Subject: Re: [patch 2/6] [Network namespace] Network device sharing by view
Posted by Andrey Savochkin on Tue, 27 Jun 2006 09:38:49 GMT
View Forum Message <> Reply to Message

On Tue, Jun 27, 2006 at 11:34:36AM +0200, Daniel Lezcano wrote:
> Andrey Savochkin wrote:
> > Daniel,
> >
> > On Mon, Jun 26, 2006 at 05:49:41PM +0200, Daniel Lezcano wrote:
> >
> >>>Then you lose the ability for each namespace to have its own routing entries.
> >>>Which implies that you'll have difficulties with devices that should exist
> >>>and be visible in one namespace only (like tunnels), as they require IP
> >>>addresses and route.
> >>
> >>I mean instead of having the route tables private to the namespace, the
> >>routes have the information to which namespace they are associated.
> >
> >
> > I think I understand what you're talking about: you want to make routing
> > responsible for determining destination namespace ID in addition to route
> > type (local, unicast etc), nexthop information, and so on. Right?
>
> Yes.
>
> >
> > My point is that if you make namespace tagging at routing time, and
> > your packets are being routed only once, you lose the ability
> > to have separate routing tables in each namespace.
>
> Right. What is the advantage of having separate the routing tables ?

Routing is everything.
For example, I want namespaces to have their private tunnel devices.
It means that namespaces should be allowed have private routes of local type,
private default routes, and so on...

Andrey

Subject: Re: [patch 2/6] [Network namespace] Network device sharing by view
Posted by Daniel Lezcano on Tue, 27 Jun 2006 11:21:02 GMT
View Forum Message <> Reply to Message

>>>My point is that if you make namespace tagging at routing time, and
>>>your packets are being routed only once, you lose the ability
>>>to have separate routing tables in each namespace.
>>

>>Right. What is the advantage of having separate the routing tables ?
>
>
> Routing is everything.
> For example, I want namespaces to have their private tunnel devices.
> It means that namespaces should be allowed have private routes of local type,
> private default routes, and so on...
>

Ok, we are talking about the same things. We do it only in a different way:

```
 * separate routing table :
   namespace
   |
   \--- route_tables
    |
    \---routes

 * tagged routing table :
  route_tables
   |
   \---routes
    |
    \---namespace
```

When using routes private to the namespace, globally the logic of the ip stack is not changed, it manipulates only differents variables. It is more clean than tagging the route for the reasons mentioned by Eric.

When using route tagging, the logic is changed because when doing lookup on the routes table which is global, the namespace is used to match the route and make it visible.

I use the second method, because I think it is more effecient and reduce the overhead. But the isolation is minimalist and only aims to avoid the application using ressources outside of the container (aka namespace) without taking care of the system. For example, I didn't take care of network devices, because as far as see I can't imagine an administrator wanting to change the network device name while there are hundred of containers running. Concerning tunnel devices for example, they should be created inside the container.

I think, private network ressources method is more elegant and involves more network ressources, but there is probably a significant overhead and some difficulties to have __lightweight__ container (aka application container), make nfs working well, etc... I did some tests with tbench and the loopback with the private namespace and there is roughly an overhead of 4 % without the isolation since with the tagging method there is 1 % with the isolation.

The network namespace aims the isolation for now, but the container
based on the namespaces will probably need checkpoint/restart and
migration ability. The migration is needed not only for servers but for
HPC jobs too.

So I don't know what level of isolation/virtualization is really needed
by users, what should be acceptable (strong isolation and overhead /
weak isolation and efficiency). I don't know if people wanting strong
isolation will not prefer Xen (cleary with much more overhead than your
patches ;))

Regards
 -- Daniel

Subject: Re: [patch 2/6] [Network namespace] Network device sharing by view
Posted by ebiederm on Tue, 27 Jun 2006 11:52:52 GMT
View Forum Message <> Reply to Message

Daniel Lezcano <dlezcano@fr.ibm.com> writes:

```
>>>>My point is that if you make namespace tagging at routing time, and
>>>>your packets are being routed only once, you lose the ability
>>>>to have separate routing tables in each namespace.
>>>
>>>Right. What is the advantage of having separate the routing tables ?
>> Routing is everything.
>> For example, I want namespaces to have their private tunnel devices.
>> It means that namespaces should be allowed have private routes of local type,
>> private default routes, and so on...
>>
>
> Ok, we are talking about the same things. We do it only in a different way:
>
>  * separate routing table :
>    namespace
>    |
>    \--- route_tables
>     |
>     \---routes
>
>  * tagged routing table :
>   route_tables
>    |
>   \---routes
>     |
>     \---namespace
```

There is a third possibility, that falls in between these two if local
communication is really the bottle neck.

We have the dst cache for caching routes and cache multiple transformations
that happen on a packet.

With a little extra knowledge it is possible to have the separate
routing tables but have special logic that recognizes the local tunnel
device that connects namespaces and have it look into the next
namespaces routes, and build up a complete stack of dst entries of
where the packet needs to go.

I keep forgetting about that possibility. But as long as everything
is done at the routing layer that should work.

> I use the second method, because I think it is more effecient and reduce the
> overhead. But the isolation is minimalist and only aims to avoid the application
> using ressources outside of the container (aka namespace) without taking care of
> the system. For example, I didn't take care of network devices, because as far
> as see I can't imagine an administrator wanting to change the network device
> name while there are hundred of containers running. Concerning tunnel devices
> for example, they should be created inside the container.

Inside the containers I want all network devices named eth0!

> I think, private network ressources method is more elegant and involves more
> network ressources, but there is probably a significant overhead and some
> difficulties to have __lightweight__ container (aka application container), make
> nfs working well, etc... I did some tests with tbench and the loopback with the
> private namespace and there is roughly an overhead of 4 % without the isolation
> since with the tagging method there is 1 % with the isolation.

The overhead went down?

> The network namespace aims the isolation for now, but the container based on the
> namespaces will probably need checkpoint/restart and migration ability. The
> migration is needed not only for servers but for HPC jobs too.

Yes.

> So I don't know what level of isolation/virtualization is really needed by
> users, what should be acceptable (strong isolation and overhead / weak isolation
> and efficiency). I don't know if people wanting strong isolation will not prefer
> Xen (cleary with much more overhead than your patches ;))

We need a clean abstraction that optimizes well.

However local communication between containers is not what we should benchmark. That can always be improved later. So long as the performance is reasonable. What needs to be benchmarked is the overhead of namespaces when connected to physical networking devices and on their own local loopback, and comparing that to a kernel without namespace support.

If we don't hurt that core case we have an implementation we can merge. There are a lot of optimization opportunities for local communications and we can do that after we have a correct and accepted implementation. Anything else is optimizing too soon, and will just be muddying the waters.

Eric

Subject: Re: [patch 2/6] [Network namespace] Network device sharing by view
Posted by Andrey Savochkin on Tue, 27 Jun 2006 11:55:25 GMT

View Forum Message <> Reply to Message

Daniel,

On Tue, Jun 27, 2006 at 01:21:02PM +0200, Daniel Lezcano wrote:
> >>>My point is that if you make namespace tagging at routing time, and
> >>>your packets are being routed only once, you lose the ability
> >>>to have separate routing tables in each namespace.
> >>
> >>Right. What is the advantage of having separate the routing tables ?
> >
> >
> > Routing is everything.
> > For example, I want namespaces to have their private tunnel devices.
> > It means that namespaces should be allowed have private routes of local type,
> > private default routes, and so on...
> >
>
> Ok, we are talking about the same things. We do it only in a different way:

We are not talking about the same things.

It isn't a technical thing whether route lookup is performed before or after namespace change.
It is a fundamental question determining functionality of network namespaces.
We are talking about the capabilities namespaces provide.

Your proposal essentially denies namespaces to have their own tunnel or other
devices. There is no point in having a device inside a namespace if the
namespace owner can't route all or some specific outgoing packets through
that device. You don't allow system administrators to completely delegate
management of network configuration to namespace owners.

 Andrey

Subject: Re: [patch 2/6] [Network namespace] Network device sharing by view
Posted by Herbert Poetzl on Tue, 27 Jun 2006 16:02:42 GMT
View Forum Message <> Reply to Message

```
On Tue, Jun 27, 2006 at 05:52:52AM -0600, Eric W. Biederman wrote:
> Daniel Lezcano <dlezcano@fr.ibm.com> writes:
>
> >>>>My point is that if you make namespace tagging at routing time,
> >>>>and your packets are being routed only once, you lose the ability
> >>>>to have separate routing tables in each namespace.
> >>>
> >>>Right. What is the advantage of having separate the routing tables ?
> >> Routing is everything. For example, I want namespaces to have their
> >> private tunnel devices. It means that namespaces should be allowed
> >> have private routes of local type, private default routes, and so
> >> on...
> >>
> >
> > Ok, we are talking about the same things. We do it only in a different way:
> >
> >  * separate routing table :
> >    namespace
> >    |
> >    \--- route_tables
> >     |
> >     \---routes
> >
> >  * tagged routing table :
> >   route_tables
> >    |
> >    \---routes
> >     |
> >     \---namespace
>
> There is a third possibility, that falls in between these two if local
> communication is really the bottle neck.
>
> We have the dst cache for caching routes and cache multiple
> transformations that happen on a packet.
```

>
> With a little extra knowledge it is possible to have the separate
> routing tables but have special logic that recognizes the local
> tunnel device that connects namespaces and have it look into the next
> namespaces routes, and build up a complete stack of dst entries of
> where the packet needs to go.
>
> I keep forgetting about that possibility. But as long as everything is
> done at the routing layer that should work.
>
> > I use the second method, because I think it is more effecient and
> > reduce the overhead. But the isolation is minimalist and only aims
> > to avoid the application using ressources outside of the container
> > (aka namespace) without taking care of the system. For example, I
> > didn't take care of network devices, because as far as see I can't
> > imagine an administrator wanting to change the network device name
> > while there are hundred of containers running. Concerning tunnel
> > devices for example, they should be created inside the container.
>
> Inside the containers I want all network devices named eth0!

huh? even if there are two of them? also tun?

I think you meant, you want to be able to have eth0 in
more than one guest where eth0 in a guest can also
be/use/relate to eth1 on the host, right?

> > I think, private network ressources method is more elegant
> > and involves more network ressources, but there is probably a
> > significant overhead and some difficulties to have __lightweight__
> > container (aka application container), make nfs working well,
> > etc... I did some tests with tbench and the loopback with the
> > private namespace and there is roughly an overhead of 4 % without
> > the isolation since with the tagging method there is 1 % with the
> > isolation.
>
> The overhead went down?

yes, this might actually happen, because the guest
has only to look at a certain subset of entries
but this needs a lot more testing, especially with
a lot of guests

> > The network namespace aims the isolation for now, but the container
> > based on the namespaces will probably need checkpoint/restart and
> > migration ability. The migration is needed not only for servers but
> > for HPC jobs too.
>

> Yes.
>
> > So I don't know what level of isolation/virtualization is really
> > needed by users, what should be acceptable (strong isolation and
> > overhead / weak isolation and efficiency). I don't know if people
> > wanting strong isolation will not prefer Xen (cleary with much more
> > overhead than your patches ;))

well, Xen claims something below 2% IIRC, and would
be clearly the better choice if you want strict
separation with the complete functionality, especially
with hardware support

> We need a clean abstraction that optimizes well.
>
> However local communication between containers is not what we
> should benchmark. That can always be improved later. So long as
> the performance is reasonable. What needs to be benchmarked is the
> overhead of namespaces when connected to physical networking devices
> and on their own local loopback, and comparing that to a kernel
> without namespace support.

well, for me (obviously advocating the lightweight case)
it seems improtant that the following conditions are met:

- loopback traffic inside a guest is insignificantly
  slower than on a normal system

- loopback traffic on the host is insignificantly
  slower than on a normal system

- inter guest traffic is faster than on-wire traffic,
  and should be withing a small tolerance of the
  loopback case (as it really isn't different)

- network (on-wire) traffic should be as fast as without
  the namespace (i.e. within 1% or so, better not really
  measurable)

- all this should be true in a setup with a significant
  number of guests, when only one guest is active, but
  all other guests are ready/configured

- all this should scale well with a few hundred guests

> If we don't hurt that core case we have an implementation we can
> merge. There are a lot of optimization opportunities for local
> communications and we can do that after we have a correct and accepted

> implementation. Anything else is optimizing too soon, and will
> just be muddying the waters.

what I fear is that once something is in, the kernel will
just become slower (as it already did in some areas) and
nobody will care/be-able to fix that later on ...

best,
Herbert

> Eric

Subject: Re: [patch 2/6] [Network namespace] Network device sharing by view
Posted by ebiederm on Tue, 27 Jun 2006 16:47:29 GMT
View Forum Message <> Reply to Message

Herbert Poetzl <herbert@13thfloor.at> writes:

> On Tue, Jun 27, 2006 at 05:52:52AM -0600, Eric W. Biederman wrote:
>>
>> Inside the containers I want all network devices named eth0!
>
> huh? even if there are two of them? also tun?
>
> I think you meant, you want to be able to have eth0 in
> _more_ than one guest where eth0 in a guest can also
> be/use/relate to eth1 on the host, right?

Right I want to have an eth0 in each guest where eth0 is
it's own network device and need have no relationship to
eth0 on the host.

>> We need a clean abstraction that optimizes well.
>>
>> However local communication between containers is not what we
>> should benchmark. That can always be improved later. So long as
>> the performance is reasonable. What needs to be benchmarked is the
>> overhead of namespaces when connected to physical networking devices
>> and on their own local loopback, and comparing that to a kernel
>> without namespace support.
>
> well, for me (obviously advocating the lightweight case)
> it seems improtant that the following conditions are met:
>
> - loopback traffic inside a guest is insignificantly
>   slower than on a normal system
>

> - loopback traffic on the host is insignificantly
>   slower than on a normal system
>
> - inter guest traffic is faster than on-wire traffic,
>   and should be withing a small tolerance of the
>   loopback case (as it really isn't different)
>
> - network (on-wire) traffic should be as fast as without
>   the namespace (i.e. within 1% or so, better not really
>   measurable)
>
> - all this should be true in a setup with a significant
>   number of guests, when only one guest is active, but
>   all other guests are ready/configured
>
> - all this should scale well with a few hundred guests

Ultimately I agree. However. Only host performance should be
a merge blocker. Allowing us to go back and reclaim the few
percentage points we lost later.

>> If we don't hurt that core case we have an implementation we can
>> merge. There are a lot of optimization opportunities for local
>> communications and we can do that after we have a correct and accepted
>> implementation. Anything else is optimizing too soon, and will
>> just be muddying the waters.
>
> what I fear is that once something is in, the kernel will
> just become slower (as it already did in some areas) and
> nobody will care/be-able to fix that later on ...

If nobody cares it doesn't matter.

If no one can fix it that is a problem. Which is why we need
high standards and clean code, not early optimizations.

But on that front each step of the way must be justified on
it's own merits. Not because it will give us some holy grail.

The way to keep the inter guest performance from degrading is
to measure it an complain. But the linux network stack is too
big to get in one pass.

Eric

Subject: Re: [patch 2/6] [Network namespace] Network device sharing by view

Posted by Alexey Kuznetsov on Tue, 27 Jun 2006 16:49:09 GMT
View Forum Message <> Reply to Message

On Tue, Jun 27, 2006 at 06:02:42PM +0200, Herbert Poetzl wrote:

> - loopback traffic inside a guest is insignificantly
> slower than on a normal system
>
> - loopback traffic on the host is insignificantly
> slower than on a normal system
>
> - inter guest traffic is faster than on-wire traffic,
> and should be withing a small tolerance of the
> loopback case (as it really isn't different)

I do not follow what are you people arguing about?

Intra-guest, guest-guest and host-guest paths have _no_ differences
from host-host loopback. Only the device is different:
* virtual loopback for intra-guest
* virtual interface for guest-guest and host-guest

But the work is exactly the same, only the place where packets
looped back is different. How could this be issue to break a lance over? :-)

Alexey

PS. The only thing, which I can imagine is "optimized" out ip_route_input()
in the case of loopback. But this optimization was an obvious design mistake
(mine, sorry) and apparently will die together with removal of current
deficiences of routing cache. Actually, it is one of deficiences.

Subject: Re: [patch 2/6] [Network namespace] Network device sharing by view
Posted by Ben Greear on Tue, 27 Jun 2006 17:19:23 GMT
View Forum Message <> Reply to Message

Eric W. Biederman wrote:
> Herbert Poetzl <herbert@13thfloor.at> writes:
>
>
>>On Tue, Jun 27, 2006 at 05:52:52AM -0600, Eric W. Biederman wrote:
>>
>>>Inside the containers I want all network devices named eth0!
>>
>>huh? even if there are two of them? also tun?
>>

>>I think you meant, you want to be able to have eth0 in
>>_more_ than one guest where eth0 in a guest can also
>>be/use/relate to eth1 on the host, right?
>
>
> Right I want to have an eth0 in each guest where eth0 is
> it's own network device and need have no relationship to
> eth0 on the host.

How does that help anything? Do you envision programs
that make special decisions on whether the interface is
called eth0 v/s eth151?

Ben

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

Subject: Re: [patch 2/6] [Network namespace] Network device sharing by view
Posted by Herbert Poetzl on Tue, 27 Jun 2006 22:52:13 GMT
View Forum Message <> Reply to Message

On Tue, Jun 27, 2006 at 10:19:23AM -0700, Ben Greear wrote:
> Eric W. Biederman wrote:
> >Herbert Poetzl <herbert@13thfloor.at> writes:
> >
> >
> >>On Tue, Jun 27, 2006 at 05:52:52AM -0600, Eric W. Biederman wrote:
> >>
> >>>Inside the containers I want all network devices named eth0!
> >>
> >>huh? even if there are two of them? also tun?
> >>
> >>I think you meant, you want to be able to have eth0 in
> >>_more_ than one guest where eth0 in a guest can also
> >>be/use/relate to eth1 on the host, right?
> >
> >
> >Right I want to have an eth0 in each guest where eth0 is
> >it's own network device and need have no relationship to
> >eth0 on the host.
>
> How does that help anything? Do you envision programs
> that make special decisions on whether the interface is
> called eth0 v/s eth151?

well, those poor folks who do not have ethernet
devices for networking :)

seriously, what I think Eric meant was that it
might be nice (especially for migration purposes)
to keep the device namespace completely virtualized
and not just isolated ...

I'm fine with that, as long as it does not add
overhead or complicate handling, and as far as I
can tell, it should not do that ...

best,
Herbert

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

Subject: Re: [patch 2/6] [Network namespace] Network device sharing by view
Posted by Dave Hansen on Tue, 27 Jun 2006 23:12:52 GMT

View Forum Message <> Reply to Message

On Wed, 2006-06-28 at 00:52 +0200, Herbert Poetzl wrote:
> seriously, what I think Eric meant was that it
> might be nice (especially for migration purposes)
> to keep the device namespace completely virtualized
> and not just isolated ...

It might be nice, but it is probably unneeded for an initial
implementation. In practice, a cluster doing
checkpoint/restart/migration will already have a system in place for
assigning unique IPs or other identifiers to each container. It could
just as easily make sure to assign unique network device names to
containers.

The issues really only come into play when you have an unstructured set
of machines and you want to migrate between them without having prepared
them with any kind of unique net device names beforehand.

It may look weird, but do application really *need* to see eth0 rather
than eth858354?

-- Dave

Subject: Re: [patch 2/6] [Network namespace] Network device sharing by view
Posted by Alexey Kuznetsov on Tue, 27 Jun 2006 23:42:10 GMT
View Forum Message <> Reply to Message

Hello!

> It may look weird, but do application really *need* to see eth0 rather
> than eth858354?

Applications do not care, humans do. :-)

What's about applications they just need to see exactly the same device
after migration. Not only name, but f.e. also its ifindex. If you do not
create a separate namespace for netdevices, you will inevitably end up
with some strange hack sort of VPIDs to translate (or to partition) ifindices
or to tell that "ping -I eth858354 xxx" is too coimplicated application
to survive migration.

Alexey

Subject: Re: [patch 2/6] [Network namespace] Network device sharing by view
Posted by ebiederm on Wed, 28 Jun 2006 03:38:14 GMT
View Forum Message <> Reply to Message

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

> Hello!
>
>> It may look weird, but do application really *need* to see eth0 rather
>> than eth858354?
>
> Applications do not care, humans do. :-)
>
> What's about applications they just need to see exactly the same device
> after migration. Not only name, but f.e. also its ifindex. If you do not
> create a separate namespace for netdevices, you will inevitably end up
> with some strange hack sort of VPIDs to translate (or to partition) ifindices
> or to tell that "ping -I eth858354 xxx" is too coimplicated application
> to survive migration.

Actually there are applications with peculiar licensing practices that
do look at devices like eth0 to verify you have the appropriate mac, and

do really weird things if you don't have an eth0.

Plus there are other cases where it can be simpler to hard code things
if it is allowable. (The human factor) Otherwise your configuration
must be done through hotplug scripts.

But yes there are misguided applications that care.

Eric

Subject: Re: [patch 2/6] [Network namespace] Network device sharing by view
Posted by Herbert Poetzl on Wed, 28 Jun 2006 13:36:40 GMT

View Forum Message <> Reply to Message

On Tue, Jun 27, 2006 at 09:38:14PM -0600, Eric W. Biederman wrote:
> Alexey Kuznetsov <kuznet@ms2.inr.ac.ru> writes:
>
> > Hello!
> >
> >> It may look weird, but do application really *need* to see eth0 rather
> >> than eth858354?
> >
> > Applications do not care, humans do. :-)
> >
> > What's about applications they just need to see exactly the same
> > device after migration. Not only name, but f.e. also its ifindex.
> > If you do not create a separate namespace for netdevices, you will
> > inevitably end up with some strange hack sort of VPIDs to translate
> > (or to partition) ifindices or to tell that "ping -I eth858354 xxx"
> > is too coimplicated application to survive migration.
>
>
> Actually there are applications with peculiar licensing practices that
> do look at devices like eth0 to verify you have the appropriate mac, and
> do really weird things if you don't have an eth0.
>
> Plus there are other cases where it can be simpler to hard code things
> if it is allowable. (The human factor) Otherwise your configuration
> must be done through hotplug scripts.
>
> But yes there are misguided applications that care.

last time I pointed to such 'misguided' apps which
made assumptions that are not necessarily true
inside a virtual environment (e.g. pstree, initpid)
the general? position was that those apps should
be fixed instead adding a 'workaround'

note: personally I'm absolutely not against virtualizing
the device names so that each guest can have a separate
name space for devices, but there should be a way to
'see' _and_ 'identify' the interfaces from outside
(i.e. host or spectator context)

best,
Herbert

> Eric

Subject: Re: [patch 2/6] [Network namespace] Network device sharing by view
Posted by jamal on Wed, 28 Jun 2006 13:53:23 GMT
View Forum Message <> Reply to Message

On Wed, 2006-28-06 at 15:36 +0200, Herbert Poetzl wrote:

> note: personally I'm absolutely not against virtualizing
> the device names so that each guest can have a separate
> name space for devices, but there should be a way to
> 'see' _and_ 'identify' the interfaces from outside
> (i.e. host or spectator context)
>

Makes sense for the host side to have naming convention tied
to the guest. Example as a prefix: guest0-eth0. Would it not
be interesting to have the host also manage these interfaces
via standard tools like ip or ifconfig etc? i.e if i admin up
guest0-eth0, then the user in guest0 will see its eth0 going
up.

Anyways, interesting discussion.

cheers,
jamal

Subject: Re: [patch 2/6] [Network namespace] Network device sharing by view
Posted by ebiederm on Wed, 28 Jun 2006 14:21:38 GMT
View Forum Message <> Reply to Message

Herbert Poetzl <herbert@13thfloor.at> writes:

> last time I pointed to such 'misguided' apps which
> made assumptions that are not necessarily true

> inside a virtual environment (e.g. pstree, initpid)
> the general? position was that those apps should
> be fixed instead adding a 'workaround'

I agree that if it was solely misguided apps. There would be
no justification.

One of the standard applications interfaces we support is renaming
a network interface. So supporting those misguided apps is a actually
a side effect of supporting one the standard operations on a network interface.

Another way of looking at it is that the names of networking devices like the
names of devices in /dev are a user space policy (today). In the
configuration of the networking stack historically we had those identifiers
hard coded. It isn't until just recently that user space has been able to
cope with dynamically added/removed network devices.

As for initpid and friends. In the context where you are simply isolating
pids and not doing a full pid namespaces it was felt that changing the few
user space applications that care was easier and probably worth doing anyway.

> note: personally I'm absolutely not against virtualizing
> the device names so that each guest can have a separate
> name space for devices, but there should be a way to
> 'see' _and_ 'identify' the interfaces from outside
> (i.e. host or spectator context)

Yep. Basically that interface comes when we fix the sysfs support,
to support per namespace reporting.

Eric

Subject: Re: [patch 2/6] [Network namespace] Network device sharing by view
Posted by ebiederm on Wed, 28 Jun 2006 14:51:33 GMT

View Forum Message <> Reply to Message

Dave Hansen <haveblue@us.ibm.com> writes:

> On Wed, 2006-06-28 at 00:52 +0200, Herbert Poetzl wrote:
>> seriously, what I think Eric meant was that it
>> might be nice (especially for migration purposes)
>> to keep the device namespace completely virtualized
>> and not just isolated ...
>
> It might be nice, but it is probably unneeded for an initial
> implementation. In practice, a cluster doing
> checkpoint/restart/migration will already have a system in place for

> assigning unique IPs or other identifiers to each container. It could
> just as easily make sure to assign unique network device names to
> containers.
>
> The issues really only come into play when you have an unstructured set
> of machines and you want to migrate between them without having prepared
> them with any kind of unique net device names beforehand.
>
> It may look weird, but do application really *need* to see eth0 rather
> than eth858354?

Actually there is a very practical reason we don't need to preserve device names across a migration event between machines, is the only sane thing to do is to generate a hotplug event that says you have removed the old interface and added a new interface.

My expectation is that during migration you will wind up with add and remove events for all of your hardware devices. But most applications because they do not access hardware devices directly will not care.

I haven't looked closely but I suspect this is one area where a container style approach will be noticeably different from a Xen or Vmware style approach.

Eric

Subject: Re: [patch 2/6] [Network namespace] Network device sharing by view
Posted by Sam Vilain on Thu, 29 Jun 2006 21:07:34 GMT
View Forum Message <> Reply to Message

---

jamal wrote:
>> note: personally I'm absolutely not against virtualizing
>> the device names so that each guest can have a separate
>> name space for devices, but there should be a way to
>> 'see' _and_ 'identify' the interfaces from outside
>> (i.e. host or spectator context)
>>
>>
>
> Makes sense for the host side to have naming convention tied
> to the guest. Example as a prefix: guest0-eth0. Would it not
> be interesting to have the host also manage these interfaces
> via standard tools like ip or ifconfig etc? i.e if i admin up
> guest0-eth0, then the user in guest0 will see its eth0 going
> up.

That particular convention only works if you have network namespaces and

UTS namespaces tightly bound. We plan to have them separate - so for
that to work, each network namespace could have an arbitrary "prefix"
that determines what the interface name will look like from the outside
when combined. We'd have to be careful about length limits.

And guest0-eth0 doesn't necessarily make sense; it's not really an
ethernet interface, more like a tun or something.

So, an equally good convention might be to use sequential prefixes on
the host, like "tun", "dummy", or a new prefix - then a property of that
is what the name of the interface is perceived to be to those who are in
the corresponding network namespace.

Then the pragmatic question becomes how to correlate what you see from
`ip addr list' to guests.

Sam.

Subject: strict isolation of net interfaces
Posted by Cedric Le Goater on Thu, 29 Jun 2006 22:14:41 GMT

View Forum Message <> Reply to Message

Sam Vilain wrote:
> jamal wrote:
>>> note: personally I'm absolutely not against virtualizing
>>> the device names so that each guest can have a separate
>>> name space for devices, but there should be a way to
>>> 'see' _and_ 'identify' the interfaces from outside
>>> (i.e. host or spectator context)
>>>
>>>
>> Makes sense for the host side to have naming convention tied
>> to the guest. Example as a prefix: guest0-eth0. Would it not
>> be interesting to have the host also manage these interfaces
>> via standard tools like ip or ifconfig etc? i.e if i admin up
>> guest0-eth0, then the user in guest0 will see its eth0 going
>> up.
>
> That particular convention only works if you have network namespaces and
> UTS namespaces tightly bound. We plan to have them separate - so for
> that to work, each network namespace could have an arbitrary "prefix"
> that determines what the interface name will look like from the outside
> when combined. We'd have to be careful about length limits.
>
> And guest0-eth0 doesn't necessarily make sense; it's not really an
> ethernet interface, more like a tun or something.
>

> So, an equally good convention might be to use sequential prefixes on
> the host, like "tun", "dummy", or a new prefix - then a property of that
> is what the name of the interface is perceived to be to those who are in
> the corresponding network namespace.
>
> Then the pragmatic question becomes how to correlate what you see from
> `ip addr list' to guests.

we could work on virtualizing the net interfaces in the host, map them to
eth0 or something in the guest and let the guest handle upper network layers ?

lo0 would just be exposed relying on skbuff tagging to discriminate traffic
between guests.

```
host                  |  guest 0  |  guest 1  |  guest2
 ----------------------+-----------+-----------+------------- -
  |                     |           |           |
  |-> l0      <-------+-> lo0 ... | lo0       | lo0
  |                     |           |           |
  |-> bar0   <--------+-> eth0    |           |
  |                     |           |           |
  |-> foo0   <--------+-----------+-----------+-> eth0
  |                     |           |           |
  `-> foo0:1  <-------+-----------+-> eth0    |
                        |           |           |
```

is that clear ? stupid ? reinventing the wheel ?

thanks,

C.

Subject: Re: [patch 2/6] [Network namespace] Network device sharing by view
Posted by jamal on Fri, 30 Jun 2006 00:15:52 GMT

View Forum Message <> Reply to Message

On Fri, 2006-30-06 at 09:07 +1200, Sam Vilain wrote:
> jamal wrote:

> > Makes sense for the host side to have naming convention tied
> > to the guest. Example as a prefix: guest0-eth0. Would it not
> > be interesting to have the host also manage these interfaces
> > via standard tools like ip or ifconfig etc? i.e if i admin up

> > guest0-eth0, then the user in guest0 will see its eth0 going
> > up.
>
> That particular convention only works if you have network namespaces and
> UTS namespaces tightly bound.

that would be one approach. Another less sophisticated approach is to have no binding whatsoever, rather some translation table to map two unrelated devices.

> We plan to have them separate - so for
> that to work, each network namespace could have an arbitrary "prefix"
> that determines what the interface name will look like from the outside
> when combined. We'd have to be careful about length limits.
>
> And guest0-eth0 doesn't necessarily make sense; it's not really an
> ethernet interface, more like a tun or something.
>

it wouldnt quiet fit as a tun device. More like a mirror side of the guest eth0 created on the host side
i.e a sort of passthrough device with one side visible on the host (send from guest0-eth0 is received on eth0 in the guest and vice-versa).

Note this is radically different from what i have heard Andrey and co talk about and i dont wanna disturb any shit because there seems to be some agreement. But if you address me i respond because it is very interesting a topic;->

> So, an equally good convention might be to use sequential prefixes on
> the host, like "tun", "dummy", or a new prefix - then a property of that
> is what the name of the interface is perceived to be to those who are in
> the corresponding network namespace.
>
> Then the pragmatic question becomes how to correlate what you see from
> `ip addr list' to guests.

on the host ip addr and the one seen on the guest side are the same.
Except one is seen (on the host) on guest0-eth0 and another is seen on eth0 (on guest).
Anyways, ignore what i am saying if it is disrupting the discussion.

cheers,
jamal

Subject: Re: strict isolation of net interfaces

Posted by serue on Fri, 30 Jun 2006 02:39:47 GMT
View Forum Message <> Reply to Message

Quoting Cedric Le Goater (clg@fr.ibm.com):
> Sam Vilain wrote:
> > jamal wrote:
> >>> note: personally I'm absolutely not against virtualizing
> >>> the device names so that each guest can have a separate
> >>> name space for devices, but there should be a way to
> >>> 'see' _and_ 'identify' the interfaces from outside
> >>> (i.e. host or spectator context)
> >>>
> >>>
> >> Makes sense for the host side to have naming convention tied
> >> to the guest. Example as a prefix: guest0-eth0. Would it not
> >> be interesting to have the host also manage these interfaces
> >> via standard tools like ip or ifconfig etc? i.e if i admin up
> >> guest0-eth0, then the user in guest0 will see its eth0 going
> >> up.
> >
> > That particular convention only works if you have network namespaces and
> > UTS namespaces tightly bound. We plan to have them separate - so for
> > that to work, each network namespace could have an arbitrary "prefix"
> > that determines what the interface name will look like from the outside
> > when combined. We'd have to be careful about length limits.
> >
> > And guest0-eth0 doesn't necessarily make sense; it's not really an
> > ethernet interface, more like a tun or something.
> >
> > So, an equally good convention might be to use sequential prefixes on
> > the host, like "tun", "dummy", or a new prefix - then a property of that
> > is what the name of the interface is perceived to be to those who are in
> > the corresponding network namespace.
> >
> > Then the pragmatic question becomes how to correlate what you see from
> > `ip addr list' to guests.
>
>
> we could work on virtualizing the net interfaces in the host, map them to
> eth0 or something in the guest and let the guest handle upper network layers ?
>
> lo0 would just be exposed relying on skbuff tagging to discriminate traffic
> between guests.

This seems to me the preferable way. We create a full virtual net
device for each new container, and fully virtualize the device
namespace.

> host | guest 0 | guest 1 | guest2

```
>  ----------------------+-----------+-----------+------------- -
>   |                    |           |           |
>   |-> l0      <-------+-> lo0 ... | lo0       | lo0
>   |                    |           |           |
>   |-> bar0   <--------+-> eth0    |           |
>   |                    |           |           |
>   |-> foo0   <--------+-----------+-----------+-> eth0
>   |                    |           |           |
>   `-> foo0:1  <-------+-----------+-> eth0    |
>                        |           |           |
>
>
```

> is that clear ? stupid ? reinventing the wheel ?

The last one in your diagram confuses me - why foo0:1? I would have thought it'd be

```
host                  |  guest 0  |  guest 1  |  guest2
 ----------------------+-----------+-----------+------------- -
  |                    |           |           |
  |-> l0      <-------+-> lo0 ... | lo0       | lo0
  |                    |           |           |
  |-> eth0             |           |           |
  |                    |           |           |
  |-> veth0  <--------+-> eth0    |           |
  |                    |           |           |
  |-> veth1  <--------+-----------+-----------+-> eth0
  |                    |           |           |
  |-> veth2   <-------+-----------+-> eth0    |
```

I think we should avoid using device aliases, as trying to do something like giving eth0:1 to guest1 and eth0:2 to guest2 while hiding eth0:1 from guest2 requires some uglier code (as I recall) than working with full devices. In other words, if a namespace can see eth0, and eth0:2 exists, it should always see eth0:2.

So conceptually using a full virtual net device per container certainly seems cleaner to me, and it seems like it should be simpler by way of statistics gathering etc, but are there actually any real gains? Or is the support for multiple IPs per device actually enough?

Herbert, is this basically how ngnet is supposed to work?

-serge

Subject: Re: strict isolation of net interfaces
Posted by Sam Vilain on Fri, 30 Jun 2006 02:48:49 GMT
View Forum Message <> Reply to Message

Serge E. Hallyn wrote:

```
> The last one in your diagram confuses me - why foo0:1?  I would
> have thought it'd be
>
> host                  |  guest 0  |  guest 1  |  guest2
>  ---------------------+-----------+-----------+------------- -
>   |                   |           |           |
>   |-> l0      <-------+-> lo0 ... | lo0       | lo0
>   |                   |           |           |
>   |-> eth0                |           |           |
>   |                   |           |           |
>   |-> veth0  <--------+-> eth0    |           |
>   |                   |           |           |
>   |-> veth1  <--------+-----------+-----------+-> eth0
>   |                   |           |           |
>   |-> veth2   <-------+-----------+-> eth0    |
>
> [...]
>
> So conceptually using a full virtual net device per container
> certainly seems cleaner to me, and it seems like it should be
> simpler by way of statistics gathering etc, but are there actually
> any real gains?  Or is the support for multiple IPs per device
> actually enough?
>
```

Why special case loopback?

Why not:

```
host                  |  guest 0  |  guest 1  |  guest2
 ---------------------+-----------+-----------+------------- -
  |                   |           |           |
  |-> lo                  |           |           |
  |                   |           |           |
  |-> vlo0  <---------+-> lo      |           |
  |                   |           |           |
  |-> vlo1  <---------+-----------+-----------+-> lo
  |                   |           |           |
  |-> vlo2   <--------+-----------+-> lo      |
  |                   |           |           |
  |-> eth0                |           |           |
  |                   |           |           |
  |-> veth0  <--------+-> eth0    |           |
  |                   |           |           |
  |-> veth1  <--------+-----------+-----------+-> eth0
  |                |          |          |
  |-> veth2   <-------+-----------+-> eth0    |
```

Sam.

Subject: Re: [patch 2/6] [Network namespace] Network device sharing by view
Posted by Herbert Poetzl on Fri, 30 Jun 2006 03:35:55 GMT
View Forum Message <> Reply to Message

On Thu, Jun 29, 2006 at 08:15:52PM -0400, jamal wrote:
> On Fri, 2006-30-06 at 09:07 +1200, Sam Vilain wrote:
> > jamal wrote:
>
> > > Makes sense for the host side to have naming convention tied
> > > to the guest. Example as a prefix: guest0-eth0. Would it not
> > > be interesting to have the host also manage these interfaces
> > > via standard tools like ip or ifconfig etc? i.e if i admin up
> > > guest0-eth0, then the user in guest0 will see its eth0 going
> > > up.
> >
> > That particular convention only works if you have network namespaces
> > and UTS namespaces tightly bound.
>
> that would be one approach. Another less sophisticated approach is to
> have no binding whatsoever, rather some translation table to map two
> unrelated devices.
>
> > We plan to have them separate - so for
> > that to work, each network namespace could have an arbitrary
> > "prefix" that determines what the interface name will look like from
> > the outside when combined. We'd have to be careful about length
> > limits.
> >
> > And guest0-eth0 doesn't necessarily make sense; it's not really an
> > ethernet interface, more like a tun or something.
>
> it wouldnt quiet fit as a tun device. More like a mirror side of the
> guest eth0 created on the host side
> i.e a sort of passthrough device with one side visible on the host (send
> from guest0-eth0 is received on eth0 in the guest and vice-versa).
>
> Note this is radically different from what i have heard Andrey and co
> talk about and i dont wanna disturb any shit because there seems to be
> some agreement. But if you address me i respond because it is very
> interesting a topic;->

thing is, we have several things we should care about
and some of them 'look' or 'sound' similar, although
they are not really ... I'll try to clarify

first, we want to have 'per guest' interfaces, which
do not clash with any interfaces on the host or in
other guests

then, we want to 'connect' them, implicitely or
explicetly with 'other' interfaces or devices inside
other guests or on the host, here we have the following
cases (some are a little special):

- lo interface, guest and host private (by default)
- tap/tun interfaces, again host/guest private
- tun like interfaces between host and guests
- tun like interfaces between guests
- 'normal' interfaces mapped into guests

on the traffic side we have the following cases:

- local traffic on the host
- local traffic on the guest
- local traffic between host and guest
- local traffic between guests
- routed traffic from guest via host
- bridged traffic from guest via host

special cases here would be tun/tap traffic inside
a guest, but that can be considered local too

> > So, an equally good convention might be to use sequential prefixes
> > on the host, like "tun", "dummy", or a new prefix - then a property
> > of that is what the name of the interface is perceived to be to
> > those who are in the corresponding network namespace.
> >
> > Then the pragmatic question becomes how to correlate what you see
> > from `ip addr list' to guests.
>
> on the host ip addr and the one seen on the guest side are the same.
> Except one is seen (on the host) on guest0-eth0 and another is seen
> on eth0 (on guest).

this depends on the way the interfaces are handled
and how they actually work, means:

if the interfaces _solely_ work via routing or

bridging, then the 'host' end has to exist and be
visible similar to 'normal' interfaces

if the traffic is (magically) mapped from guest
interfaces to real (outside) host interfaces, we
might want the same view as the guest has
(i.e. basically a 'copy' which is not real)

> Anyways, ignore what i am saying if it is disrupting the discussion.

IMHO input is always welcome .. helps the folks to
do better thinking :)

> cheers,
> jamal
>
>
>

Subject: Re: [patch 2/6] [Network namespace] Network device sharing by view
Posted by Andrey Savochkin on Fri, 30 Jun 2006 07:45:51 GMT
View Forum Message <> Reply to Message

Hi Jamal,

On Thu, Jun 29, 2006 at 08:15:52PM -0400, jamal wrote:
> On Fri, 2006-30-06 at 09:07 +1200, Sam Vilain wrote:
[snip]
> > We plan to have them separate - so for
> > that to work, each network namespace could have an arbitrary "prefix"
> > that determines what the interface name will look like from the outside
> > when combined. We'd have to be careful about length limits.
> >
> > And guest0-eth0 doesn't necessarily make sense; it's not really an
> > ethernet interface, more like a tun or something.
> >
>
> it wouldnt quiet fit as a tun device. More like a mirror side of the
> guest eth0 created on the host side
> i.e a sort of passthrough device with one side visible on the host (send
> from guest0-eth0 is received on eth0 in the guest and vice-versa).
>
> Note this is radically different from what i have heard Andrey and co
> talk about and i dont wanna disturb any shit because there seems to be
> some agreement. But if you address me i respond because it is very
> interesting a topic;->

I do not have anything against guest-eth0 - eth0 pairs _if_ they are set up
by the host administrators explicitly for some purpose.
For example, if these guest-eth0 and eth0 devices stay as pure virtual ones,
i.e. they don't have any physical NIC, host administrator may route traffic
to guestXX-eth0 interfaces to pass it to the guests.

However, I oppose the idea of automatic mirroring of _all_ devices appearing
inside some namespaces ("guests") to another namespace (the "host").
This clearly goes against the concept of namespaces as independent realms,
and creates a lot of problems with applications running in the host, hotplug
scripts and so on.

>
> > So, an equally good convention might be to use sequential prefixes on
> > the host, like "tun", "dummy", or a new prefix - then a property of that
> > is what the name of the interface is perceived to be to those who are in
> > the corresponding network namespace.
> >
> > Then the pragmatic question becomes how to correlate what you see from
> > `ip addr list' to guests.
>
> on the host ip addr and the one seen on the guest side are the same.
> Except one is seen (on the host) on guest0-eth0 and another is seen
> on eth0 (on guest).

Then what to do if the host system has 10.0.0.1 as a private address on eth3,
and then interfaces guest1-tun0 and guest2-tun0 both get address 10.0.0.1
when each guest has added 10.0.0.1 to their tun0 device?

Regards,

Andrey

Subject: Re: strict isolation of net interfaces
Posted by Cedric Le Goater on Fri, 30 Jun 2006 08:56:13 GMT
View Forum Message <> Reply to Message

Serge E. Hallyn wrote:
>
> The last one in your diagram confuses me - why foo0:1? I would
> have thought it'd be

just thinking aloud. I thought that any kind/type of interface could be
mapped from host to guest.

```
> host                 |  guest 0  |  guest 1  |  guest2
>  ----------------------+-----------+-----------+------------- -
>   |                |          |          |
>   |-> l0      <-------+-> lo0 ... | lo0       | lo0
>   |                |          |          |
>   |-> eth0            |          |          |
>   |                |          |          |
>   |-> veth0  <--------+-> eth0    |          |
>   |                |          |          |
>   |-> veth1  <--------+-----------+-----------+-> eth0
>   |                |          |          |
>   |-> veth2   <-------+-----------+-> eth0    |
>
```

> I think we should avoid using device aliases, as trying to do
> something like giving eth0:1 to guest1 and eth0:2 to guest2
> while hiding eth0:1 from guest2 requires some uglier code (as
> I recall) than working with full devices. In other words,
> if a namespace can see eth0, and eth0:2 exists, it should always
> see eth0:2.
>
> So conceptually using a full virtual net device per container
> certainly seems cleaner to me, and it seems like it should be
> simpler by way of statistics gathering etc, but are there actually
> any real gains? Or is the support for multiple IPs per device
> actually enough?
>
> Herbert, is this basically how ngnet is supposed to work?

Subject: Re: strict isolation of net interfaces
Posted by Daniel Lezcano on Fri, 30 Jun 2006 12:23:16 GMT
View Forum Message <> Reply to Message

Serge E. Hallyn wrote:
> Quoting Cedric Le Goater (clg@fr.ibm.com):
>
>>we could work on virtualizing the net interfaces in the host, map them to
>>eth0 or something in the guest and let the guest handle upper network layers ?
>>
>>lo0 would just be exposed relying on skbuff tagging to discriminate traffic
>>between guests.
>
>
> This seems to me the preferable way. We create a full virtual net
> device for each new container, and fully virtualize the device
> namespace.

I have a few questions about all the network isolation stuff:

  * What level of isolation is wanted for the network ? network devices

? IPv4/IPv6 ? TCP/UDP ?

   * How is handled the incoming packets from the network ? I mean what will be mecanism to dispatch the packet to the right virtual device ?

   * How to handle the SO_BINDTODEVICE socket option ?

   * Has the virtual device a different MAC address ? How to manage it with the real MAC address on the system ? How to manage ARP, ICMP, multicasting and IP ?

It seems for me, IMHO that will require a lot of translation and browsing table. It will probably add a very significant overhead.

    * How to handle NFS access mounted outside of the container ?

    * How to handle ICMP_REDIRECT ?

Regards

Subject: Re: [patch 2/6] [Network namespace] Network device sharing by view
Posted by jamal on Fri, 30 Jun 2006 13:50:52 GMT

View Forum Message <> Reply to Message

Hi Andrey,

BTW - I was just looking at openvz, very impressive. To the other folks,
I am not putting down any of your approaches - just havent
had time to study them. Andrey, this is the same thing you guys have
been working on for a few years now, you just changed the name, correct?

Ok, since you guys are encouraging me to speak, here goes ;->
Hopefully this addresses the other email from Herbert et al.

On Fri, 2006-30-06 at 11:45 +0400, Andrey Savochkin wrote:
> Hi Jamal,
>
> On Thu, Jun 29, 2006 at 08:15:52PM -0400, jamal wrote:
> > On Fri, 2006-30-06 at 09:07 +1200, Sam Vilain wrote:
> [snip]

>
> I do not have anything against guest-eth0 - eth0 pairs _if_ they are set up
> by the host administrators explicitly for some purpose.
> For example, if these guest-eth0 and eth0 devices stay as pure virtual ones,
> i.e. they don't have any physical NIC, host administrator may route traffic
> to guestXX-eth0 interfaces to pass it to the guests.

>

Well there will be purely virtual of course. Something along the lines
for openvz:

```
// create the guest
[host-node]# vzctl create 101 --ostemplate fedora-core-5-minimal
// create guest101::eth0, seems to only create config to boot up with
[host-node]# vzctl create 101 --netdev eth0
// bootup guest101
[host-node]# vzctl start 101
```

As soon as bootup of guest101 happens, creating guest101::eth0 should activate
creation of the host side netdevice. This could be triggered for example by
the netlink event message seen on host whic- which is a result of creating guest101::eth0
Which means control sits purely in user space.

at that point if i do ifconfig on host i see g101-eth0
on guest101 i see just name eth0.

My earlier suggestion was that instead of:
host-node]# vzctl set 101 --ipadd 10.1.2.3

you do:
host-node]# ip addr add g101-eth0 10.1.2.3/32
you should still use vzctl to save config for next bootup

> However, I oppose the idea of automatic mirroring of _all_ devices appearing
> inside some namespaces ("guests") to another namespace (the "host").
> This clearly goes against the concept of namespaces as independent realms,
> and creates a lot of problems with applications running in the host, hotplug
> scripts and so on.
>

I was thinking that the host side is the master i.e you can peek at
namespaces in the guest from the host.
Also note that having the pass through device allows for guests to be
connected via standard linux schemes in the host side (bridge, point
routes, tc redirect etc); so you dont need a speacial device to hook
them together.

> > > Then the pragmatic question becomes how to correlate what you see from
> > > `ip addr list' to guests.
> >
> > on the host ip addr and the one seen on the guest side are the same.
> > Except one is seen (on the host) on guest0-eth0 and another is seen
> > on eth0 (on guest).
>

> Then what to do if the host system has 10.0.0.1 as a private address on eth3,
> and then interfaces guest1-tun0 and guest2-tun0 both get address 10.0.0.1
> when each guest has added 10.0.0.1 to their tun0 device?
>

Yes, that would be a conflict that needs to be resolved. If you look at
ip addresses as also belonging to namespaces, then it should work, no?
i am assuming a tag at the ifa table level.

cheers,
jamal

Subject: Re: strict isolation of net interfaces
Posted by ebiederm on Fri, 30 Jun 2006 14:20:24 GMT
View Forum Message <> Reply to Message

Daniel Lezcano <dlezcano@fr.ibm.com> writes:

> Serge E. Hallyn wrote:
>> Quoting Cedric Le Goater (clg@fr.ibm.com):
>>
>>>we could work on virtualizing the net interfaces in the host, map them to
>>>eth0 or something in the guest and let the guest handle upper network layers ?
>>>
>>>lo0 would just be exposed relying on skbuff tagging to discriminate traffic
>>>between guests.
>> This seems to me the preferable way. We create a full virtual net
>> device for each new container, and fully virtualize the device
>> namespace.
>
> I have a few questions about all the network isolation stuff:

So far I have seen two viable possibilities on the table,
neither of them involve multiple names for a network device.

layer 3 (filtering the allowed ip addresses at bind time roughly the current vserver).
  - implementable as a security hook.
  - Benefit no measurable performance impact.
  - Downside not many things we can do.

layer 2 (What appears to applications a separate instance of the network stack).
  - Implementable as a namespace.
  - Each network namespace would have dedicated network devices.
  - Benefit extremely flexible.
  - Downside since at least the slow path must examine the packet
    it has the possibility of slowing down the networking stack.

For me the important characteristics.
- Allows for application migration, when we take our ip address with us.
  In particular it allows for importation of addresses assignments
  mad on other machines.

- No measurable impact on the existing networking when the code
  is compiled in.

- Clean predictable semantics.

This whole debate on network devices show up in multiple network namespaces
is just silly. The only reason for wanting that appears to be better management.
We have deeper issues like can we do a reasonable implementation without a
network device showing up in multiple namespaces.

I think the reason the debate exists at all is that it is a very approachable
topic, as opposed to the fundamentals here.

If we can get layer 2 level isolation working without measurable overhead
with one namespace per device it may be worth revisiting things. Until
then it is a side issue at best.

Eric

Subject: Re: [patch 2/6] [Network namespace] Network device sharing by view
Posted by Andrey Savochkin on Fri, 30 Jun 2006 15:01:55 GMT

View Forum Message <> Reply to Message

Jamal,

On Fri, Jun 30, 2006 at 09:50:52AM -0400, jamal wrote:
>
> BTW - I was just looking at openvz, very impressive. To the other folks,

Thanks!

> I am not putting down any of your approaches - just havent
> had time to study them. Andrey, this is the same thing you guys have
> been working on for a few years now, you just changed the name, correct?

The relations are more complicated than just the change of name,
but yes, OpenVZ represents the result of our work for a few years.

>
> Ok, since you guys are encouraging me to speak, here goes ;->

> Hopefully this addresses the other email from Herbert et al.
>
[snip]
> // create the guest
> [host-node]# vzctl create 101 --ostemplate fedora-core-5-minimal
> // create guest101::eth0, seems to only create config to boot up with
> [host-node]# vzctl create 101 --netdev eth0
> // bootup guest101
> [host-node]# vzctl start 101
>
> As soon as bootup of guest101 happens, creating guest101::eth0 should activate
> creation of the host side netdevice. This could be triggered for example by
> the netlink event message seen on host whic- which is a result of creating guest101::eth0
> Which means control sits purely in user space.

I'd like to clarify you idea: whether this host-side device is a real
device capable of receiving and transmitting packets (by moving them between
namespaces), or it's a fake device creating only a view of other namespace's
devices?

[snip]
> > However, I oppose the idea of automatic mirroring of _all_ devices appearing
> > inside some namespaces ("guests") to another namespace (the "host").
> > This clearly goes against the concept of namespaces as independent realms,
> > and creates a lot of problems with applications running in the host, hotplug
> > scripts and so on.
> >
>
> I was thinking that the host side is the master i.e you can peek at
> namespaces in the guest from the host.

"Host(master)-guest" relations is a valid and useful scheme.
However, I'm thinking about broader application of network namespaces,
when they can form an arbitrary tree and may not be in "host-guest" relations.

> Also note that having the pass through device allows for guests to be
> connected via standard linux schemes in the host side (bridge, point
> routes, tc redirect etc); so you dont need a speacial device to hook
> them together.

What do you mean under pass through device?
Do you mean using guest1-tun0 as a backdoor to talk to the guest?

>
> > > > Then the pragmatic question becomes how to correlate what you see from
> > > > `ip addr list' to guests.
> > >
> > > on the host ip addr and the one seen on the guest side are the same.

> > > Except one is seen (on the host) on guest0-eth0 and another is seen
> > > on eth0 (on guest).
> >
> > Then what to do if the host system has 10.0.0.1 as a private address on eth3,
> > and then interfaces guest1-tun0 and guest2-tun0 both get address 10.0.0.1
> > when each guest has added 10.0.0.1 to their tun0 device?
> >
>
> Yes, that would be a conflict that needs to be resolved. If you look at
> ip addresses as also belonging to namespaces, then it should work, no?
> i am assuming a tag at the ifa table level.

I'm not sure, it's complicated.
You wouldn't want automatic local routes to be added for IP addresses on
the host-side interfaces, right?
Do you expect these IP addresses to act as local addresses in other places,
like answering to arp requests about these IP on all physical devices?

But anyway, you'll have conflicts on the application level.
Many programs like ntpd, bind, and others fetch the device list using the
same ioctls as ifconfig, and make (un)intelligent decisions basing on what
they see.
Mirroring may have some advantages if I am both host and guest administrator.
But if I create a namespace for my friend Joe to play with IPv6 and sit
tunnels, why should I face inconveniences because of what he does there?

Best regards

Andrey

Subject: Re: strict isolation of net interfaces
Posted by Daniel Lezcano on Fri, 30 Jun 2006 15:22:51 GMT
View Forum Message <> Reply to Message

Eric W. Biederman wrote:
> Daniel Lezcano <dlezcano@fr.ibm.com> writes:
>
>
>>Serge E. Hallyn wrote:
>>
>>>Quoting Cedric Le Goater (clg@fr.ibm.com):
>>>
>>>
>>>>we could work on virtualizing the net interfaces in the host, map them to
>>>>eth0 or something in the guest and let the guest handle upper network layers ?
>>>>
>>>>lo0 would just be exposed relying on skbuff tagging to discriminate traffic

>>>>between guests.
>>>
>>>This seems to me the preferable way. We create a full virtual net
>>>device for each new container, and fully virtualize the device
>>>namespace.
>>
>>I have a few questions about all the network isolation stuff:
>

It seems these questions are not important.

>
> So far I have seen two viable possibilities on the table,
> neither of them involve multiple names for a network device.
>
> layer 3 (filtering the allowed ip addresses at bind time roughly the current vserver).
>   - implementable as a security hook.
>   - Benefit no measurable performance impact.
>   - Downside not many things we can do.

What things ? Can you develop please ? Can you give some examples ?

>
> layer 2 (What appears to applications a separate instance of the network stack).
>   - Implementable as a namespace.

what about accessing a NFS mounted outside the container ?

>   - Each network namespace would have dedicated network devices.
>   - Benefit extremely flexible.

For what ? For who ? Do you have examples ?

>   - Downside since at least the slow path must examine the packet
>   it has the possibility of slowing down the networking stack.

What is/are the slow path(s) you identified ?

> For me the important characteristics.
> - Allows for application migration, when we take our ip address with us.
>   In particular it allows for importation of addresses assignments
>   mad on other machines.

Ok for the two methods no ?

> - No measurable impact on the existing networking when the code
>   is compiled in.

You contradict ...

> - Clean predictable semantics.

What that means ? Can you explain, please ?

> This whole debate on network devices show up in multiple network namespaces
> is just silly.

The debate is not on the network device show up. The debate is can we have a network isolation ___usable for everybody___ not only for the beauty of having namespaces and for a system container like.

I am not against the network device virtualization or against the namespaces. I am just asking if the namespace is the solution for all the network isolation. Should we nest layer 2 and layer 3 vitualization into namespaces or separate them in order to have the flexibility to choose isolation/performance.

> The only reason for wanting that appears to be better management.
> We have deeper issues like can we do a reasonable implementation without a
> network device showing up in multiple namespaces.

Again, I am not against having the network device virtualization. It is a good idea.

> I think the reason the debate exists at all is that it is a very approachable
> topic, as opposed to the fundamentals here.
>
> If we can get layer 2 level isolation working without measurable overhead
> with one namespace per device it may be worth revisiting things. Until
> then it is a side issue at best.

I agree, so where are the answers of the questions I asked in my previous email ? You said you did some implementation of network isolation with and without namespaces, so you should be able to answer...

-- Daniel

Subject: Re: strict isolation of net interfaces
Posted by serue on Fri, 30 Jun 2006 16:14:42 GMT
View Forum Message <> Reply to Message

Quoting Eric W. Biederman (ebiederm@xmission.com):
> This whole debate on network devices show up in multiple network namespaces
> is just silly. The only reason for wanting that appears to be better management.

A damned good reason. Clearly we want the parent namespace to be able
to control what the child can do. So whatever interface a child gets,
the parent should be able to somehow address. Simple iptables rules
controlling traffic between it's own netdevice and the one it hands it's
children seem a good option.

> We have deeper issues like can we do a reasonable implementation without a
> network device showing up in multiple namespaces.

Isn't that the same issue?

> If we can get layer 2 level isolation working without measurable overhead
> with one namespace per device it may be worth revisiting things. Until
> then it is a side issue at best.

Ok, and in the meantime we can all use the network part of the bsdjail
lsm? :)

-serge

Subject: Re: strict isolation of net interfaces
Posted by ebiederm on Fri, 30 Jun 2006 17:41:59 GMT
View Forum Message <> Reply to Message

"Serge E. Hallyn" <serue@us.ibm.com> writes:

> Quoting Eric W. Biederman (ebiederm@xmission.com):
>> This whole debate on network devices show up in multiple network namespaces
>> is just silly. The only reason for wanting that appears to be better
> management.
>
> A damned good reason.

Better management is a good reason. But constructing the management in
a way that hampers the implementation and confuses existing applications is
a problem.

Things are much easier if namespaces are completely independent.

Among other things the semantics are clear and obvious.

> Clearly we want the parent namespace to be able
> to control what the child can do. So whatever interface a child gets,
> the parent should be able to somehow address. Simple iptables rules
> controlling traffic between it's own netdevice and the one it hands it's
> children seem a good option.

That or we setup the child and then drop CAP_NET_ADMIN.

>> We have deeper issues like can we do a reasonable implementation without a
>> network device showing up in multiple namespaces.
>
> Isn't that the same issue?

I guess I was thinking from the performance and cleanliness point of
view.

>> If we can get layer 2 level isolation working without measurable overhead
>> with one namespace per device it may be worth revisiting things. Until
>> then it is a side issue at best.
>
> Ok, and in the meantime we can all use the network part of the bsdjail
> lsm? :)

If necessary. But mostly we concentrate on the fundamentals and figure
out what it takes to take the level 2 stuff working.

Eric

Subject: Re: strict isolation of net interfaces
Posted by ebiederm on Fri, 30 Jun 2006 17:58:47 GMT

View Forum Message <> Reply to Message

Daniel Lezcano <dlezcano@fr.ibm.com> writes:

> Eric W. Biederman wrote:
>> Daniel Lezcano <dlezcano@fr.ibm.com> writes:
>>
>>>Serge E. Hallyn wrote:
>>>
>>>>Quoting Cedric Le Goater (clg@fr.ibm.com):
>>>>
>>>>
>>>>>we could work on virtualizing the net interfaces in the host, map them to
>>>>>eth0 or something in the guest and let the guest handle upper network layers
> ?
>>>>>
>>>>>lo0 would just be exposed relying on skbuff tagging to discriminate traffic
>>>>>between guests.
>>>>
>>>>This seems to me the preferable way. We create a full virtual net
>>>>device for each new container, and fully virtualize the device
>>>>namespace.

>>>
>>>I have a few questions about all the network isolation stuff:
>>
>
> It seems these questions are not important.

I'm just trying to get us back to a productive topic.

>> So far I have seen two viable possibilities on the table,
>> neither of them involve multiple names for a network device.
>> layer 3 (filtering the allowed ip addresses at bind time roughly the current
>> vserver).
>>   - implementable as a security hook.
>>   - Benefit no measurable performance impact.
>>   - Downside not many things we can do.
>
> What things ? Can you develop please ? Can you give some examples ?

DHCP, tcpdump,.. Probably a bad way of phrasing it. But there
is a lot more that we can do using a pure layer 2 approach.

>> layer 2 (What appears to applications a separate instance of the network
>> stack).
>>   - Implementable as a namespace.
>
> what about accessing a NFS mounted outside the container ?

As I replied earlier it isn't a problem. If you get to it through the
filesystem namespace it uses the network namespace it was mounted with
for it's connection.

>>   - Each network namespace would have dedicated network devices.
>>   - Benefit extremely flexible.
>
> For what ? For who ? Do you have examples ?

See above.

>>   - Downside since at least the slow path must examine the packet
>>   it has the possibility of slowing down the networking stack.
>
> What is/are the slow path(s) you identified ?

Grr. I put that badly. Basically at least on the slow path you need to
look at a per network namespace data structure. The extra pointer
indirection could slow things down. The point is that we may be
able to have a fast path that is exactly the same as the rest
of the network stack.

If the obvious approach does not work my gut the feeling the
network stack fast path will give us an implementation without overhead.

>> For me the important characteristics.
>> - Allows for application migration, when we take our ip address with us.
>>   In particular it allows for importation of addresses assignments
>>   mad on other machines.
>
> Ok for the two methods no ?

So far.

>> - No measurable impact on the existing networking when the code
>>   is compiled in.
>
> You contradict ...

How so? As far as I can tell this is a basic requirement to get
merged.

>> - Clean predictable semantics.
>
> What that means ? Can you explain, please ?

>> This whole debate on network devices show up in multiple network namespaces
>> is just silly.
>
> The debate is not on the network device show up. The debate is can we have a
> network isolation ___usable for everybody___ not only for the beauty of having
> namespaces and for a system container like.

This subthread talking about devices showing up in multiple namespaces seemed
very much exactly on how network devices show up.

> I am not against the network device virtualization or against the namespaces. I
> am just asking if the namespace is the solution for all the network
> isolation. Should we nest layer 2 and layer 3 vitualization into namespaces or
> separate them in order to have the flexibility to choose isolation/performance.

I believe I addressed Herbert Poetzl's concerns earlier. To me the question
is can we implement an acceptable layer 2 solution, that distrubutions and
other people who do not need isolation would have no problem compiling in
by default.

The joy of namespaces is that if you don't want it you don't have to use it.
Layer 2 can do everything and is likely usable by everyone iff the performance
is acceptable.

>> The only reason for wanting that appears to be better management.
>> We have deeper issues like can we do a reasonable implementation without a
>> network device showing up in multiple namespaces.
>
> Again, I am not against having the network device virtualization. It is a good
> idea.
>
>> I think the reason the debate exists at all is that it is a very approachable
>> topic, as opposed to the fundamentals here.
>> If we can get layer 2 level isolation working without measurable overhead
>> with one namespace per device it may be worth revisiting things. Until
>> then it is a side issue at best.
>
> I agree, so where are the answers of the questions I asked in my previous email
> ? You said you did some implementation of network isolation with and without
> namespaces, so you should be able to answer...

Sorry. More than anything those questions looked retorical and aimed
at disarming some of the silliness. I will go back and try and
answer those. Fundamentally when we have one namespace that includes
network devices, network sockets, and all of the data structures necessary
to use them (routing tables and the like) and we have a tunnel device
that can connect namespaces the answers are trivial and I though obvious.

Eric

Subject: Re: strict isolation of net interfaces
Posted by ebiederm on Fri, 30 Jun 2006 18:09:44 GMT
View Forum Message <> Reply to Message

Daniel Lezcano <dlezcano@fr.ibm.com> writes:

> Serge E. Hallyn wrote:
>> Quoting Cedric Le Goater (clg@fr.ibm.com):
>>
>>>we could work on virtualizing the net interfaces in the host, map them to
>>>eth0 or something in the guest and let the guest handle upper network layers ?
>>>
>>>lo0 would just be exposed relying on skbuff tagging to discriminate traffic
>>>between guests.
>> This seems to me the preferable way. We create a full virtual net
>> device for each new container, and fully virtualize the device
>> namespace.
>

Answers with respect to how I see layer 2 isolation,

with network devices and sockets as well as the associated routing
information given per namespace.

> I have a few questions about all the network isolation stuff:
>
> * What level of isolation is wanted for the network ? network devices ?
> IPv4/IPv6 ? TCP/UDP ?
>
> * How is handled the incoming packets from the network ? I mean what will be
> mecanism to dispatch the packet to the right virtual device ?

Wrong question. A better question is to ask how do you know which namespace
a packet is in.
Answer: By looking at which device or socket it just came from.

How do you get a packet into a non-default namespace?
Either you move a real network interface into that namespace.
Or you use a tunnel device that shows up as two network interfaces in
two different namespaces.

Then you route, or bridge packets between the two. Trivial.

> * How to handle the SO_BINDTODEVICE socket option ?

Just like we do now.

> * Has the virtual device a different MAC address ?

All network devices are abstractions of the hardware so they are all
sort of virtual. My implementation of a tunnel device has a mac
address so I can use it with ethernet bridging but that isn't a hard
requirement. And yes the mac address is different because you can't
do layer 2 switching if everyone has the same mac address.

But there is no special ``virtual'' device.

> How to manage it with the real MAC address on the system ?
Manage?

> How to manage ARP, ICMP, multicasting and IP ?

Like you always do. It would be a terrible implementation if
we had to change that logic. There is a little bit of that
where we need to detect which network namespace we are going to because
the answers can differ but that is pretty straight forward.

> It seems for me, IMHO that will require a lot of translation and browsing
> table. It will probably add a very significant overhead.

Then look at:
 git://git.kernel.org/pub/scm/linux/kernel/git/ebiederm/linux -2.6-ns.git#proof-of-concept
or the OpenVZ implementation.

It isn't serious overhead.

>   * How to handle NFS access mounted outside of the container ?

The socket should remember it's network namespace.
It works fine.

>   * How to handle ICMP_REDIRECT ?

Just like we always do?

Eric

Subject: Re: [patch 2/6] [Network namespace] Network device sharing by view
Posted by ebiederm on Fri, 30 Jun 2006 18:22:24 GMT
View Forum Message <> Reply to Message

jamal <hadi@cyberus.ca> writes:

>> > > Then the pragmatic question becomes how to correlate what you see from
>> > > `ip addr list' to guests.
>> >
>> > on the host ip addr and the one seen on the guest side are the same.
>> > Except one is seen (on the host) on guest0-eth0 and another is seen
>> > on eth0 (on guest).
>>
>> Then what to do if the host system has 10.0.0.1 as a private address on eth3,
>> and then interfaces guest1-tun0 and guest2-tun0 both get address 10.0.0.1
>> when each guest has added 10.0.0.1 to their tun0 device?
>
> Yes, that would be a conflict that needs to be resolved. If you look at
> ip addresses as also belonging to namespaces, then it should work, no?
> i am assuming a tag at the ifa table level.

Yes. The conception is that everything belongs to the namespace,
so it looks like you have multiple instances of the network stack.

Which means through existing interfaces it would be a real problem
if a network device showed up in more than one network stack as
that would confuse things.

Basically the reading and configuration through existing interfaces

is expected to be in the namespace as well which is where the difficulty
shows up.

When you get serious about splitting up roots powers this becomes a real
advantage. Because you might want to have one person responsible for
what would normally be eth0 and another person responsible for eth1.

Anyway Jamal can you see the problem the aliases present to the implementation?

Eric

Subject: Re: [patch 2/6] [Network namespace] Network device sharing by view
Posted by jamal on Fri, 30 Jun 2006 21:51:58 GMT

View Forum Message <> Reply to Message

On Fri, 2006-30-06 at 12:22 -0600, Eric W. Biederman wrote:

>
> Anyway Jamal can you see the problem the aliases present to the implementation?
>

I think more than anything i may have a different view of things and no
code ;-> And you are trying to restore order in the discussion - so my
wild ideas dont help. If you guys have a meeting somewhere like this
coming OLS I will come over and disrupt your meeting ;-> I actually have
attempted to implement things like virtual routers but you guys seem way
ahead of anywhere i was.

cheers,
jamal

Subject: Re: [patch 2/6] [Network namespace] Network device sharing by view
Posted by ebiederm on Sat, 01 Jul 2006 00:50:20 GMT

View Forum Message <> Reply to Message

jamal <hadi@cyberus.ca> writes:

> On Fri, 2006-30-06 at 12:22 -0600, Eric W. Biederman wrote:
>
>>
>> Anyway Jamal can you see the problem the aliases present to the
> implementation?
>>
>

> I think more than anything i may have a different view of things and no
> code ;-> And you are trying to restore order in the discussion - so my
> wild ideas dont help. If you guys have a meeting somewhere like this
> coming OLS I will come over and disrupt your meeting ;-> I actually have
> attempted to implement things like virtual routers but you guys seem way
> ahead of anywhere i was.

Currently I think we have both a talk, and a BOFH at OLS plus probably
a little time at kernel summit.

Eric

Subject: Re: strict isolation of net interfaces
Posted by Herbert Poetzl on Mon, 03 Jul 2006 13:36:02 GMT
View Forum Message <> Reply to Message

On Fri, Jun 30, 2006 at 10:56:13AM +0200, Cedric Le Goater wrote:
> Serge E. Hallyn wrote:
> >
> > The last one in your diagram confuses me - why foo0:1? I would
> > have thought it'd be
>
> just thinking aloud. I thought that any kind/type of interface could be
> mapped from host to guest.
>

```
> > host                 |  guest 0  |  guest 1  |  guest2
> >  ----------------------+-----------+-----------+------------- -
> >   |                    |           |           |
> >   |-> l0      <-------+-> lo0 ... | lo0       | lo0
> >   |                    |           |           |
> >   |-> eth0             |           |           |
> >   |                    |           |           |
> >   |-> veth0  <--------+-> eth0    |           |
> >   |                    |           |           |
> >   |-> veth1  <--------+-----------+-----------+-> eth0
> >   |                    |           |           |
> >   |-> veth2   <-------+-----------+-> eth0    |
```

> >
> > I think we should avoid using device aliases, as trying to do
> > something like giving eth0:1 to guest1 and eth0:2 to guest2
> > while hiding eth0:1 from guest2 requires some uglier code (as
> > I recall) than working with full devices. In other words,
> > if a namespace can see eth0, and eth0:2 exists, it should always
> > see eth0:2.
> >
> > So conceptually using a full virtual net device per container
> > certainly seems cleaner to me, and it seems like it should be

```
> > simpler by way of statistics gathering etc, but are there actually
> > any real gains?  Or is the support for multiple IPs per device
> > actually enough?
> >
> > Herbert, is this basically how ngnet is supposed to work?
```

hard to tell, we have at least three ngnet prototypes
and basically all variants are covered there, from
separate interfaces which map to real ones to perfect
isolation of addresses assigned to global interfaces

IMHO the 'virtual' interface per guest is fine, as
the overhead and consumed resources are non critical
and it will definitely simplify handling for the
guest side

I'd really appreciate if we could find a solution which
allows both, isolation and virtualization, and if the
bridge scenario is as fast as a direct mapping, I'm
perfectly fine with a big bridge + ebtables to handle
security issues

best,
Herbert

Subject: Re: strict isolation of net interfaces
Posted by Andrey Savochkin on Mon, 03 Jul 2006 14:53:50 GMT
View Forum Message <> Reply to Message

Sam, Serge, Cedric,

On Fri, Jun 30, 2006 at 02:49:05PM +1200, Sam Vilain wrote:

```
> Serge E. Hallyn wrote:
> > The last one in your diagram confuses me - why foo0:1?  I would
> > have thought it'd be
> >
> > host                  |  guest 0  |  guest 1  |  guest2
> >  ----------------------+-----------+-----------+------------- -
> >   |                   |           |           |
> >   |-> l0      <-------+-> lo0 ... | lo0       | lo0
> >   |                   |           |           |
> >   |-> eth0            |           |           |
> >   |                   |           |           |
> >   |-> veth0  <--------+-> eth0    |           |
> >   |                   |           |           |
> >   |-> veth1  <--------+-----------+-----------+-> eth0
> >   |                   |           |           |
> >   |-> veth2   <-------+-----------+-> eth0    |
> >
> > [...]
> >
> > So conceptually using a full virtual net device per container
> > certainly seems cleaner to me, and it seems like it should be
> > simpler by way of statistics gathering etc, but are there actually
> > any real gains?  Or is the support for multiple IPs per device
> > actually enough?
> >
>
> Why special case loopback?
>
> Why not:
>
> host                  |  guest 0  |  guest 1  |  guest2
>  ----------------------+-----------+-----------+------------- -
>   |                   |           |           |
>   |-> lo              |           |           |
>   |                   |           |           |
>   |-> vlo0  <---------+-> lo      |           |
>   |                   |           |           |
>   |-> vlo1  <---------+-----------+-----------+-> lo
>   |                   |           |           |
>   |-> vlo2   <--------+-----------+-> lo      |
>   |                   |           |           |
>   |-> eth0            |           |           |
>   |                   |           |           |
>   |-> veth0  <--------+-> eth0    |           |
>   |                   |           |           |
>   |-> veth1  <--------+-----------+-----------+-> eth0
>   |                   |           |           |
>   |-> veth2   <-------+-----------+-> eth0    |
```

I still can't completely understand your direction of thoughts.
Could you elaborate on IP address assignment in your diagram, please? For example, guest0 wants 127.0.0.1 and 192.168.0.1 addresses on its lo interface, and 10.1.1.1 on its eth0 interface.
Does this diagram assume any local IP addresses on v* interfaces in the "host"?

And the second question.
Are vlo0, veth0, etc. devices supposed to have hard_xmit routines?

Best regards

Andrey

Subject: Re: strict isolation of net interfaces
Posted by Sam Vilain on Tue, 04 Jul 2006 03:00:05 GMT
View Forum Message <> Reply to Message

Andrey Savochkin wrote:

```
>> Why special case loopback?
>>
>> Why not:
>>
>> host                  | guest 0  | guest 1  | guest2
>>  ----------------------+-----------+-----------+------------- -
>>   |                   |          |          |
>>   |-> lo               |          |          |
>>   |                   |          |          |
>>   |-> vlo0  <---------+-> lo     |          |
>>   |                   |          |          |
>>   |-> vlo1  <---------+-----------+-----------+-> lo
>>   |                   |          |          |
>>   |-> vlo2   <--------+-----------+-> lo     |
>>   |                   |          |          |
>>   |-> eth0             |          |          |
>>   |                   |          |          |
>>   |-> veth0  <--------+-> eth0   |          |
>>   |                   |          |          |
>>   |-> veth1  <--------+-----------+-----------+-> eth0
>>   |                   |          |          |
>>   |-> veth2   <-------+-----------+-> eth0   |
>>
>
> I still can't completely understand your direction of thoughts.
> Could you elaborate on IP address assignment in your diagram, please?  For
> example, guest0 wants 127.0.0.1 and 192.168.0.1 addresses on its lo
> interface, and 10.1.1.1 on its eth0 interface.
> Does this diagram assume any local IP addresses on v* interfaces in the
> "host"?
>
```

Well, Eric already pointed out some pretty good reasons why this thread should die.

The idea is that each "lo" interface would have the same set of addresses. Which would make routing on the host confusing. Yet another reason to kill this idea. Let's just make better tools instead.

Sam.

> And the second question.
> Are vlo0, veth0, etc. devices supposed to have hard_xmit routines?
>

Subject: Re: strict isolation of net interfaces
Posted by Daniel Lezcano on Tue, 04 Jul 2006 12:29:28 GMT

View Forum Message <> Reply to Message

Andrey Savochkin wrote:
>
> I still can't completely understand your direction of thoughts.
> Could you elaborate on IP address assignment in your diagram, please? For
> example, guest0 wants 127.0.0.1 and 192.168.0.1 addresses on its lo
> interface, and 10.1.1.1 on its eth0 interface.
> Does this diagram assume any local IP addresses on v* interfaces in the
> "host"?
>
> And the second question.
> Are vlo0, veth0, etc. devices supposed to have hard_xmit routines?

Andrey,

some people are interested by a network full isolation/virtualization like you did with the layer 2 isolation and some other people are interested by a light network isolation done at the layer 3. This one is intended to implement "application container" aka "lightweight container".

In the case of a layer 3 isolation, the network interface is not totally isolated and the debate here is to find a way to have something intuitive to manage the network devices.

IHMO, all the discussion we had convinced me of the needs to have the possibility to choose between a layer 2 or a layer 3 isolation.

If it is ok for you, we can collaborate to merge the two solutions in one. I will focus on layer 3 isolation and you on the layer 2.

Regards

  - Daniel

Subject: Re: strict isolation of net interfaces
Posted by Sam Vilain on Tue, 04 Jul 2006 13:13:56 GMT

View Forum Message <> Reply to Message

Daniel Lezcano wrote:
>
> If it is ok for you, we can collaborate to merge the two solutions in
> one. I will focus on layer 3 isolation and you on the layer 2.

So, you're writing a LSM module or adapting the BSD Jail LSM, right? :)

Sam.

Subject: Re: strict isolation of net interfaces
Posted by Daniel Lezcano on Tue, 04 Jul 2006 13:19:52 GMT
View Forum Message <> Reply to Message

Sam Vilain wrote:
> Daniel Lezcano wrote:
>
>>If it is ok for you, we can collaborate to merge the two solutions in
>>one. I will focus on layer 3 isolation and you on the layer 2.
>
>
> So, you're writing a LSM module or adapting the BSD Jail LSM, right? :)
>
> Sam.

No. I am adapting a prototype of network application container we did.

-- Daniel