



# Linux虚拟网络设备之veth

赞7

收藏17

网络 linux wuyangchun 2017年05月01日发布

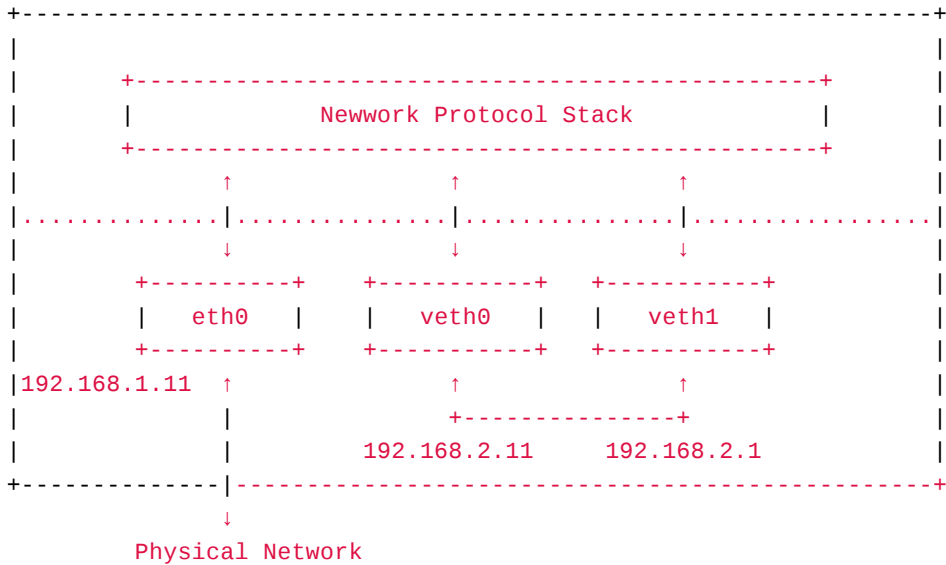
6.2k 次浏览

有了上一篇关于[tun/tap的介绍](#)之后，大家应该对虚拟网络设备有了一定的了解，本篇将接着介绍另一种虚拟网络设备veth。

## veth设备的特点

- veth和其它的网络设备都一样，一端连接的是内核协议栈。
- veth设备是成对出现的，另一端两个设备彼此相连
- 一个设备收到协议栈的数据发送请求后，会将数据发送到另一个设备上。

下面这张关系图很清楚的说明了veth设备的特点：



上图中，我们给物理网卡eth0配置的IP为192.168.1.11，而veth0和veth1的IP分别是192.168.2.11和192.168.2.1。

## 示例



首页



问答



专栏



讲堂



更多

## 只给一个veth设备配置IP

先通过ip link命令添加veth0和veth1，然后配置veth0的IP，并将两个设备都启动起来

```
dev@debian:~$ sudo ip link add veth0 type veth peer name veth1
dev@debian:~$ sudo ip addr add 192.168.2.11/24 dev veth0
dev@debian:~$ sudo ip link set veth0 up
dev@debian:~$ sudo ip link set veth1 up
```

这里不给veth1设备配置IP的原因就是想看看在veth1没有IP的情况下，veth0收到协议栈的数据后会不会转发给veth1。

ping一下192.168.2.1，由于veth1还没配置IP，所以肯定不通

```
dev@debian:~$ ping -c 4 192.168.2.1
PING 192.168.2.1 (192.168.2.1) 56(84) bytes of data.
From 192.168.2.11 icmp_seq=1 Destination Host Unreachable
From 192.168.2.11 icmp_seq=2 Destination Host Unreachable
From 192.168.2.11 icmp_seq=3 Destination Host Unreachable
From 192.168.2.11 icmp_seq=4 Destination Host Unreachable

--- 192.168.2.1 ping statistics ---
4 packets transmitted, 0 received, +4 errors, 100% packet loss, time 3015ms
pipe 3
```

但为什么ping不通呢？是到哪一步失败的呢？

先看看抓包的情况，从下面的输出可以看出，veth0和veth1收到了同样的ARP请求包，但没有看到ARP应答包：

```
dev@debian:~$ sudo tcpdump -n -i veth0
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on veth0, link-type EN10MB (Ethernet), capture size 262144 bytes
20:20:18.285230 ARP, Request who-has 192.168.2.1 tell 192.168.2.11, length 28
20:20:19.282018 ARP, Request who-has 192.168.2.1 tell 192.168.2.11, length 28
20:20:20.282038 ARP, Request who-has 192.168.2.1 tell 192.168.2.11, length 28
20:20:21.300320 ARP, Request who-has 192.168.2.1 tell 192.168.2.11, length 28
20:20:22.298783 ARP, Request who-has 192.168.2.1 tell 192.168.2.11, length 28
20:20:23.298923 ARP, Request who-has 192.168.2.1 tell 192.168.2.11, length 28

dev@debian:~$ sudo tcpdump -n -i veth1
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on veth1, link-type EN10MB (Ethernet), capture size 262144 bytes
20:20:48.570459 ARP, Request who-has 192.168.2.1 tell 192.168.2.11, length 28
20:20:49.570012 ARP, Request who-has 192.168.2.1 tell 192.168.2.11, length 28
```

[首页](#)[问答](#)[专栏](#)[讲堂](#)[更多](#)

```
20:20:51.570023 ARP, Request who-has 192.168.2.1 tell 192.168.2.11, length 28
20:20:52.569988 ARP, Request who-has 192.168.2.1 tell 192.168.2.11, length 28
20:20:53.570833 ARP, Request who-has 192.168.2.1 tell 192.168.2.11, length 28
```

为什么会这样呢？了解ping背后发生的事情后就明白了：

1. ping进程构造ICMP echo请求包，并通过socket发给协议栈，
2. 协议栈根据目的IP地址和系统路由表，知道去192.168.2.1的数据包应该要由192.168.2.11口出去
3. 由于是第一次访问192.168.2.1，且目的IP和本地IP在同一个网段，所以协议栈会先发送ARP出去，询问192.168.2.1的mac地址
4. 协议栈将ARP包交给veth0，让它发出去
5. 由于veth0的另一端连的是veth2，所以ARP请求包就转发给了veth1
6. veth1收到ARP包后，转交给另一端的协议栈
7. 协议栈一看自己的设备列表，发现本地没有192.168.2.1这个IP，于是就丢弃了该ARP请求包，这就是为什么只能看到ARP请求包，看不到应答包的原因

## 给两个veth设备都配置IP

给veth1也配置上IP

```
dev@debian:~$ sudo ip addr add 192.168.2.1/24 dev veth1
```

再ping 192.168.2.1成功（由于192.168.2.1是本地IP，所以默认会走lo设备，为了避免这种情况，这里使用ping命令带上了-I参数，指定数据包走指定设备）

```
dev@debian:~$ ping -c 4 192.168.2.1 -I veth0
PING 192.168.2.1 (192.168.2.1) from 192.168.2.11 veth0: 56(84) bytes of data.
64 bytes from 192.168.2.1: icmp_seq=1 ttl=64 time=0.032 ms
64 bytes from 192.168.2.1: icmp_seq=2 ttl=64 time=0.048 ms
64 bytes from 192.168.2.1: icmp_seq=3 ttl=64 time=0.055 ms
64 bytes from 192.168.2.1: icmp_seq=4 ttl=64 time=0.050 ms

--- 192.168.2.1 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3002ms
rtt min/avg/max/mdev = 0.032/0.046/0.055/0.009 ms
```

注意：对于非debian系统，这里有可能ping不通，主要是因为内核中的一些ARP相关配置导致veth1不返回ARP应答包，如ubuntu上就会出现这种情况，解决办法如下：

```
root@ubuntu:~# echo 1 > /proc/sys/net/ipv4/conf/veth1/accept_local
root@ubuntu:~# echo 1 > /proc/svs/net/inv4/conf/veth0/accept_local
```



首页



问答



专栏



讲堂



更多

```
root@ubuntu:~# echo 0 > /proc/sys/net/ipv4/conf/veth0/rp_filter
root@ubuntu:~# echo 0 > /proc/sys/net/ipv4/conf/veth1/rp_filter
```

再看看抓包情况，我们在veth0和veth1上都看到了ICMP echo的请求包，但为什么没有应答包呢？上面不是显示ping进程已经成功收到了应答包吗？

```
dev@debian:~$ sudo tcpdump -n -i veth0
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on veth0, link-type EN10MB (Ethernet), capture size 262144 bytes
20:23:43.113062 IP 192.168.2.11 > 192.168.2.1: ICMP echo request, id 24169, seq 1, length 64
20:23:44.112078 IP 192.168.2.11 > 192.168.2.1: ICMP echo request, id 24169, seq 2, length 64
20:23:45.111091 IP 192.168.2.11 > 192.168.2.1: ICMP echo request, id 24169, seq 3, length 64
20:23:46.110082 IP 192.168.2.11 > 192.168.2.1: ICMP echo request, id 24169, seq 4, length 64
```

```
dev@debian:~$ sudo tcpdump -n -i veth1
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on veth1, link-type EN10MB (Ethernet), capture size 262144 bytes
20:24:12.221372 IP 192.168.2.11 > 192.168.2.1: ICMP echo request, id 24174, seq 1, length 64
20:24:13.222089 IP 192.168.2.11 > 192.168.2.1: ICMP echo request, id 24174, seq 2, length 64
20:24:14.224836 IP 192.168.2.11 > 192.168.2.1: ICMP echo request, id 24174, seq 3, length 64
20:24:15.223826 IP 192.168.2.11 > 192.168.2.1: ICMP echo request, id 24174, seq 4, length 64
```

看看数据包的流程就明白了：

1. ping进程构造ICMP echo请求包，并通过socket发给协议栈，
2. 由于ping程序指定了走veth0，并且本地ARP缓存里面已经有了相关记录，所以不用再发送ARP出去，协议栈就直接将该数据包交给了veth0
3. 由于veth0的另一端连的是veth2，所以ICMP echo请求包就转发给了veth1
4. veth1收到ICMP echo请求包后，转交给另一端的协议栈
5. 协议栈一看自己的设备列表，发现本地有192.168.2.1这个IP，于是构造ICMP echo应答包，准备返回
6. 协议栈查看自己的路由表，发现回给192.168.2.11的数据包应该走lo口，于是将应答包交给lo设备
7. lo接到协议栈的应答包后，啥都没干，转手又把数据包还给了协议栈（相当于协议栈通过发送流程把数据包给lo，然后lo再将数据包交给协议栈的接收流程）
8. 协议栈收到应答包后，发现有socket需要该包，于是交给了相应的socket
9. 这个socket正好是ping进程创建的socket，于是ping进程收到了应答包

抓一下lo设备上的数据，发现应答包确实是从lo口回来的：

```
dev@debian:~$ sudo tcpdump -n -i lo
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
```

[首页](#)[问答](#)[专栏](#)[讲堂](#)[更多](#)

```
20:25:49.590273 IP 192.168.2.1 > 192.168.2.11: ICMP echo reply, id 24177, seq 1, length 64
20:25:50.590018 IP 192.168.2.1 > 192.168.2.11: ICMP echo reply, id 24177, seq 2, length 64
20:25:51.590027 IP 192.168.2.1 > 192.168.2.11: ICMP echo reply, id 24177, seq 3, length 64
20:25:52.590030 IP 192.168.2.1 > 192.168.2.11: ICMP echo reply, id 24177, seq 4, length 64
```

## 试着ping下其它的IP

ping 192.168.2.0/24网段的其它IP失败，ping一个公网的IP也失败：

```
dev@debian:~$ ping -c 1 -I veth0 192.168.2.2
PING 192.168.2.2 (192.168.2.2) from 192.168.2.11 veth0: 56(84) bytes of data.
From 192.168.2.11 icmp_seq=1 Destination Host Unreachable

--- 192.168.2.2 ping statistics ---
1 packets transmitted, 0 received, +1 errors, 100% packet loss, time 0ms

dev@debian:~$ ping -c 1 -I veth0 baidu.com
PING baidu.com (111.13.101.208) from 192.168.2.11 veth0: 56(84) bytes of data.
From 192.168.2.11 icmp_seq=1 Destination Host Unreachable

--- baidu.com ping statistics ---
1 packets transmitted, 0 received, +1 errors, 100% packet loss, time 0ms
```

从抓包来看，和上面第一种veth1没有配置IP的情况是一样的，ARP请求没人处理

```
dev@debian:~$ sudo tcpdump -i veth1
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on veth1, link-type EN10MB (Ethernet), capture size 262144 bytes
02:25:23.223947 ARP, Request who-has 192.168.2.2 tell 192.168.2.11, length 28
02:25:24.224352 ARP, Request who-has 192.168.2.2 tell 192.168.2.11, length 28
02:25:25.223471 ARP, Request who-has 192.168.2.2 tell 192.168.2.11, length 28
02:25:27.946539 ARP, Request who-has 123.125.114.144 tell 192.168.2.11, length 28
02:25:28.946633 ARP, Request who-has 123.125.114.144 tell 192.168.2.11, length 28
02:25:29.948055 ARP, Request who-has 123.125.114.144 tell 192.168.2.11, length 28
```

## 结束语

从上面的介绍中可以看出，从veth0设备出去的数据包，会转发到veth1上，如果目的地址是veth1的IP的话，就能被协议栈处理，否则连ARP那关都过不了，IP forward啥的都用不上，所以不借助其它虚拟设备的话，这样的数据包只能在本地协议栈里面打转转，没法走到eth0上去，即没法发送到外面的网络中去。

下一篇将介绍Linux下的网桥，到时候veth设备就有用武之地了。



首页



问答



专栏



讲堂



更多

## 参考

- [Linux Switching – Interconnecting Namespaces](#)

2017年05月01日发布 ...

赞 | 7

收藏 | 17

## 你可能感兴趣的文章

[centos6.6学习笔记：linux网络配置](#) 670 浏览

[飞龙的程序员书单 – Linux](#) 12 收藏, 1.3k 浏览

[【源起Netty 前传】Linux网络模型小记](#) 1 收藏, 398 浏览

## 2 条评论

默认排序 | 时间排序



ruanhao · 2017年06月28日

"这个socket正好是ping进程创建的socket，于是ping进程收到了应答包"，因为 veth0 和 veth1 其实共享了一个namespace下的protocol stack，若 veth0 和 veth1 分属不同的ns，就ping不通了，即使lo收到echo request :)

👍 赞 回复



yiqiaoxihui · 1月30日

"协议栈查看自己的路由表，发现回给192.168.2.11的数据包应该走lo口"，请问为什么路由表中记录的192.168.2.11的数据包应该走lo口？我查看路由表，并没有此记录啊？谢谢！

👍 赞 回复



首页



问答



专栏



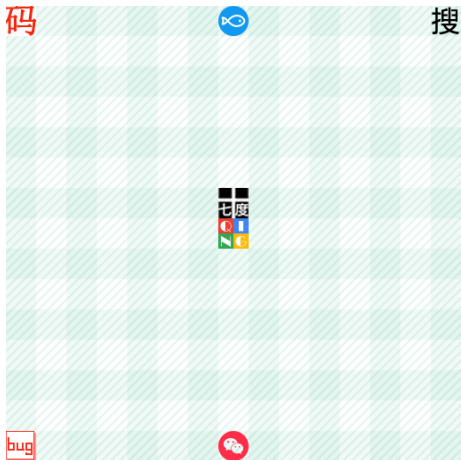
讲堂



更多

文明社会，理性评论

发布评论



想在上方展示你的广告？



wuyangchun

关注作者

899 声望

发布于专栏

Linux程序员

专注Linux相关技术

103 人关注

关注专栏

系列文章

Linux虚拟网络设备之tun/tap 43 收藏， 8k 浏览

Linux虚拟网络设备之bridge(桥) 33 收藏， 5.8k 浏览



首页



问答



专栏



讲堂



更多

- 热门问答

热门专栏

热门讲堂

最新活动

圈子

找工作

移动客户端
- 每周精选

用户排行榜

徽章

帮助中心

声望与权限

社区服务中心

开发手册
- 人才服务

企业培训

活动策划

广告投放

区块链解决方案

合作联系
- 关于我们

加入我们

联系我们
- 产品技术日志


社区运营日志

市场运营日志

团队日志

社区访谈
- 服务条款

内容许可



扫一扫下载 App

Copyright © 2011-2018 SegmentFault. 当前呈现版本 17.06.16

浙ICP备 15005796号-2 浙公网安备 33010602002000号 杭州堆栈科技有限公司版权所有

CDN 存储服务由 又拍云 赞助提供