace implementation. Further it sounds like the module you are
describing defines a kernel ABI without being merged and hopes that
ABI will still be supportable in the future. Honestly I think doing so
is horrible code maintenance policy.

>> I would certainly prefer to not have to patch around any problems with calling
>> dev_get_by_name
>> from a non-gpl module, but if required, I can probably figure something out...
>
>
> For all I care binary modules can break, but frankly I don't see
> how encapsulating a couple of structures and pointers in a new
> structure and adding a new argument to existing functions shifts
> the decision about how a function should be usable to the namespace
> guys. IMO all functions should continue to be usable as before,
> as decided by whoever actually wrote them. The only exception
> might be stuff where an existing EXPORT_SYMBOL is clearly wrong,
> but that would be a seperate discussion.

I don't think we have actually shifted the decision.

Further from a namespace perspective if I had to support out of tree
modules and the current in kernel API the implementation would be
impossible short of loading kernel modules multiple times once
for each namespace. I totally refuse to give out of tree modules
that power whatever their license.

Right now the network namespace code that has been merged isn't that
interesting as it does not include ipv4 and ipv6 support which everyone
uses.

One of the tests for completion of the network namespace work is
grepping for &init_net and making certain we have cleanly removed
all references to except in a handful of cases like the boot code.

Once things are largely complete it makes sense to argue with out of
tree module authors that because they don't have network namespace
support in their modules, their modules are broken.

Right now I suspect to many developers even of in-tree modules
I have just shifted code around in an annoying looking way. I can
completely see other developers not getting the point.

Eric

Subject: Re: namespace support requires network modules to say "GPL"
Posted by Ben Greear on Mon, 03 Dec 2007 18:19:02 GMT
View Forum Message <> Reply to Message

Eric W. Biederman wrote:
> Patrick McHardy <kaber@trash.net> writes:
>
>
>> Ben Greear wrote:
>>
>>> I have a binary module that uses dev_get_by_name...it's sort of a bridge-like
>>> thing and
>>> needs user-space to tell it which device to listen for packets on...
>>>
>>> This code doesn't need or care about name-spaces, so I don't see how it could
>>> really
>>> be infringing on the author's code (any worse than loading a binary driver
>>> into the kernel
>>> ever does).
>>>
>
> Regardless of infringement it is incompatible with a complete network
> namespace implementation. Further it sounds like the module you are
> describing defines a kernel ABI without being merged and hopes that
> ABI will still be supportable in the future. Honestly I think doing so
> is horrible code maintenance policy.
>
I don't mind if the ABI changes, so long as I can still use something
similar.

The namespace logic is interesting to me in general, but at this point I
can't think of a way that
it actually helps this particular module. All I really need is a way to
grab every frame
from eth0 and then transmit it to eth1. I'm currently doing this by
finding the netdevice
and registering a raw-packet protocol (ie, like tcpdump would do). At
least up to 2.6.23,
this does not require any hacks to the kernel and uses only non GPL
exported symbols.

Based on my understanding of the namespace logic, if I never add any

namespaces,
the general network layout should look similar to how it does today, so
I should have
no logical problem with my module.

> Once things are largely complete it makes sense to argue with out of
> tree module authors that because they don't have network namespace
> support in their modules, their modules are broken.
>
Does this imply that every module that accesses the network code *must*
become
GPL simply because it must interact with namespace logic that is
exported as GPL only symbols?

Thanks,
Ben

--
Ben Greear <greearb@candelatech.com>
Candela Technologies Inc http://www.candelatech.com

____

Subject: Re: namespace support requires network modules to say "GPL"
Posted by Daniel Lezcano on Mon, 03 Dec 2007 18:57:53 GMT
View Forum Message <> Reply to Message

Ben Greear wrote:
> Eric W. Biederman wrote:
>> Patrick McHardy <kaber@trash.net> writes:
>>
>>
>>> Ben Greear wrote:
>>>
>>>> I have a binary module that uses dev_get_by_name...it's sort of a
>>>> bridge-like
>>>> thing and
>>>> needs user-space to tell it which device to listen for packets on...
>>>>
>>>> This code doesn't need or care about name-spaces, so I don't see how
>>>> it could
>>>> really
>>>> be infringing on the author's code (any worse than loading a binary

>>>> driver
>>>> into the kernel
>>>> ever does).
>>>>
>>
>> Regardless of infringement it is incompatible with a complete network
>> namespace implementation. Further it sounds like the module you are
>> describing defines a kernel ABI without being merged and hopes that
>> ABI will still be supportable in the future. Honestly I think doing so
>> is horrible code maintenance policy.
>>
> I don't mind if the ABI changes, so long as I can still use something
> similar.
>
> The namespace logic is interesting to me in general, but at this point I
> can't think of a way that
> it actually helps this particular module. All I really need is a way to
> grab every frame
> from eth0 and then transmit it to eth1. I'm currently doing this by
> finding the netdevice
> and registering a raw-packet protocol (ie, like tcpdump would do). At
> least up to 2.6.23,
> this does not require any hacks to the kernel and uses only non GPL
> exported symbols.
>
> Based on my understanding of the namespace logic, if I never add any
> namespaces,
> the general network layout should look similar to how it does today, so
> I should have
> no logical problem with my module.
>
>> Once things are largely complete it makes sense to argue with out of
>> tree module authors that because they don't have network namespace
>> support in their modules, their modules are broken.
> Does this imply that every module that accesses the network code *must*
> become
> GPL simply because it must interact with namespace logic that is
> exported as GPL only symbols?

That's right, with init_net's EXPORT_SYMBOL_GPL and dev_get_xx, we
enforce people to be GPL whatever they didn't asked to have the
namespaces in their code.

Eric, why can we simply change EXPORT_SYMBOL_GPL to EXPORT_SYMBOL for
init_net ?

Subject: Re: namespace support requires network modules to say "GPL"
Posted by Daniel Lezcano on Tue, 04 Dec 2007 15:19:38 GMT
View Forum Message <> Reply to Message

Daniel Lezcano wrote:
> Ben Greear wrote:
>> Eric W. Biederman wrote:
>>> Patrick McHardy <kaber@trash.net> writes:
>>>
>>>
>>>> Ben Greear wrote:
>>>>
>>>>> I have a binary module that uses dev_get_by_name...it's sort of a
>>>>> bridge-like
>>>>> thing and
>>>>> needs user-space to tell it which device to listen for packets on...
>>>>>
>>>>> This code doesn't need or care about name-spaces, so I don't see
>>>>> how it could
>>>>> really
>>>>> be infringing on the author's code (any worse than loading a binary
>>>>> driver
>>>>> into the kernel
>>>>> ever does).
>>>>>
>>>
>>> Regardless of infringement it is incompatible with a complete network
>>> namespace implementation. Further it sounds like the module you are
>>> describing defines a kernel ABI without being merged and hopes that
>>> ABI will still be supportable in the future. Honestly I think doing so
>>> is horrible code maintenance policy.
>>>
>> I don't mind if the ABI changes, so long as I can still use something
>> similar.
>>
>> The namespace logic is interesting to me in general, but at this point
>> I can't think of a way that
>> it actually helps this particular module. All I really need is a way
>> to grab every frame
>> from eth0 and then transmit it to eth1. I'm currently doing this by
>> finding the netdevice
>> and registering a raw-packet protocol (ie, like tcpdump would do). At
>> least up to 2.6.23,

>> this does not require any hacks to the kernel and uses only non GPL
>> exported symbols.
>>
>> Based on my understanding of the namespace logic, if I never add any
>> namespaces,
>> the general network layout should look similar to how it does today,
>> so I should have
>> no logical problem with my module.
>>
>>> Once things are largely complete it makes sense to argue with out of
>>> tree module authors that because they don't have network namespace
>>> support in their modules, their modules are broken.
>> Does this imply that every module that accesses the network code
>> *must* become
>> GPL simply because it must interact with namespace logic that is
>> exported as GPL only symbols?
>
> That's right, with init_net's EXPORT_SYMBOL_GPL and dev_get_xx, we
> enforce people to be GPL whatever they didn't asked to have the
> namespaces in their code.
>
> Eric, why can we simply change EXPORT_SYMBOL_GPL to EXPORT_SYMBOL for
> init_net ?

Another suggestion/question, is it acceptable to say non-gpl driver
should use init_task.nsproxy->net_ns instead of &init_net ?

Or does it make sense to have init_net gpl-exported, since we can access
it through init_task which is exported without gpl mention ?

____

Subject: Re: namespace support requires network modules to say "GPL"
Posted by ebiederm on Tue, 04 Dec 2007 17:59:05 GMT
View Forum Message <> Reply to Message

Ben Greear <greearb@candelatech.com> writes:

>> Regardless of infringement it is incompatible with a complete network
>> namespace implementation. Further it sounds like the module you are
>> describing defines a kernel ABI without being merged and hopes that
>> ABI will still be supportable in the future. Honestly I think doing so
>> is horrible code maintenance policy.
>>

> I don't mind if the ABI changes, so long as I can still use something similar.

It has occurred to me that I am seeing an implication here that may in fact not
exist.

My impression of dev_get_by_xxxx is that the function is only able to be used
sanely when being part of the definition of a kernel/userspace interface. With
the further assumption on my part that you need to define a new instance of
dev_get_by_xxxx

It has just occurred to me that it is possible to reuse the SIOCBRADDIF
and SIOCBRDELIF for that same purpose without defining a new kernel/userspace
interface.

What and how are you using dev_get_by_xxx?

Eric

____
Containers mailing list
Containers@lists.linux-foundation.org
https://lists.linux-foundation.org/mailman/listinfo/containers

---

Subject: Re: namespace support requires network modules to say "GPL"
Posted by ebiederm on Tue, 04 Dec 2007 18:03:01 GMT

View Forum Message <> Reply to Message

Daniel Lezcano <daniel.lezcano@free.fr> writes:

> Ben Greear wrote:

>>> Once things are largely complete it makes sense to argue with out of
>>> tree module authors that because they don't have network namespace
>>> support in their modules, their modules are broken.
>> Does this imply that every module that accesses the network code *must* become
>> GPL simply because it must interact with namespace logic that is exported as
>> GPL only symbols?
>
> That's right, with init_net's EXPORT_SYMBOL_GPL and dev_get_xx, we enforce
> people to be GPL whatever they didn't asked to have the namespaces in their
> code.
>
> Eric, why can we simply change EXPORT_SYMBOL_GPL to EXPORT_SYMBOL for init_net ?

Hmm. I need to think this one through.

EXPORT_SYMBOL_GPL acts as a strong hint, and a hindrance to using
symbols in a non-GPL'd module. Not exactly an enforcement mechanism.

...

The current pattern is to first change the code to only work in the
initial network namespace. Which can usually be done with a few
trivial lines of code that utilize init_net.

Then the pattern is to move the globals (or at least a pointer to
them) into struct net, and utilize register_pernet_subsys to ensure
those variables are properly initialized and cleaned up after.

However there also seem to be simpler cases like Ben's bridge module,
that don't appear to have any global state.

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".

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.

...

Currently I don't know if the _GPL hint on the export of init_net buys
us anything except trouble so I am almost inclined to do something
there.

....

What really disturbs me is that as I look at this I see that we have
historically at least done a very haphazard job of maintaining our
kernel/userspace ABIs while making a commitment to maintain them
forever. Especially if as it seems that some would see that
commitment extending beyond the code that is ever potentially
mergable with the kernel.

....

Currently the only angle that I can see that makes sense to me in the
argument for change of how we are currently doing things is that by
adding a parameter to new existing functions I make it very difficult
for code with network namespace support to have one version that works
on both old and new kernels as we can not define the new API on the

old hardware.

I can see some technical merit in making that case better.

.....

My thinking on the namespaces have been that their interfaces are new
core kernel interfaces that have not existed on any other kernel. And
as such any code that needed to use those interfaces was:
a) definitely a derived work of the kernel.
b) was a core part of the kernel, and we don't even want normal
   day to day drivers using those interfaces much less weird
   random code outside of the kernel.

The above is why I habitually place a _GPL hint on my exports
of namespace related functions and data. To strongly suggest
to module authors that they are getting into hot water if
they use these interfaces and don't merge their code.

So far I really don't see anything to challenge my understanding above
but I am human and as such my heuristics for analysis and understanding
are not guaranteed to give me the right answer.

....

I don't want this to be a stupid political fight about GPL stuff.
Generally I am with Alan in not seeing any basis for distributing
non-GPL code that works in the kernel. Although I see Linus' point
that a legal case may be made that certain modules are not a
derivative work of the kernel.

...

I am confused. I don't see a path forward that feels right.
So I am going to sit and think about this some more, before I do anything.

Eric

Subject: Re: namespace support requires network modules to say "GPL"
Posted by Ben Greear on Tue, 04 Dec 2007 18:44:53 GMT

View Forum Message <> Reply to Message

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. 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?
> 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.

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...
> 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....

Thanks,

Ben

--
Ben Greear <greearb@candelatech.com>
Candela Technologies Inc http://www.candelatech.com

Subject: Re: namespace support requires network modules to say "GPL"
Posted by Ben Greear on Tue, 04 Dec 2007 18:57:01 GMT
View Forum Message <> Reply to Message

Eric W. Biederman wrote:
> Ben Greear <greearb@candelatech.com> writes:
>
>
>>> Regardless of infringement it is incompatible with a complete network
>>> namespace implementation. Further it sounds like the module you are
>>> describing defines a kernel ABI without being merged and hopes that
>>> ABI will still be supportable in the future. Honestly I think doing so
>>> is horrible code maintenance policy.
>>>
>>>
>> I don't mind if the ABI changes, so long as I can still use something similar.
>>
>
> It has occurred to me that I am seeing an implication here that may in fact not
> exist.
>
> My impression of dev_get_by_xxxx is that the function is only able to be used
> sanely when being part of the definition of a kernel/userspace interface. With
> the further assumption on my part that you need to define a new instance of
> dev_get_by_xxxx
>
> It has just occurred to me that it is possible to reuse the SIOCBRADDIF
> and SIOCBRDELIF for that same purpose without defining a new kernel/userspace
> interface.
>
> What and how are you using dev_get_by_xxx?
>
I have a module that has a collection of 2-port bridges. These bridges
are used for emulation
purposes (somewhat similar to netem's feature set). Each bridge is

logically independent
of the others.   To set up a bridge, I do something like:

echo add_my_bridge my_br1 eth0 eth1 > /proc/net/foo/config

Inside the module, it reads "eth0" and "eth1" and needs to find those
devices (ie, dev_get_by_name). It then registers to receive all pkts from
eth1 and transmit them on eth0, and vice versa.

If it would not require GPL symbols, I have no problem changing my API
to be something
like:

echo add_my_bridge my_br1 eth0 namespaceX eth1 namespaceY >
/proc/net/foo/config

I am using procfs so that I don't have to define any new 'official'
kernel ABI, as that would more likely be a derivative work, and is a pain
to keep up to date with changing kernels anyway...

Personally, it seems useful for my module to be able to have eth0 in one
namespace
and eth1 in another, but I won't complain if they both have to be in the
same namespace
or even just in the default namespace due to GPL symbol issues.

Thanks,
Ben

--
Ben Greear <greearb@candelatech.com>
Candela Technologies Inc http://www.candelatech.com

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

____
Containers mailing list
Containers@lists.linux-foundation.org
https://lists.linux-foundation.org/mailman/listinfo/containers

---

Subject: Re: namespace support requires network modules to say "GPL"
Posted by Ben Greear on Tue, 04 Dec 2007 19:35:47 GMT

View Forum Message <> Reply to Message

Eric W. Biederman wrote:
> 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.

Ok, that sounds fine.

>> 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.

They are completely contained within my module, and no one is going
to change my module w/out me knowing, so actually I have very little
problem here :)

>> 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.

Ok, but can a netdev at least know what namespace it is in? I don't
need this for my module, but it seems very useful knowledge...

Thanks,
Ben

--
Ben Greear <greearb@candelatech.com>
Candela Technologies Inc http://www.candelatech.com

Subject: Re: namespace support requires network modules to say "GPL"
Posted by ebiederm on Tue, 04 Dec 2007 20:01:09 GMT
View Forum Message <> Reply to Message

Ben Greear <greearb@candelatech.com> writes:

> I have a module that has a collection of 2-port bridges. These bridges are used
> for emulation
> purposes (somewhat similar to netem's feature set). Each bridge is logically
> independent
> of the others.   To set up a bridge, I do something like:
>
> echo add_my_bridge my_br1 eth0 eth1 > /proc/net/foo/config

Interesting. Currently /proc/net is also per network namespace.
So normally I would say just call get_proc_net from inside your
proc handler and all would be well.

At another location in /proc (not under /proc/net) I would just do:
dev_get_by_name(current->nsproxy->net_ns, "ethX");

I would probably be paranoid and grab current->nsproxy->net_ns
when the file was opened and put it when the file was closed
just to ensure that if someone opened it and then passed
the file descriptor to someone else there were not any
weird little races. But I don't expect that is a problem
in your case.

> Personally, it seems useful for my module to be able to have eth0 in one
> namespace
> and eth1 in another, but I won't complain if they both have to be in the same
> namespace
> or even just in the default namespace due to GPL symbol issues.

It probably is easiest to move the devices after your module has
bridged them.

Eric

Subject: Re: namespace support requires network modules to say "GPL"
Posted by ebiederm on Tue, 04 Dec 2007 20:09:24 GMT
View Forum Message <> Reply to Message

Ben Greear <greearb@candelatech.com> writes:

> Ok, but can a netdev at least know what namespace it is in? I don't
> need this for my module, but it seems very useful knowledge...

Sure.   dev->nd_net
It is a don't care not a don't know, and there should be device
events when it goes in and out of a network namespace.

I don't know if the device gets those or not.

Eric

Subject: Re: namespace support requires network modules to say "GPL"
Posted by davem on Wed, 05 Dec 2007 06:01:50 GMT
View Forum Message <> Reply to Message

From: ebiederm@xmission.com (Eric W. Biederman)
Date: Tue, 04 Dec 2007 11:03:01 -0700

> I am confused. I don't see a path forward that feels right.

Eric, instead of writing a book about how you feel, look
at the simple facts and resolve this quickly.

You added a new key, the namespace, to the looking up of
network objects.

Big deal.

That does not imply a change in licensing for the interfaces

where you added that new aspect of the key to the argument
list.

This symbol licensing decision was not your's to make, so you must
revert the new licensing change you are enforcing upon everyone.

I want a de-GPL patch in my mailbox from you within the next 24 hours
or I will code one up myself. This is getting beyond rediculious.

My patience is completely gone on this matter, resolve this now.

Thanks.

Subject: Re: namespace support requires network modules to say "GPL"
Posted by davem on Wed, 05 Dec 2007 06:07:44 GMT
View Forum Message <> Reply to Message

From: Ben Greear <greearb@candelatech.com>
Date: Tue, 04 Dec 2007 10:57:01 -0800

> echo add_my_bridge my_br1 eth0 namespaceX eth1 namespaceY >
> /proc/net/foo/config

Each process executes in a namespace, so specifying the namespace
is redundant, just fetch the current process's namespace to pass
into the dev_get_by_*() routines.

Anyone interested in using a different namespace's devices
can change the namespace the process is executing in before
the procfs echo.

Subject: Re: namespace support requires network modules to say "GPL"
Posted by davem on Wed, 05 Dec 2007 06:14:47 GMT
View Forum Message <> Reply to Message

From: ebiederm@xmission.com (Eric W. Biederman)
Date: Tue, 04 Dec 2007 12:17:57 -0700

> Ben Greear <greearb@candelatech.com> writes:
>
> > 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.

Eric, YANAL and you are also full of hot air. You are really
testing my patience on this issue.

You fail to ever describe on what factual basis you are making
these claims. And the reason is that you have ZERO factual basis
for your claims.

Here are the facts:

1) Never, ever, have the function for looking up network devices been
   classified as GPL-only symbols.

   They provide a device based upon a lookup key.

2) You in no way have changed what those functions do in any way
   whatsoever. They still provide a reference to a network device
   based upon a given lookup key.

   The functions are still doing the same thing they always have.

Therefore, you have decided to uniliaterally change the licensing of
these functions based solely upon your opinion, and not because of
some real change you've made to the code in question.

You have no right to do this.

This is unreasonable, and you must fix this immediately.

And I do mean now, not after you've written several more excessively
long diatribes about how you feel in this matter.

Thank you.