
Subject: Re: Network virtualization/isolation

Posted by [Herbert Poetzl](#) on Sun, 26 Nov 2006 18:34:02 GMT

[View Forum Message](#) <> [Reply to Message](#)

On Sat, Nov 25, 2006 at 01:21:39AM -0700, Eric W. Biederman wrote:

>

> jamal <hadi@cyberus.ca> writes:

>

> > On Fri, 2006-27-10 at 11:10 +0200, Daniel Lezcano wrote:

> >

> >> No, it uses virtualization at layer 2 and I had already mention it

> >> before (see the first email of the thread), but thank you for the email

> >> thread pointer.

> >

> >

> > What would be really useful is someone takes the time and creates a

> > matrix of the differences between the implementations.

> > It seems there are quite a few differences but without such comparison

> > (to which all agree to) it is hard to form an opinion without a document

> > of some form.

> >

> > For one, I am puzzled by the arguments about L2 vs L3 - Is this the

> > host side or inside the VE?

> >

> > If it is a discussion of the host side:

> > To me it seems it involves the classification of some packet header

> > arriving on a physical netdevice on the host side (irrelevant whether

> > they are L2 or L7) and reaching a decision to select some redirected to

> > virtual netdevice.

>

> There are two techniques in real use.

> - Bind/Accept filtering

>

> Which layer 3 addresses a socket can bind/accept are filtered,

> but otherwise the network stack remains unchanged. When your

> container/VE only has a single IP address this works great.

> When you get multiple IPs this technique starts to fall down because

> it is not obvious how to make this correctly handle wild card ip

> addresses.

not really, you have to check for a (sub)set of IPs

that's quite simple and not hard to get right, I agree

that it increases the overhead on those checks, but

this only occurs at bind/connect/accept time ...

> This technique also falls down because it is very hard to

> support raw IP packets.

raw ip packets could be header checked in the same way
but in general, this type of isolation does not make
too much sense for raw ip purposes ...

- > The major advantage of this approach is that it is insanely simple
- > and cheap.

and don't forget flexible, because it also allows a few
things not quite easily done with layer2 techniques
(see below)

- > When the discussion started this is what I called Layer 3, bind
- > filtering is probably a more precise name.
- >
- > - Network object tagging
- >
- > Every network device, each socket, each routing table, each net
- > filter table, everything but the packets themselves is associated
- > with a single VE/container. In principle the network stack doesn't
- > change except everything that currently access global variables gets
- > an additional pointer indirection.
- >
- > To find where a packet is you must look at it's network device on
- > ingress, and you must look at it's socket on egress.
- >
- > This allows capabilities like CAP_NET_ADMIN to be fairly safely
- > given to people inside a container without problems.
- >
- > There are two basic concerns here.
- > 1) This is a lot of code that needs to be touched.
- > 2) There are not enough physical network devices to go around so
- > we need something that maps packets coming in a physical network
- > device into multiple virtual network devices.
- >
- > The obvious way to do this mapping is with either ethernet
- > bridging or with the linux routing code if the external network
- > is not ethernet, and some tunnel devices between the VE and the
- > host environment. This allows firewalling and in general the
- > full power of the linux network stack.
- >
- > The concern is that the extra trip through the network stack adds
- > overhead.
- >
- > This is the technique we have been calling layer two because it
- > works below the IP layer and as such should work for everything.
- >
- > There have been some intermediate solutions considered but generally

> the have the down sides of additional expense without the upsides of
 > more power.
 >
 > > The admin (on the host) decides what packets any VE can see.
 > > Once within the VE, standard Linux net stack applies. The same applies
 > > on the egress. The admin decides what packets emanating from the VE
 > > go where.
 > > I dont think this is a simple L2 vs L3. You need to be able to process
 > > IP as well as Decnet[1]
 >
 > Except for some implementation details I don't think we have
 > disagreement about what we are talking about although there is
 > certainly a little confusion. The true issue is can we implement
 > something that places the full power of the network stack (including
 > things like creating virtual tunnel devices) into the container/VE
 > without sacrificing performance.
 >
 > I think we could all agree on the most general technique if we could
 > convince ourselves the overhead was unmeasurable.

and we do not give away features like:

- sharing an IP between different guests, so that
 services can bind to the same IP as long as they
 do not collide
- allow simple wrapping (ala chroot()) for processes
 without requiring a complete routing/network setup
 plus a virtual switch/router/whatever

> Given that performance is the primary concern this is something a
 > network stack expert might be able to help with. My gut feel is
 > the extra pointer indirection for the more general technique is
 > negligible and will not affect the network performance. The network
 > stack can be very sensitive to additional cache misses so I could be
 > wrong. Opinions?

well, here we are talking about layer2 _isolation_
 if I got that right, i.e. you split the physical
 interfaces up into separate network namespaces, which
 then can make full use of the assigned interfaces

this is something which I'm perfectly fine with, as
 I do not think it adds significant overhead (nevertheless
 it needs some testing) but at the same time, this is
 something which isn't very useful in the generic case,
 where folks will have, let's say two network interfaces
 and want to share one of them between 100 guests ...

> Then the question is how do we reduce the overhead when we don't have
> enough physical network interfaces to go around. My feeling is that
> we could push the work to the network adapters and allow single
> physical network adapters to support multiple network interfaces, each
> with a different link-layer address.

that would be something interesting, but again, the
number of nics allowing for an arbitrary number of
filters, which also can be identified/correlated to
the network context without adding even more overhead
is probably insignificant ... so IMHO that would:

- keep all interfaces in promisc mode
- check each packet for the set of MACs

as the checks would require to identify the interface,
that would immediately result in $O(N)$ overhead for
each packet received, plus the overhead added by
disabling the hardware filters ... but maybe that
changed over the years, I'm definitely no network
stack/device expert ...

> At which point the overhead is nearly nothing and newer network
> adapters may start implementing enough filtering in hardware to do all
> of the work for us.

well, might be a solution in 4-5 years, when basically
any computer system uses such nics ...

> > [1] Since Linux has the only SMP-capable, firewall-capable Decnet
> > implementation - wouldnt it be fun to have it be virtualized as
> > well? ;->

>

> Yes. The only problem I have seen with Decnet is the pain of teaching
> it that it's variables aren't global anymore. Not a show stopper
> but something that keeps Decnet out of existing proof of concept
> implications.

>

> Eric

>

> p.s. Sorry for the long delayed reply. I have had my head down
> stabilizing 2.6.19 and netdev is not a list I regularly watch.

>

> p.p.s. I have CC'd the containers list so we catch all of the
> relevant people on the discussion. I believe this is an open
> list so shouldn't cause any problems.

thanks a lot!

best,
Herbert

> _____
> Containers mailing list
> Containers@lists.osdl.org
> <https://lists.osdl.org/mailman/listinfo/containers>

Containers mailing list
Containers@lists.osdl.org
<https://lists.osdl.org/mailman/listinfo/containers>
