Subject: Re: [patch 2/6] [Network namespace] Network device sharing by view
Posted by ebiederm on Mon, 26 Jun 2006 16:40:59 GMT
View Forum Message <> Reply to Message

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

>> 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.

Is this an implementation difference or is this a user visible difference?
As an implementation difference this is sensible, as it is pretty insane
to allocate hash tables at run time.

As a user visible difference that affects semantics of the operations
this is not something we want to do.

Eric

Subject: Re: [patch 2/6] [Network namespace] Network device sharing by view
Posted by Herbert Poetzl on Mon, 26 Jun 2006 18:36:49 GMT
View Forum Message <> Reply to Message

On Mon, Jun 26, 2006 at 10:40:59AM -0600, Eric W. Biederman wrote:
> Daniel Lezcano <dlezcano@fr.ibm.com> writes:
>
> >> 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.
>
> Is this an implementation difference or is this a user visible
> difference? As an implementation difference this is sensible, as it is
> pretty insane to allocate hash tables at run time.
>
> As a user visible difference that affects semantics of the operations
> this is not something we want to do.

well, I guess there are even more options here, for
example I'd like to propose the following idea, which

might be a viable solution for the policy/isolation
problem, with the actual overhead on the setup part
not the hot pathes for packet and connection handling

we could use the multiple routing tables to provide
a single routing table for each guest, which could
be used inside the guest to add arbitrary routes, but
would allow to keep the 'main' policy on the host, by
selecting the proper table based on IPs and guest tags

similar we could allow to have a separate iptables
chain for each guest (or several chains), which are
once again directed by the host system (applying the
required prolicy) which can be managed and configured
via normal iptable interfaces (both on the guest and
host) but actually provide at least to layers

note: this does not work for hierarchical network
contexts, but I do not see that the yet proposed
implementations would do, so I do not think that
is of concern here ...

best,
Herbert

> Eric

Subject: Re: [patch 2/6] [Network namespace] Network device sharing by view
Posted by ebiederm on Mon, 26 Jun 2006 19:35:15 GMT
View Forum Message <> Reply to Message

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

> On Mon, Jun 26, 2006 at 10:40:59AM -0600, Eric W. Biederman wrote:
>> Daniel Lezcano <dlezcano@fr.ibm.com> writes:
>>
>> >> 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.
>>
>> Is this an implementation difference or is this a user visible
>> difference? As an implementation difference this is sensible, as it is

>> pretty insane to allocate hash tables at run time.
>>
>> As a user visible difference that affects semantics of the operations
>> this is not something we want to do.
>
> well, I guess there are even more options here, for
> example I'd like to propose the following idea, which
> might be a viable solution for the policy/isolation
> problem, with the actual overhead on the setup part
> not the hot pathes for packet and connection handling
>
> we could use the multiple routing tables to provide
> a single routing table for each guest, which could
> be used inside the guest to add arbitrary routes, but
> would allow to keep the 'main' policy on the host, by
> selecting the proper table based on IPs and guest tags
>
> similar we could allow to have a separate iptables
> chain for each guest (or several chains), which are
> once again directed by the host system (applying the
> required prolicy) which can be managed and configured
> via normal iptable interfaces (both on the guest and
> host) but actually provide at least to layers

I have real concerns about the complexity of the route you
have described.

> note: this does not work for hierarchical network
> contexts, but I do not see that the yet proposed
> implementations would do, so I do not think that
> is of concern here ...

Well we are hierarchical in the sense that a parent
can have a different network namespace then a child.
So recursive containers work fine. So this is like
the uts namespace or the ipc namespace rather than
like the pid namespace.

I really do not believe we have a hotpath issue, if this
is implemented properly. Benchmarks of course need to be taken,
to prove this.

There are only two places a sane implementation should show issues.
- When the access to a pointer goes through a pointer to find
  that global variable.
- When doing a lookup in a hash table we need to look at an additional
  field to verify a hash match. Because having a completely separate
  hash table is likely too expensive.

If that can be shown to really slow down packets on the hot path
I am willing to consider other possibilities. Until then I think
we are on path to the simplest and most powerful version of building
a network namespace usable by containers.

The routing between network namespaces does have the potential to
be more expensive than just a packet trivially coming off the wire
into a socket. However that is fundamentally from a lack of hardware.
If the rest works smarter filters in the drivers should enable to
remove the cost.

Basically it is just a matter of:
if (dest_mac == my_mac1) it is for device 1.
If (dest_mac == my_mac2) it is for device 2.
etc.

At a small count of macs it is trivial to understand it will go
fast for a larger count of macs it only works with a good data
structure. We don't hit any extra cache lines of the packet,
and the above test can be collapsed with other routing lookup tests.

Eric

Subject: Re: [patch 2/6] [Network namespace] Network device sharing by view
Posted by Herbert Poetzl on Mon, 26 Jun 2006 20:02:25 GMT
View Forum Message <> Reply to Message

On Mon, Jun 26, 2006 at 01:35:15PM -0600, Eric W. Biederman wrote:
> Herbert Poetzl <herbert@13thfloor.at> writes:
>
> > On Mon, Jun 26, 2006 at 10:40:59AM -0600, Eric W. Biederman wrote:
> >> Daniel Lezcano <dlezcano@fr.ibm.com> writes:
> >>
> >> >> 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.
> >>
> >> Is this an implementation difference or is this a user visible
> >> difference? As an implementation difference this is sensible, as it is
> >> pretty insane to allocate hash tables at run time.
> >>

> >> As a user visible difference that affects semantics of the operations
> >> this is not something we want to do.
> >
> > well, I guess there are even more options here, for
> > example I'd like to propose the following idea, which
> > might be a viable solution for the policy/isolation
> > problem, with the actual overhead on the setup part
> > not the hot pathes for packet and connection handling
> >
> > we could use the multiple routing tables to provide
> > a single routing table for each guest, which could
> > be used inside the guest to add arbitrary routes, but
> > would allow to keep the 'main' policy on the host, by
> > selecting the proper table based on IPs and guest tags
> >
> > similar we could allow to have a separate iptables
> > chain for each guest (or several chains), which are
> > once again directed by the host system (applying the
> > required prolicy) which can be managed and configured
> > via normal iptable interfaces (both on the guest and
> > host) but actually provide at least to layers
>
> I have real concerns about the complexity of the route you
> have described.
>
> > note: this does not work for hierarchical network
> > contexts, but I do not see that the yet proposed
> > implementations would do, so I do not think that
> > is of concern here ...
>
> Well we are hierarchical in the sense that a parent
> can have a different network namespace then a child.
> So recursive containers work fine. So this is like
> the uts namespace or the ipc namespace rather than
> like the pid namespace.

yes, but you will not be able to apply policy on
the parent, restricting the child networking in a
proper way without jumping through hoops ...

> I really do not believe we have a hotpath issue, if this
> is implemented properly. Benchmarks of course need to be taken,
> to prove this.

I'm fine with proper testing and good numbers here
but until then, I can only consider it a prototype

> There are only two places a sane implementation should show issues.

> - When the access to a pointer goes through a pointer to find
>   that global variable.
> - When doing a lookup in a hash table we need to look at an additional
>   field to verify a hash match. Because having a completely separate
>   hash table is likely too expensive.
>
> If that can be shown to really slow down packets on the hot path I am
> willing to consider other possibilities. Until then I think we are on
> path to the simplest and most powerful version of building a network
> namespace usable by containers.

keep in mind that you actually have three kinds
of network traffic on a typical host/guest system:

 - traffic between unit and outside
   - host traffic should be quite minimal
   - guest traffic will be quite high

 - traffic between host and guest
   probably minimal too (only for shared services)

 - traffic between guests
   can be as high (or even higher) than the
   outbound traffic, just think web guest and
   database guest

> The routing between network namespaces does have the potential to be
> more expensive than just a packet trivially coming off the wire into a
> socket.

IMHO the routing between network namespaces should
not require more than the current local traffic
does (i.e. you should be able to achieve loopback
speed within an insignificant tolerance) and not
nearly the time required for on-wire stuff ...

> However that is fundamentally from a lack of hardware. If the
> rest works smarter filters in the drivers should enable to remove the
> cost.
>
> Basically it is just a matter of:
> if (dest_mac == my_mac1) it is for device 1.
> If (dest_mac == my_mac2) it is for device 2.
> etc.

hmm, so you plan on providing a different MAC for
each guest? how should that be handled from the
user PoV? you cannot simply make up MACs as you

go, and, depending on the network card, operation
in promisc mode might be slower than for a given
set (maybe only one) MAC, no?

> At a small count of macs it is trivial to understand it will go
> fast for a larger count of macs it only works with a good data
> structure. We don't hit any extra cache lines of the packet,
> and the above test can be collapsed with other routing lookup tests.

well, I'm absolutely not against flexibility or
full virtualization, but the proposed 'routing'
on the host effectively doubles the time the
packet spends in the network stack(s), so I can
not believe that this approach would not add
(significant) overhead to the hot path ...

best,
Herbert

> Eric

Subject: Re: [patch 2/6] [Network namespace] Network device sharing by view
Posted by ebiederm on Mon, 26 Jun 2006 20:37:15 GMT
View Forum Message <> Reply to Message

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

> On Mon, Jun 26, 2006 at 01:35:15PM -0600, Eric W. Biederman wrote:
>> Herbert Poetzl <herbert@13thfloor.at> writes:
>>
>
> yes, but you will not be able to apply policy on
> the parent, restricting the child networking in a
> proper way without jumping through hoops ...

? I don't understand where you are coming from.
There is no restriction on where you can apply policy.

>> I really do not believe we have a hotpath issue, if this
>> is implemented properly. Benchmarks of course need to be taken,
>> to prove this.
>
> I'm fine with proper testing and good numbers here
> but until then, I can only consider it a prototype

We are taking the first steps to get this all sorted out.
I think what we have is more than a prototype but less then

the final implementation. Call it the very first draft version.

>> There are only two places a sane implementation should show issues.
>> - When the access to a pointer goes through a pointer to find
>>   that global variable.
>> - When doing a lookup in a hash table we need to look at an additional
>>   field to verify a hash match. Because having a completely separate
>>   hash table is likely too expensive.
>>
>> If that can be shown to really slow down packets on the hot path I am
>> willing to consider other possibilities. Until then I think we are on
>> path to the simplest and most powerful version of building a network
>> namespace usable by containers.
>
> keep in mind that you actually have three kinds
> of network traffic on a typical host/guest system:
>
> - traffic between unit and outside
>   - host traffic should be quite minimal
>   - guest traffic will be quite high
>
> - traffic between host and guest
>   probably minimal too (only for shared services)
>
> - traffic between guests
>   can be as high (or even higher) than the
>   outbound traffic, just think web guest and
>   database guest

Interesting.

>> The routing between network namespaces does have the potential to be
>> more expensive than just a packet trivially coming off the wire into a
>> socket.
>
> IMHO the routing between network namespaces should
> not require more than the current local traffic
> does (i.e. you should be able to achieve loopback
> speed within an insignificant tolerance) and not
> nearly the time required for on-wire stuff ...

That assumes on the wire stuff is noticeably slower.
You can achieve over 1GB/s on some networks.

But I agree that the cost should resemble the current
loopback device. I have seen nothing that suggests
it is not.

>> However that is fundamentally from a lack of hardware. If the
>> rest works smarter filters in the drivers should enable to remove the
>> cost.
>>
>> Basically it is just a matter of:
>> if (dest_mac == my_mac1) it is for device 1.
>> If (dest_mac == my_mac2) it is for device 2.
>> etc.
>
> hmm, so you plan on providing a different MAC for
> each guest? how should that be handled from the
> user PoV? you cannot simply make up MACs as you
> go, and, depending on the network card, operation
> in promisc mode might be slower than for a given
> set (maybe only one) MAC, no?

The speed is a factor certainly. As for making up
macs. There is a local assignment bit that you can set.
With that set it is just a matter of using a decent random
number generator. The kernel already does this is some places.

>> At a small count of macs it is trivial to understand it will go
>> fast for a larger count of macs it only works with a good data
>> structure. We don't hit any extra cache lines of the packet,
>> and the above test can be collapsed with other routing lookup tests.
>
> well, I'm absolutely not against flexibility or
> full virtualization, but the proposed 'routing'
> on the host effectively doubles the time the
> packet spends in the network stack(s), so I can
> not believe that this approach would not add
> (significant) overhead to the hot path ...

It might, but I am pretty certain it won't double
the cost, as you don't do 2 full network stack traversals.
And even at a full doubling I doubt it will affect bandwith
or latency very much. If it does we have a lot more to optimize
in the network stack than just this code.

Eric

Subject: Re: [patch 2/6] [Network namespace] Network device sharing by view
Posted by Herbert Poetzl on Mon, 26 Jun 2006 21:26:16 GMT
View Forum Message <> Reply to Message

On Mon, Jun 26, 2006 at 02:37:15PM -0600, Eric W. Biederman wrote:
> Herbert Poetzl <herbert@13thfloor.at> writes:

>
> > On Mon, Jun 26, 2006 at 01:35:15PM -0600, Eric W. Biederman wrote:
> >> Herbert Poetzl <herbert@13thfloor.at> writes:
> >
> > yes, but you will not be able to apply policy on
> > the parent, restricting the child networking in a
> > proper way without jumping through hoops ...
>
> ? I don't understand where you are coming from.
> There is no restriction on where you can apply policy.

in a fully hierarchical system (not that I really think
this is required here) you would be able to 'delegate'
certain addresses (ranges) to a namespace (the child)
below the current one (the parent) with the ability to
limit/control the input/output (which is required for
security)

> >> I really do not believe we have a hotpath issue, if this
> >> is implemented properly. Benchmarks of course need to be taken,
> >> to prove this.
> >
> > I'm fine with proper testing and good numbers here
> > but until then, I can only consider it a prototype
>
> We are taking the first steps to get this all sorted out.
> I think what we have is more than a prototype but less then
> the final implementation. Call it the very first draft version.

what we are desperately missing here is a proper way
to testing this, maybe the network folks can come up
with some test equipment/ideas here ...

> >> There are only two places a sane implementation should show issues.
> >> - When the access to a pointer goes through a pointer to find
> >>   that global variable.
> >> - When doing a lookup in a hash table we need to look at an additional
> >>   field to verify a hash match. Because having a completely separate
> >>   hash table is likely too expensive.
> >>
> >> If that can be shown to really slow down packets on the hot path I am
> >> willing to consider other possibilities. Until then I think we are on
> >> path to the simplest and most powerful version of building a network
> >> namespace usable by containers.
> >
> > keep in mind that you actually have three kinds
> > of network traffic on a typical host/guest system:
> >

> > - traffic between unit and outside
> >   - host traffic should be quite minimal
> >   - guest traffic will be quite high
> >
> > - traffic between host and guest
> >   probably minimal too (only for shared services)
> >
> > - traffic between guests
> >   can be as high (or even higher) than the
> >   outbound traffic, just think web guest and
> >   database guest
>
> Interesting.
>
> >> The routing between network namespaces does have the potential to be
> >> more expensive than just a packet trivially coming off the wire into a
> >> socket.
> >
> > IMHO the routing between network namespaces should
> > not require more than the current local traffic
> > does (i.e. you should be able to achieve loopback
> > speed within an insignificant tolerance) and not
> > nearly the time required for on-wire stuff ...
>
> That assumes on the wire stuff is noticeably slower.
> You can achieve over 1GB/s on some networks.

well, have you ever tried how much you can achieve
over loopback :)

> But I agree that the cost should resemble the current
> loopback device. I have seen nothing that suggests
> it is not.
>
> >> However that is fundamentally from a lack of hardware. If the
> >> rest works smarter filters in the drivers should enable to remove the
> >> cost.
> >>
> >> Basically it is just a matter of:
> >> if (dest_mac == my_mac1) it is for device 1.
> >> If (dest_mac == my_mac2) it is for device 2.
> >> etc.
> >
> > hmm, so you plan on providing a different MAC for
> > each guest? how should that be handled from the
> > user PoV? you cannot simply make up MACs as you
> > go, and, depending on the network card, operation
> > in promisc mode might be slower than for a given

> > set (maybe only one) MAC, no?
>
> The speed is a factor certainly. As for making up
> macs. There is a local assignment bit that you can set.

well, local is fine, but you cannot utilize that
on-wire which basically means that you would have
either to 'map' the MAC on transmission (to the
real one) which would basically make the approach
useless, or to use addresses which are fine within
a certain range of routers ...

> With that set it is just a matter of using a decent random
> number generator. The kernel already does this is some places.

sure you can make up MACs, but you will never
be able to use them 'outside' the box

> >> At a small count of macs it is trivial to understand it will go
> >> fast for a larger count of macs it only works with a good data
> >> structure. We don't hit any extra cache lines of the packet,
> >> and the above test can be collapsed with other routing lookup tests.
> >
> > well, I'm absolutely not against flexibility or
> > full virtualization, but the proposed 'routing'
> > on the host effectively doubles the time the
> > packet spends in the network stack(s), so I can
> > not believe that this approach would not add
> > (significant) overhead to the hot path ...
>
> It might, but I am pretty certain it won't double
> the cost, as you don't do 2 full network stack traversals.

> And even at a full doubling I doubt it will affect bandwith
> or latency very much.

well, for loopback that would mean half the bandwidth
and twice the latency, no?

> If it does we have a lot more to optimize in the network stack than
> just this code.

why? duplicate stack traversal takes roughly twice
the time, or am I wrong here? if so, please enlighten
me ...

best,
Herbert

> Eric

Subject: Re: [patch 2/6] [Network namespace] Network device sharing by view
Posted by Ben Greear on Mon, 26 Jun 2006 21:59:36 GMT
View Forum Message <> Reply to Message

Herbert Poetzl wrote:
> On Mon, Jun 26, 2006 at 02:37:15PM -0600, Eric W. Biederman wrote:
>
>>Herbert Poetzl <herbert@13thfloor.at> writes:
>>
>>
>>>On Mon, Jun 26, 2006 at 01:35:15PM -0600, Eric W. Biederman wrote:
>>>
>>>>Herbert Poetzl <herbert@13thfloor.at> writes:
>>>
>>>yes, but you will not be able to apply policy on
>>>the parent, restricting the child networking in a
>>>proper way without jumping through hoops ...
>>
>>? I don't understand where you are coming from.
>>There is no restriction on where you can apply policy.
>
>
> in a fully hierarchical system (not that I really think
> this is required here) you would be able to 'delegate'
> certain addresses (ranges) to a namespace (the child)
> below the current one (the parent) with the ability to
> limit/control the input/output (which is required for
> security)
>
>
>>>>I really do not believe we have a hotpath issue, if this
>>>>is implemented properly. Benchmarks of course need to be taken,
>>>>to prove this.
>>>
>>>I'm fine with proper testing and good numbers here
>>>but until then, I can only consider it a prototype
>>
>>We are taking the first steps to get this all sorted out.
>>I think what we have is more than a prototype but less then
>>the final implementation. Call it the very first draft version.
>
>
> what we are desperately missing here is a proper way
> to testing this, maybe the network folks can come up

> with some test equipment/ideas here ...
>
>
>>>>There are only two places a sane implementation should show issues.
>>>>- When the access to a pointer goes through a pointer to find
>>>> that global variable.
>>>>- When doing a lookup in a hash table we need to look at an additional
>>>> field to verify a hash match. Because having a completely separate
>>>> hash table is likely too expensive.
>>>>
>>>>If that can be shown to really slow down packets on the hot path I am
>>>>willing to consider other possibilities. Until then I think we are on
>>>>path to the simplest and most powerful version of building a network
>>>>namespace usable by containers.
>>>
>>>keep in mind that you actually have three kinds
>>>of network traffic on a typical host/guest system:
>>>
>>> - traffic between unit and outside
>>>   - host traffic should be quite minimal
>>>   - guest traffic will be quite high
>>>
>>> - traffic between host and guest
>>>   probably minimal too (only for shared services)
>>>
>>> - traffic between guests
>>>   can be as high (or even higher) than the
>>>   outbound traffic, just think web guest and
>>>   database guest
>>
>>Interesting.
>>
>>
>>>>The routing between network namespaces does have the potential to be
>>>>more expensive than just a packet trivially coming off the wire into a
>>>>socket.
>>>
>>>IMHO the routing between network namespaces should
>>>not require more than the current local traffic
>>>does (i.e. you should be able to achieve loopback
>>>speed within an insignificant tolerance) and not
>>>nearly the time required for on-wire stuff ...
>>
>>That assumes on the wire stuff is noticeably slower.
>>You can achieve over 1GB/s on some networks.
>
>
> well, have you ever tried how much you can achieve

> over loopback :)
>
>
>>But I agree that the cost should resemble the current
>>loopback device. I have seen nothing that suggests
>>it is not.
>>
>>
>>>>However that is fundamentally from a lack of hardware. If the
>>>>rest works smarter filters in the drivers should enable to remove the
>>>>cost.
>>>>
>>>>Basically it is just a matter of:
>>>>if (dest_mac == my_mac1) it is for device 1.
>>>>If (dest_mac == my_mac2) it is for device 2.
>>>>etc.
>>>
>>>hmm, so you plan on providing a different MAC for
>>>each guest? how should that be handled from the
>>>user PoV? you cannot simply make up MACs as you
>>>go, and, depending on the network card, operation
>>>in promisc mode might be slower than for a given
>>>set (maybe only one) MAC, no?

> well, local is fine, but you cannot utilize that
> on-wire which basically means that you would have
> either to 'map' the MAC on transmission (to the
> real one) which would basically make the approach
> useless, or to use addresses which are fine within
> a certain range of routers ...
>
>
>>With that set it is just a matter of using a decent random
>>number generator. The kernel already does this is some places.
>
>
> sure you can make up MACs, but you will never
> be able to use them 'outside' the box

I do it all the time with my mac-vlan patch and it works fine..so
long as you pay minimal attention. Just let the user specify the
MAC addr and they can manage the uniqueness as they wish....

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 ebiederm on Mon, 26 Jun 2006 22:11:02 GMT
View Forum Message <> Reply to Message

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

> On Mon, Jun 26, 2006 at 02:37:15PM -0600, Eric W. Biederman wrote:
>> Herbert Poetzl <herbert@13thfloor.at> writes:
>>
>> > On Mon, Jun 26, 2006 at 01:35:15PM -0600, Eric W. Biederman wrote:
>> >> Herbert Poetzl <herbert@13thfloor.at> writes:
>> >
>> > yes, but you will not be able to apply policy on
>> > the parent, restricting the child networking in a
>> > proper way without jumping through hoops ...
>>
>> ? I don't understand where you are coming from.
>> There is no restriction on where you can apply policy.
>
> in a fully hierarchical system (not that I really think
> this is required here) you would be able to 'delegate'
> certain addresses (ranges) to a namespace (the child)
> below the current one (the parent) with the ability to
> limit/control the input/output (which is required for
> security)

All of that is possible with the current design.
It is merely a matter of using the features the kernel
currently has.

The trick is know that a child namespace only has a loopback
by default, and has to have a network interface added from
the parent to be able to talk to anything.

>> >> I really do not believe we have a hotpath issue, if this
>> >> is implemented properly. Benchmarks of course need to be taken,
>> >> to prove this.
>> >
>> > I'm fine with proper testing and good numbers here
>> > but until then, I can only consider it a prototype
>>
>> We are taking the first steps to get this all sorted out.
>> I think what we have is more than a prototype but less then
>> the final implementation. Call it the very first draft version.
>
> what we are desperately missing here is a proper way
> to testing this, maybe the network folks can come up
> with some test equipment/ideas here ...

>

>> That assumes on the wire stuff is noticeably slower.
>> You can achieve over 1GB/s on some networks.
>
> well, have you ever tried how much you can achieve
> over loopback :)

Not recently.

> well, local is fine, but you cannot utilize that
> on-wire which basically means that you would have
> either to 'map' the MAC on transmission (to the
> real one) which would basically make the approach
> useless, or to use addresses which are fine within
> a certain range of routers ...

I believe on the wire is fine as well. Certainly I had
no problems when I tested it. I do agree that it increases
the chance for a mac address collision so should be handled
carefully. But we are talking a number with almost as many random
bits as a UUID.

As I recall the rule from the birthday paradox is something like: if
you have N possibilities you get a 50% chance of collision when
you have sqrt(N) items present. sqrt(2**40) = 2**20 ~= 1 Million.

So as long as you have a good random generator the odds of a collision
are quite small until you have used a million local mac addresses.

Now while I can see some small chance of that happening on a very
crowded local area network, using lots of logical servers the
kernel arp cache and switch mac cache limits would start causing
real problems long before you got that far. So you would need to
start routing at which point the practical problem goes away.

>
> well, for loopback that would mean half the bandwidth
> and twice the latency, no?

Not at all. The usual bottle neck is copying the data. The data
only gets put in a skb once and then pointers to the skb are
passed around. So you get copied in once and copied out once.
We are certainly not going to add an extra copy to it.

For practical purposes the fast path through the network stack
is a series of hash table looks.

That added to the fact that we don't make a full trip through
the network stack on both sides (unless someone is running tcp dump)
for example.

>> If it does we have a lot more to optimize in the network stack than
>> just this code.
>
> why? duplicate stack traversal takes roughly twice
> the time, or am I wrong here? if so, please enlighten
> me ...

The network stack is how we decide what goes where. Sending
and receiving packets should always have hardware as the bottleneck,
not software. So software should be a tiny percentage of the time
it takes to send or receive any packet.

With packets limited to 1.5K and below things aren't always that
clear cut but the ideal remains. But basically anything we can
do besides remove the copy in and the copy out of the network stack
we should do.

The copy in and the copy out are fundamentally hard to remove without
modifying page tables which because of tlb invalidates can be even
more expensive than a copy.

The best you can hope for on a loopback scenario is a single copy
from one user space buffer to another skipping the intermediate
kernel buffer. But that is tricky for an entirely different set
of reasons.

Eric

Subject: Re: [patch 2/6] [Network namespace] Network device sharing by view
Posted by Ben Greear on Mon, 26 Jun 2006 22:13:17 GMT
View Forum Message <> Reply to Message

Eric W. Biederman wrote:

> Basically it is just a matter of:
> if (dest_mac == my_mac1) it is for device 1.
> If (dest_mac == my_mac2) it is for device 2.
> etc.
>
> At a small count of macs it is trivial to understand it will go
> fast for a larger count of macs it only works with a good data
> structure. We don't hit any extra cache lines of the packet,
> and the above test can be collapsed with other routing lookup tests.

I think you should do this at the layer-2 level, well before you get
to routing. That will make the virtual mac-vlan work with arbitrary
protocols and appear very much like a regular ethernet interface. This
approach worked well with .1q vlans, and with my version of the mac-vlan
module.

Using the mac-vlan and source-based routing tables, I can give a unique
'interface' to each process and have each process able to bind to the
same IP port, for instance. Using source-based routing (by binding to a local
IP explicitly and adding a route table for that source IP), I can give unique
default routes to each interface as well. Since we cannot have more than 256
routing tables, this approach is currently limitted to around 250 virtual
interfaces, but that is still a substantial amount.

My mac-vlan patch, redirect-device patch, and other hackings are consolidated
in this patch:

http://www.candelatech.com/oss/candela_2.6.16.patch

Thanks,
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 Andrey Savochkin on Tue, 27 Jun 2006 09:09:11 GMT
View Forum Message <> Reply to Message

Herbert,

On Mon, Jun 26, 2006 at 10:02:25PM +0200, Herbert Poetzl wrote:
>
> keep in mind that you actually have three kinds
> of network traffic on a typical host/guest system:
>
> - traffic between unit and outside
>   - host traffic should be quite minimal
>   - guest traffic will be quite high
>
> - traffic between host and guest
>   probably minimal too (only for shared services)
>
> - traffic between guests
>   can be as high (or even higher) than the

> outbound traffic, just think web guest and
> database guest

My experience with host-guest systems tells me the opposite:
outside traffic is a way higher than traffic between guests.
People put web server and database in different guests not more frequent than they put them on separate physical server.
Unless people are building a really huge system when 1 server can't take the whole load, web and database live together and benefit from communications over UNIX sockets.

Guests are usually comprised of web-db pairs, and people place many such guests on a single computer.

>
> > The routing between network namespaces does have the potential to be
> > more expensive than just a packet trivially coming off the wire into a
> > socket.
>
> IMHO the routing between network namespaces should
> not require more than the current local traffic
> does (i.e. you should be able to achieve loopback
> speed within an insignificant tolerance) and not
> nearly the time required for on-wire stuff ...

I'd like to caution about over-optimizing communications between different network namespaces.
Many optimizations of local traffic (such as high MTU) don't look so appealing when you start to think about live migration of namespaces.

Regards
Andrey

Subject: Re: [patch 2/6] [Network namespace] Network device sharing by view
Posted by Herbert Poetzl on Tue, 27 Jun 2006 15:48:19 GMT
View Forum Message <> Reply to Message

On Tue, Jun 27, 2006 at 01:09:11PM +0400, Andrey Savochkin wrote:
> Herbert,
>
> On Mon, Jun 26, 2006 at 10:02:25PM +0200, Herbert Poetzl wrote:
> > >
> > > keep in mind that you actually have three kinds
> > > of network traffic on a typical host/guest system:
> > >
> > > - traffic between unit and outside
> > > - host traffic should be quite minimal

> > - guest traffic will be quite high
> >
> > - traffic between host and guest
> > probably minimal too (only for shared services)
> >
> > - traffic between guests
> > can be as high (or even higher) than the
> > outbound traffic, just think web guest and
> > database guest
>
> My experience with host-guest systems tells me the opposite: outside
> traffic is a way higher than traffic between guests. People put web
> server and database in different guests not more frequent than they
> put them on separate physical server. Unless people are building a
> really huge system when 1 server can't take the whole load, web and
> database live together and benefit from communications over UNIX
> sockets.

well, that's probably because you (or your company)
focuses on providers which simply (re)sell the entities
to their customers, in which case it would be more
expensive to put e.g. the database into a separate
guest. but let me state here that this is not the only
application for this technology

many folks use Linux-VServer for separating services
(e.g. mail, web, database, ...) and here a _lot_ of
traffic happens between guests (as it would on a normal
linux system or within a single guest in your case)

> Guests are usually comprised of web-db pairs, and people place many
> such guests on a single computer.

in case two guests cost more than one, yes, in case
two guests allow for better isolation and easier
maintainance without additional cost, no :)

> > > The routing between network namespaces does have the potential to
> > > be more expensive than just a packet trivially coming off the wire
> > > into a socket.
> >
> > IMHO the routing between network namespaces should
> > not require more than the current local traffic
> > does (i.e. you should be able to achieve loopback
> > speed within an insignificant tolerance) and not
> > nearly the time required for on-wire stuff ...
>
> I'd like to caution about over-optimizing communications between

> different network namespaces. Many optimizations of local traffic
> (such as high MTU) don't look so appealing when you start to think
> about live migration of namespaces.

I think the 'optimization' (or to be precise: desire
not to sacrifice local/loopback traffic for some use
case as you describe it) does not interfere with live
migration at all, we still will have 'local' and 'remote'
traffic, and personally I doubt that the live migration
is a feature for the masses ...

best,
Herbert

> Regards
> Andrey

Subject: Re: [patch 2/6] [Network namespace] Network device sharing by view
Posted by Andrey Savochkin on Tue, 27 Jun 2006 16:19:01 GMT
View Forum Message <> Reply to Message

Herbert,

On Tue, Jun 27, 2006 at 05:48:19PM +0200, Herbert Poetzl wrote:
> On Tue, Jun 27, 2006 at 01:09:11PM +0400, Andrey Savochkin wrote:
> >
> > On Mon, Jun 26, 2006 at 10:02:25PM +0200, Herbert Poetzl wrote:
> > >
> > > - traffic between guests
> > >   can be as high (or even higher) than the
> > >   outbound traffic, just think web guest and
> > >   database guest
> >
> > My experience with host-guest systems tells me the opposite: outside
> > traffic is a way higher than traffic between guests. People put web
> > server and database in different guests not more frequent than they
> > put them on separate physical server. Unless people are building a
> > really huge system when 1 server can't take the whole load, web and
> > database live together and benefit from communications over UNIX
> > sockets.
>
> well, that's probably because you (or your company)
> focuses on providers which simply (re)sell the entities
> to their customers, in which case it would be more
> expensive to put e.g. the database into a separate
> guest. but let me state here that this is not the only
> application for this technology

I'm just sharing my experience.
You have one experience, I have another, and your classification of traffic
importance is not the universal one.
My point was that we shouldn't overestimate the use of INET sockets vs. UNIX
ones in configurations where communications but not web/db operations play a
big role in overall performance.
And indeed I've talked with many different people, from universities to
large enterprises.

>
[snip]
> > I'd like to caution about over-optimizing communications between
> > different network namespaces. Many optimizations of local traffic
> > (such as high MTU) don't look so appealing when you start to think
> > about live migration of namespaces.
>
> I think the 'optimization' (or to be precise: desire
> not to sacrifice local/loopback traffic for some use
> case as you describe it) does not interfere with live
> migration at all, we still will have 'local' and 'remote'
> traffic, and personally I doubt that the live migration
> is a feature for the masses ...

Why not for the masses?

Andrey

Subject: Re: [patch 2/6] [Network namespace] Network device sharing by view
Posted by ebiederm on Tue, 27 Jun 2006 16:40:18 GMT
View Forum Message <> Reply to Message

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

> On Tue, Jun 27, 2006 at 01:09:11PM +0400, Andrey Savochkin wrote:
>>
>> I'd like to caution about over-optimizing communications between
>> different network namespaces. Many optimizations of local traffic
>> (such as high MTU) don't look so appealing when you start to think
>> about live migration of namespaces.
>
> I think the 'optimization' (or to be precise: desire
> not to sacrifice local/loopback traffic for some use
> case as you describe it) does not interfere with live
> migration at all, we still will have 'local' and 'remote'
> traffic, and personally I doubt that the live migration
> is a feature for the masses ...

Several things.
- The linux loopback device is not strongly optimized, it is a compatibility
  layer.
- Traffic between guests is an implementation detail.
  There is nothing fundamental in our semantics that says the traffic
  has to be slow for any workload (except for the limuts imposed by using
  actual on the wire protocols). The lo shares the same problem.

Worry about this case now when it has clearly been shown that there are several possible ways to optimize this and get back any lost local performance is optimizing way too early.

Criticize the per namespace performance and all you want. That is pretty much a merge blocker. Unless we do worse than a 1-5% penalty the communication across namespaces is really a non-issue.

Even with your large communications flows between guests 1-5% is nothing.

Eric
