Subject: Re: [patch 2/6] [Network namespace] Network device sharing by view
Posted by ebiederm on Wed, 28 Jun 2006 16:10:57 GMT
View Forum Message <> Reply to Message

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

>> Have a few more network interfaces for a layer 2 solution
>> is fundamental. Believing without proof and after arguments
>> to the contrary that you have not contradicted that a layer 2
>> solution is inherently slower is non-productive.
>
> assuming that it will not be slower, although it
> will now pass two network stacks and the bridging
> code is non-productive too, let's see how it goes
> but do not ignore the overhead just because it
> might simplify the implementation ...

Sure. Mostly I have set it aside because the overhead
is not horrible and it is a very specific case that
can be heavily optimized if the core infrastructure is
solid.

>> Arguing that a layer 2 only solution most prove itself on
>> guest to guest communication is also non-productive.
>>
>> So just to sink one additional nail in the coffin of the silly
>> guest to guest communication issue. For any two guests where
>> fast communication between them is really important I can run
>> an additional interface pair that requires no routing or bridging.
>> Given that the implementation of the tunnel device is essentially
>> the same as the loopback interface and that I make only one
>> trip through the network stack there will be no performance overhead.
>
> that is a good argument and I think I'm perfectly
> fine with this, given that the implementation
> allows that (i.e. the network stack can handle
> two interfaces with the same IP assigned and will
> choose the local interface over the remote one
> when the traffic will be between guests)

Yep. That exists today. The network stack prefers routes
as specific as possible.

>> Similarly for any critical guest communication to the outside world
>> I can give the guest a real network adapter.
>
> with a single MAC assigned, that is, I presume?

Yes.

>
> guess that's what this discussion is about,
> finding out the various aspects how isolation
> and/or vitrtualization can be accomplished and
> what features we consider common/useful enough
> for mainline ... for me that is still in the
> brainstorming phase, although several 'working
> prototypes' already exist. IMHO the next step
> is to collect a set of representative use cases
> and test them with each implementation, regarding
> performance, usability and practicability

I am fairly strongly convinced a layer 2 solution will
do fine. So for me it is a matter of proving that
and ensuring a good implementation.

> not necessarily, but I _know_ that the overhead
> added at layer 3 is unmeasureable, and it still
> needs to be proven that this is true for a layer
> 2 solution (which I'd actually prefer, because
> it solves the protocol _and_ setup issues)

That is a good perspective. Layer 3 is free, is layer 2 also free?
Unless the cache miss penalty is a killer layer 2 should come very
close. Of course VJ recently gave some evidence that packet processing
is dominated by cache misses.

>> >From what I have seen of layer 3 solutions it is a
>> bloody maintenance nightmare, and an inflexible mess.
>
> that is your opinion, I really doubt that you
> will have less maintenance when you apply policy
> to the guests ...

Yes and mostly of the layer 3 things that I implemented.
At a moderately fundamental level I see layer 3 implementations
being a special case that is a tangent from the rest of the
networking code. So I don't see a real synthesis with what
the rest of the networking stack is doing. Plus all of the
limitations that come with a layer 3 implementation.

> example here (just to clarify):
>
> - let's assume we have eth0 on the host and in
>   guest A and B, with the following setup:
>

> eth0(H) 192.168.0.1/24
> eth0(A) 10.0.1.1/16 10.0.1.2/16
> eth0(B) 10.0.2.1/16
>
> - now what keeps guest B from jsut assigning
> 10.0.2.2/16 to eth0? you need some kind of
> mechanism to prevent that, and/or to block
> the packets using inappropriate IPs
>
> * in the first case, i.e. you prevent assigning
> certain IPs inside a guest, you get a semantic
> change in the behaviour compared to a normal
> system, but there is no additional overhead
> on the communication
>
> * in the second case, you have to maintain the
> policy mechanism and keep it in sync with the
> guest configuration (somehow), and of course
> you have to verify every communication
>
> - OTOH, if you do not care about collisions
> basically assuming the point "that's like
> a hub on a network, if there are two guests
> with the same ip, it will be trouble, but
> that's okay" then this becomes a real issue
> for providers with potentially 'evil' customers

So linux when serving as a router has strong filter capabilities.

So we can either use the strong network filtering linux already has
making work for the host administrator who has poorly behaved customers.
Or we can simply not give those poorly behaved guests CAP_NET_ADMIN,
and assign the IP address at guest startup before dropping the
capability. At which point the guest cannot misbehave.

Eric
