Subject: Re: [PATCH] Virtual ethernet tunnel (v.2)
Posted by Ben Greear on Thu, 07 Jun 2007 15:23:09 GMT
View Forum Message <> Reply to Message

Pavel Emelianov wrote:
> Veth stands for Virtual ETHernet. It is a simple tunnel driver
> that works at the link layer and looks like a pair of ethernet
> devices interconnected with each other.
>
As Dave mentioned, there is already a driver known as 'veth'. Maybe borrow
the etun name as well?

I would also like some way to identify veth from other device types,
preferably
something like a value in sysfs. However, that should not hold up
consideration of
this patch, and I am willing to submit a patch after this goes in to add
the functionality
I want...

> +/*
> + * xmit
> + */
> +
> +static int veth_xmit(struct sk_buff *skb, struct net_device *dev)
> +{
> + struct net_device *rcv = NULL;
> + struct veth_device_stats *stats;
> + struct veth_priv *priv, *rcv_priv;
> + int length, cpu;
> +
> + skb_orphan(skb);
> +
> + priv = netdev_priv(dev);
> + cpu = smp_processor_id();
> + stats = per_cpu_ptr(priv->stats, cpu);
> + rcv = priv->peer;
> +
> + if (!(rcv->flags & IFF_UP))
> + goto outf;
>
I think you need at least the option to zero out the time-stamp,
otherwise it will
not be re-calculated when received on the peer, and it potentially spent
significant
time since it was last calculated (think netem delay or similar).

+ /* Zero out the time-stamp so that receiving code is forced

```
+         * to recalculate it.
+         */
+        skb->tstamp.off_sec = 0;
+        skb->tstamp.off_usec = 0;

> +
> + rcv_priv = netdev_priv(rcv);
> + skb->pkt_type = PACKET_HOST;
> + skb->protocol = eth_type_trans(skb, rcv);
> + if (dev->features & NETIF_F_NO_CSUM)
> +  skb->ip_summed = rcv_priv->ip_summed;
> +
> + dst_release(skb->dst);
> + skb->dst = NULL;
> + secpath_reset(skb);
> + nf_reset(skb);
> + skb->mark = 0;
> +
> + length = skb->len;
>
```
This should be done before you do the eth_type_trans, as that pulls the
header and your
byte counters will be off.

```
> +
> + stats->tx_bytes += length;
> + stats->tx_packets++;
>
> +
> + stats = per_cpu_ptr(rcv_priv->stats, cpu);
> + stats->rx_bytes += length;
> + stats->rx_packets++;
> +
> + netif_rx(skb);
> + return 0;
> +
> +outf:
> + kfree_skb(skb);
> + stats->tx_dropped++;
> + return 0;
> +}
>
```
Thanks,
Ben

--
Ben Greear <greearb@candelatech.com>
Candela Technologies Inc http://www.candelatech.com

Subject: Re: [PATCH] Virtual ethernet tunnel (v.2)
Posted by Pavel Emelianov on Thu, 07 Jun 2007 15:39:50 GMT
View Forum Message <> Reply to Message

Ben Greear wrote:
> Pavel Emelianov wrote:
>> Veth stands for Virtual ETHernet. It is a simple tunnel driver
>> that works at the link layer and looks like a pair of ethernet
>> devices interconnected with each other.
>>
> As Dave mentioned, there is already a driver known as 'veth'. Maybe borrow
> the etun name as well?

We have already seen that this driver uses ethXXX names for
its devices and Dave agreed with veth one. Moreover Alexey
Kuznetsov said that he would prefer the name veth for etun.

> I would also like some way to identify veth from other device types,
> preferably
> something like a value in sysfs. However, that should not hold up

We can do this with ethtool. It can get and print the driver
name of the device.

> consideration of
> this patch, and I am willing to submit a patch after this goes in to add
> the functionality
> I want...

Ok. Thanks.

```
>> +/*
>> + * xmit
>> + */
>> +
>> +static int veth_xmit(struct sk_buff *skb, struct net_device *dev)
>> +{
>> +    struct net_device *rcv = NULL;
>> +    struct veth_device_stats *stats;
>> +    struct veth_priv *priv, *rcv_priv;
```

```
>> +    int length, cpu;
>> +
>> +    skb_orphan(skb);
>> +
>> +    priv = netdev_priv(dev);
>> +    cpu = smp_processor_id();
>> +    stats = per_cpu_ptr(priv->stats, cpu);
>> +    rcv = priv->peer;
>> +
>> +    if (!(rcv->flags & IFF_UP))
>> +        goto outf;
>>
> I think you need at least the option to zero out the time-stamp,
> otherwise it will
> not be re-calculated when received on the peer, and it potentially spent
> significant
> time since it was last calculated (think netem delay or similar).
>
> +        /* Zero out the time-stamp so that receiving code is forced
> +         * to recalculate it.
> +         */
> +        skb->tstamp.off_sec = 0;
> +        skb->tstamp.off_usec = 0;
>
>> +
>> +    rcv_priv = netdev_priv(rcv);
>> +    skb->pkt_type = PACKET_HOST;
>> +    skb->protocol = eth_type_trans(skb, rcv);
>> +    if (dev->features & NETIF_F_NO_CSUM)
>> +        skb->ip_summed = rcv_priv->ip_summed;
>> +
>> +    dst_release(skb->dst);
>> +    skb->dst = NULL;
>> +    secpath_reset(skb);
>> +    nf_reset(skb);
>> +    skb->mark = 0;
>> +
>> +    length = skb->len;
>>
> This should be done before you do the eth_type_trans, as that pulls the
> header and your
> byte counters will be off.

This will be ETH_HLEN larger, do you mean this? I think this is
normal as this device tries to look like an "iron" ethernet card :)

>> +
>> +    stats->tx_bytes += length;
```

```
>> +    stats->tx_packets++;
>>   +
>> +    stats = per_cpu_ptr(rcv_priv->stats, cpu);
>> +    stats->rx_bytes += length;
>> +    stats->rx_packets++;
>> +
>> +    netif_rx(skb);
>> +    return 0;
>> +
>> +outf:
>> +    kfree_skb(skb);
>> +    stats->tx_dropped++;
>> +    return 0;
>> +}
>>
> Thanks,
> Ben
>
```