Subject: Re: L2 network namespace benchmarking
Posted by ebiederm on Wed, 28 Mar 2007 02:04:37 GMT
View Forum Message <> Reply to Message

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

>
> 3. General observations
> ----------------------
>
> The objective to have no performances degrations, when the network
> namespace is off in the kernel, is reached in both solutions.
>
> When the network is used outside the container and the network
> namespace are compiled in, there is no performance degradations.
>
> Eric's patchset allows to move network devices between namespaces and
> this is clearly a good feature, missing in the Dmitry's patchset. This
> feature helps us to see that the network namespace code does not add
> overhead when using directly the physical network device into the
> container.

Assuming these results are not contradicted this says that the extra
dereference where we need it does not add measurable to the overhead
in the Linus network stack. Performance wise this should be good
enough to allow merging the code into the linux kernel, as it does
not measurably affect networking when we do not have multiple
containers in use.

Things are good enough that we can even consider not providing
an option to compile the support out.

> The loss of performances is very noticeable inside the container and
> seems to be directly related to the usage of the pair device and the
> specific network configuration needed for the container. When the
> packets are sent by the container, the mac address is for the pair
> device but the IP address is not owned by the host. That directly
> implies to have the host to act as a router and the packets to be
> forwarded. That adds a lot of overhead.

Well it adds measurable overhead.

> A hack has been made in the ip_forward function to avoid useless
> skb_cow when using the pair device/tunnel device and the overhead
> is reduced by the half.

To be fully satisfactory how we get the packets to the namespace
still appears to need work.

We have overhead in routing. That may simply be the cost of
performing routing or there may be some optimizations opportunities
there.

We have about the same overhead when performing bridging which I
actually find more surprising, as the bridging code should involve
less packet handling.

Ideally we can optimize the bridge code or something equivalent to
it so that we can take one look at the destination mac address and
know which network namespace we should be in. Potentially moving this
work to hardware when the hardware supports multiple queues.

If we can get the overhead out of the routing code that would be
tremendous. However I think it may be more realistic to get the
overhead out of the ethernet bridging code where we know we don't need
to modify the packet.

Eric

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

---

Subject: Re: L2 network namespace benchmarking
Posted by dev on Wed, 28 Mar 2007 07:49:26 GMT
View Forum Message <> Reply to Message

>>The loss of performances is very noticeable inside the container and
>>seems to be directly related to the usage of the pair device and the
>>specific network configuration needed for the container. When the
>>packets are sent by the container, the mac address is for the pair
>>device but the IP address is not owned by the host. That directly
>>implies to have the host to act as a router and the packets to be
>>forwarded. That adds a lot of overhead.
>
>
> Well it adds measurable overhead.
>
>
>>A hack has been made in the ip_forward function to avoid useless
>>skb_cow when using the pair device/tunnel device and the overhead
>>is reduced by the half.
>
>
> To be fully satisfactory how we get the packets to the namespace

> still appears to need work.
>
> We have overhead in routing. That may simply be the cost of
> performing routing or there may be some optimizations opportunities
> there.
>
> We have about the same overhead when performing bridging which I
> actually find more surprising, as the bridging code should involve
> less packet handling.
>
> Ideally we can optimize the bridge code or something equivalent to
> it so that we can take one look at the destination mac address and
> know which network namespace we should be in. Potentially moving this
> work to hardware when the hardware supports multiple queues.
yes, we can hack the bridge, so that packets coming out of eth devices
can go directly to the container and get out of veth devices from
inside the container.

> If we can get the overhead out of the routing code that would be
> tremendous. However I think it may be more realistic to get the
> overhead out of the ethernet bridging code where we know we don't need
> to modify the packet.
Why not optimize both? :)

Thanks,
Kirill

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

---

Subject: Re: L2 network namespace benchmarking
Posted by Daniel Lezcano on Wed, 28 Mar 2007 07:55:46 GMT
View Forum Message <> Reply to Message

Eric W. Biederman wrote:
> Daniel Lezcano <dlezcano@fr.ibm.com> writes:
>
>> 3. General observations
>> ----------------------
>>
>> The objective to have no performances degrations, when the network
>> namespace is off in the kernel, is reached in both solutions.
>>
>> When the network is used outside the container and the network
>> namespace are compiled in, there is no performance degradations.

>>
>> Eric's patchset allows to move network devices between namespaces and
>> this is clearly a good feature, missing in the Dmitry's patchset. This
>> feature helps us to see that the network namespace code does not add
>> overhead when using directly the physical network device into the
>> container.
>
> Assuming these results are not contradicted this says that the extra
> dereference where we need it does not add measurable to the overhead
> in the Linus network stack. Performance wise this should be good
> enough to allow merging the code into the linux kernel, as it does
> not measurably affect networking when we do not have multiple
> containers in use.

I have a few questions about merging code into the linux kernel.

* How do you plan to do that ?
* When do you expect to have the network namespace into mainline ?
* Are Dave Miller and Alexey Kuznetov aware of the network namespace ?
* Did they saw your patchset or ever know it exists ?
* Do you have any feedbacks from netdev about the network namespace ?

>
> Things are good enough that we can even consider not providing
> an option to compile the support out.
>
>> The loss of performances is very noticeable inside the container and
>> seems to be directly related to the usage of the pair device and the
>> specific network configuration needed for the container. When the
>> packets are sent by the container, the mac address is for the pair
>> device but the IP address is not owned by the host. That directly
>> implies to have the host to act as a router and the packets to be
>> forwarded. That adds a lot of overhead.
>
> Well it adds measurable overhead.
>
>> A hack has been made in the ip_forward function to avoid useless
>> skb_cow when using the pair device/tunnel device and the overhead
>> is reduced by the half.
>
> To be fully satisfactory how we get the packets to the namespace
> still appears to need work.
>
> We have overhead in routing. That may simply be the cost of
> performing routing or there may be some optimizations opportunities
> there.
> We have about the same overhead when performing bridging which I
> actually find more surprising, as the bridging code should involve

> less packet handling.

Yep. I will try to figure out what is happening.

> Ideally we can optimize the bridge code or something equivalent to
> it so that we can take one look at the destination mac address and
> know which network namespace we should be in. Potentially moving this
> work to hardware when the hardware supports multiple queues.
>
> If we can get the overhead out of the routing code that would be
> tremendous. However I think it may be more realistic to get the
> overhead out of the ethernet bridging code where we know we don't need
> to modify the packet.

The routing was optimized for the loopback, no ? Why can't we do the
same for the etun device ?

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