

---

Subject: Re: [patch 2/6] [Network namespace] Network device sharing by view  
Posted by [Herbert Poetzl](#) on Mon, 26 Jun 2006 18:28:48 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

On Mon, Jun 26, 2006 at 06:08:03PM +0400, Andrey Savochkin wrote:

> Hi Herbert,

>

> On Mon, Jun 26, 2006 at 03:02:03PM +0200, Herbert Poetzl wrote:

> > On Mon, Jun 26, 2006 at 01:47:11PM +0400, Andrey Savochkin wrote:

> >

> > > I see a fundamental problem with this approach. When a device

> > > presents an skb to the protocol layer, it needs to know to which

> > > namespace this skb belongs.

> >

> > > Otherwise you would never get rid of problems with bind: what to

> > > do if device eth1 is visible in namespace1, namespace2, and root

> > > namespace, and each namespace has a socket bound to 0.0.0.0:80?

> >

> > this is something which isn't a fundamental problem at

> > all, and IMHO there are at least three options here

> > (probably more)

> >

> > - check at 'bind' time if the binding would overlap

> > and give the 'proper' error (as it happens right

> > now on the host)

> > (this is how Linux-VServer currently handles the

> > network isolation, and yes, it works quite fine :)

>

> I'm not comfortable with this as a permanent mainstream solution.

> It means that network namespaces are actually not namespaces: you

> can't run some program (e.g., apache) with default configs in a new

> namespace without regards to who runs what in other namespaces.

not at all, maybe you should take a closer look at the  
current Linux-VServer implementation, which is quite  
simple and does allow guests to bind to IP\_ANY quite  
fine, only the host (which has all privileges) has to  
be careful with binding to 0.0.0.0 ...

> In other words, name "0.0.0.0:80" creates a collision in your  
> implementation, so socket "names" do not form isolated spaces.

>

> >

> > - allow arbitrary binds and 'tag' the packets according

> > to some 'host' policy (e.g. iptables or tc)

> > (this is how the Linux-VServer ngnet was designed)

> >

> > - deliver packets to all bound sockets/destinations

> > (this is probably a more unusable but quite thinkable  
> > solution)  
>  
> Deliver TCP packets to all sockets?  
> How many connections do you expect to be established in this case?

well, roughly the same number of connections you'll  
get when you have two boxes with the same IP on the  
same subnet :)

in other words, if there are more than one guest  
with the same ip and port open, then we have some  
kind of misconfiguration (i.e. policy is required)

> > > We have to conclude that each device should be visible only in one  
> > > namespace.  
> >  
> > I disagree here, especially some supervisor context or  
> > the host context should be able to 'see' and probably  
> > manipulate \_all\_ of the devices  
>  
> Right, manipulating all devices from some supervisor context is useful.  
>  
> But this shouldn't necessarily be done by regular ip/ifconfig tools.  
> Besides, it could be quite confusing if in ifconfig output in the  
> supervisor context you see 325 "tun0" devices coming from  
> different namespaces :)

isolation would not provide more than \_one\_ tun0  
interfaces, virtualization OTOH will ...

> So I'm all for network namespace management mechanisms not bound  
> to existing tools/APIs.

well, I'm not against new APIs/tools, but I prefer  
to keep it simple, and elegant, which often includes  
reusing existing APIs and tools ...

> > > Complete isolation will allow each namespace to set up own tun/tap  
> > > devices, have own routes, netfilter tables, and so on.  
> >  
> > tun/tap devices are quite possible with this approach  
> > too, I see no problem here ...  
> >  
> > for iptables and routes, I'm worried about the required  
> > 'policy' to make them secure, i.e. how do you ensure  
> > that the packets 'leaving' guest X do not contain  
> > 'evil' packets and/or disrupt your host system?

>  
> Sorry, I don't get your point.  
> How do you ensure that packets leaving your neighbor's computer  
> do not disrupt your system?

by having a strong 'policy' on the router/switch  
which will (hopefully) reject everything sent in error  
or to disrupt/harm other boxes ...

> > From my point of view, network namespaces are just neighbors.

yes, but you need policy there the same way you need  
it for resources, i.e. you cannot simply allow everyone  
to do everything with his network interface, especially  
if that interface is shared with all others ...

> > > My follow-up messages will contain the first set of patches with  
> > > network namespaces implemented in the same way as network isolation  
> > > in OpenVZ.  
> >  
> > hmm, you probably mean 'network virtualization' here  
>  
> I meant isolation between different network contexts/namespaces.

well, isolation is basically what we do in Linux-VServer  
by allowing to bind to certain IPs (or ranges) instead  
of binding all available IPs ... this can be extended  
for routing and iptables as well, and does not require  
any 'virtualization' which would give each guest it's own  
set of interfaces, routes, iptables etc ... and it is  
usually more lightweight too ..

> > > This patchset introduces namespaces for device list and IPv4  
> > > FIB/routing. Two technical issues are omitted to make the patch  
> > > idea clearer: device moving between namespaces, and selective  
> > > routing cache flush + garbage collection.  
> > >  
> > > If this patchset is agreeable, the next patchset will finalize  
> > > integration with nsproxy, add namespaces to socket lookup code and  
> > > neighbour cache, and introduce a simple device to pass traffic  
> > > between namespaces.  
> >  
> > passing traffic 'between' namespaces should happen via  
> > lo, no? what kind of 'device' is required there, and  
> > what overhead does it add to the networking?  
>  
> OpenVZ provides 2 options.

>  
> 1) A packet appears right inside some namespace, without any additional  
> overhead. Usually this implies that either all packets from this  
> device belong to this namespace, i.e. simple device->namespace  
> assignment. However, there is nothing conceptually wrong with  
> having namespace-aware device drivers or netfilter modules  
> selecting namespaces for each incoming packet. It all depends on  
> how you want packets go through various network layers, and how  
> much network management abilities you want to have in non-root  
> namespaces. My point is that for network namespaces being real  
> namespaces, decision making should be done somewhere before socket  
> lookup.

well, I doubt that many providers will be able to put  
roughly hundred or more network interface cards into  
their machines, plus a proper switch to do the policy :)

> 2) Parent network namespace acts as a router forwarding packets to child  
> namespaces. This scheme is the preferred one in OpenVZ for various  
> reasons, most important being the simplicity of migration of network  
> namespaces. In this case flexibility has the cost of going through  
> packet handling layers two times.  
  
> Technically, this is implemented via a simple netdevice doing  
> netif\_rx in hard\_xmit.

which results in a duplicate stack traversal and kernel  
side policy to decide which goes where ... i.e. at least  
twice as much overhead than any isolation would have

best,  
Herbert

> Regards  
>  
> Andrey

---