Subject: Re: [PATCH 1/5] net: Modify all rtnetlink methods to only work in the initial namespace
Posted by ebiederm on Thu, 11 Oct 2007 07:57:11 GMT
View Forum Message <> Reply to Message

"Denis V. Lunev" <den@sw.ru> writes:

>> This patchset does need to get rebased on top of net-2.6.25 when it
>> opens and hopefully your patchset to remove the unnecessary work in
>> rtnl_unlock, and to really process netlink requests in process
>> context. I see a need for the more fundamental change you seem to
>> be advocating.

Grr. That last sentence should have been I do not see a need for the more fundamental change you seem to be advocating.

> I see that current patchset of RTNL code is attached. I'll start the
> next piece of work next week after some reaction from people.

> I understand your position. But still have some points :)
>
> First. A real-life usecase we have recently fixed in the OpenVZ. I have
> described it in the previous post, but (may be) not in great details.
>
> UDP buffers management for outgoing traffic.
> The packet carries over a destructor, which is called in skb_orphan in
> the real networking device. This destructor allows to queue more
> packets. There is no problem in the current implementation. But there is
> one after namespace introduction in the following configuration:
> [NS1] <-> [NS2] <-> [world]
> There are two namespaces on the host. One of them is connected to the
> outside world via another and the packet follows usual forwarding path
> in NS2.

Yes. We can't quite do that today because of the missing ipv4 support, but the basic infrastructure exists to describe this is already merged.

> The problem: if we call skb_orphan in [NS1] outgoing device, there is a
> great possibility to have a really huge packet drop in [NS2] on queuing
> operation.

Yes. Removing skb_orphan from veth to solve this looks like a pretty substantial hack. A clean solution is more likely to resemble ethernet pause frames so we temporarily plug the virtual device. Although there are issues with that as currently virtual devices don't have queues.

> In OpenVZ we have added skb_orphan into receive path (tcp_v4_rcv,
> __udp4_lib_rcv etc) and removed one from our virtual devices. As
> unfortunate side effect we have packet in NS2 with a socket from NS1.

That also does some really nasty things to accounting. Using
the macvlan devices solve all of this rather neatly and with
higher performance.

> So, we should have an architectural solution for this from a beginning.
> It will be too late to hack around and change namespace lookup scheme.

However what you are specifically concerned about seems to be
using skb->sk->sk_net. Currently the only place I use this
is in the rtnetlink code because the functions that process packets
are not also passed a socket. Those functions we could easily
pass in an explicit namespace or a socket parameter, and so those
functions would not care.

> I see that we should adopt a generic approach:
> - each concrete packet belongs to a concrete namespace
> - if function has a packet as a parameter, we should get namespace from
> packet

This approach when suggested in earlier review was pretty
substantially shot down. Shrinking the size of struct sk_buff is
currently an ongoing todo for the linux networking stack.

> - if skb->dev is present we should get namespace as skb->dev->nd_net
> - otherwise we should get skb->sk->sk_net. If skb->sk is NULL -> this is
> a bug

Currently killing skb->dev is a todo list item, so using it more is
probably the wrong approach.

I do agree the duplication information between fields on the skb
and fields passed to functions is a pain. Moving more onto the
skb seems contrary to the how the rest of the networking stack would
like to go, so it is likely better to simply remove skb->dev and pass
an explicit dev parameter instead. Please note that parameters passed
explicitly will live in registers and will be quite fast.

> So, for the case of all netfilter calls we'll have a pskb->dev->nd_net
> defined correctly.

That would certainly be a nice extra, but that really seems the wrong
way to go. I would rather add an explicit struct net *net parameter
to nf_hookfn.

Eric