Subject: Re: [patch 2/6] [Network namespace] Network device sharing by view
Posted by ebiederm on Mon, 26 Jun 2006 14:05:24 GMT
View Forum Message <> Reply to Message

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

> On Mon, Jun 26, 2006 at 01:47:11PM +0400, Andrey Savochkin wrote:
>> Hi Daniel,
>>
>> It's good that you kicked off network namespace discussion Although I.
>> wish you'd Cc'ed someone at OpenVZ so I could notice it earlier :) .
>
>> Indeed, the first point to agree in this discussion is device list.
>> In your patch, you essentially introduce a data structure parallel
>> to the main device list, creating a "view" of this list.
>
>> 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)

I agree that there are other implementations that can be used for
containers. However when you think namespaces this is what you need.

For several reasons.
1) So you can use AF_PACKET safely.
   This allows a network namespace to use DHCP and all of the other
   usual network autoconfiguration tools. 0.0.0.0:80 is just
   a special subset of that.

2) It means the existing network stack can be used without
   logic changes. All that is needed is a lookup of the appropriate
   context. This is very straight forward to audit.

3) Since all of the network stack is trivially available all of
   the advanced network stack features like iptables are easily
   available.

4) There is no retraining or other rules for user to learn.
   Because people understand what is going on it is more likely
   a setup will be secure. Most of the other implementations

don't quite act like a normal network setup and the special
rules can be hard to learn.

> - 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 :)

It works yes but it limits you to a subset of the network
stack. And has serious problems with concepts like INADDR_ANY.
PF_PACKET is not an option.

> - 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)

A little more general but very weird.

> - deliver packets to _all_ bound sockets/destinations
> (this is probably a more unusable but quite thinkable
> solution)
>
>> 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

This part really is necessary. This does not preclude managing
a network namespace from outside of the namespace.

>> In this case, instead of introducing net_ns_dev and net_ns_dev_list
>> structures, we can simply have a separate dev_base list head in each
>> namespace. Moreover, separate device list in each namespace will be in
>> line with making namespace isolation complete.
>
>> 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?

In the traditional ways. When you control the router and/or the switch
someone is directly connected to.

We don't need to reinvent the wheel if we do this properly.

>> 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?

Definitely not. lo is a local loopback interface.

What is needed is a two headed device that is the cousin of lo.
But with one network interface in each network namespace.

Note even connecting network namespaces is optional.

Eric
