Subject: Re: [patch 2/6] [Network namespace] Network device sharing by view
Posted by Andrey Savochkin on Mon, 26 Jun 2006 14:08:03 GMT
View Forum Message <> Reply to Message

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.
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?

>
> > 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 :)

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

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

>
> > 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.

>
> > 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.

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.

Regards

Andrey
