003-MACVLAN

1. MACVLAN简述

macvlan本身是linux kernel模块，其功能是允许在同一个物理网卡上配置多个mac地址，也就是多个interface，每个interface可以配置自己的ip。macvlan下的虚拟机或者容器网络在同一个网络中，共享同一个广播域。macvlan和bridge比较类似，省去了bridge的存在，配置简单，效率也相对较高，除此之外macvlan也完美支持vlan。

一句话，macvlan相当于物理网卡施展了分身之术，由一个变多个。

2.MACVLAN工作原理

macvlan是linux kernel支持的新特性，一般是以内核模块的形式存在，我们可以通过以下方法判断当前系统是否支持

```shell
# lsmod | grep macvlan
macvlan                24576  0
```

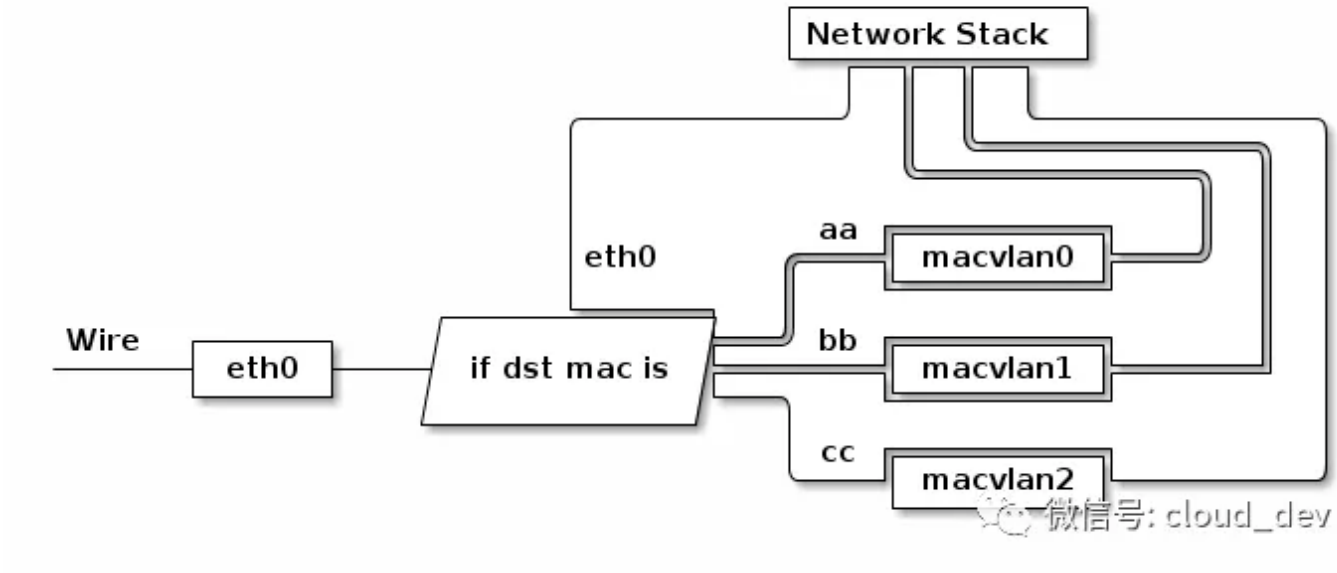
如果没有加载可以手动加载该mod

```shell
# modprobe macvlan
```

> 如果第一个命令报错，或者第二个命令没有返回，说明当前系统不支持 macvlan，需要升级内核。

macvlan听起来跟vlan很像，但是他们实现的机制是完全不同的，macvlan子接口和原来的主接口（父接口）是完全独立的，可以单独配置mac地址和ip地址，而vlan子接口和主接口共用相同的mac地址。vlan用来划分广播域。macvlan是共享同一个广播域。

通过不同的子接口，macvlan也能做到流量的隔离，macvlan会根据收到包的目的mac地址判断这个包，转发给哪个子接口，然后子接口再把包交给上层的协议栈处理。



3.MACVLAN模式

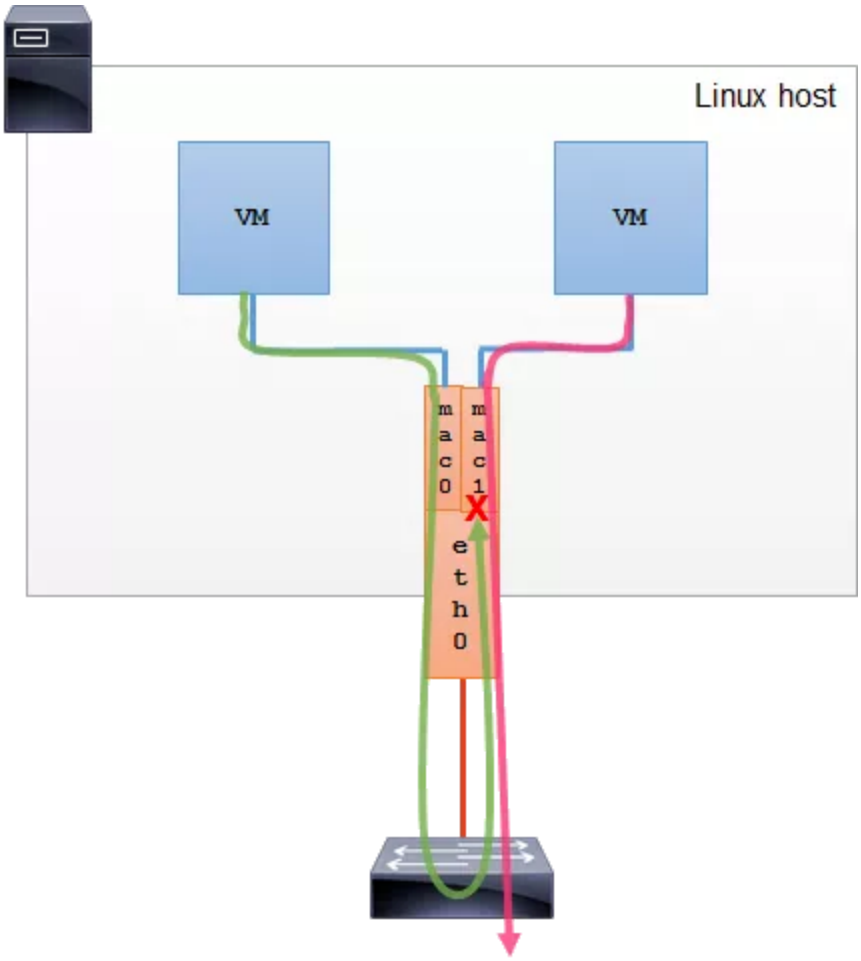
根据macvlan子接口之间的通信模式，macvlan有四中网络模式：

- private模式
- vepa（virtual ethernet port aggregator）模式
- bridge模式
- passthrough模式

> 默认是vepa模式

3.1 Private模式

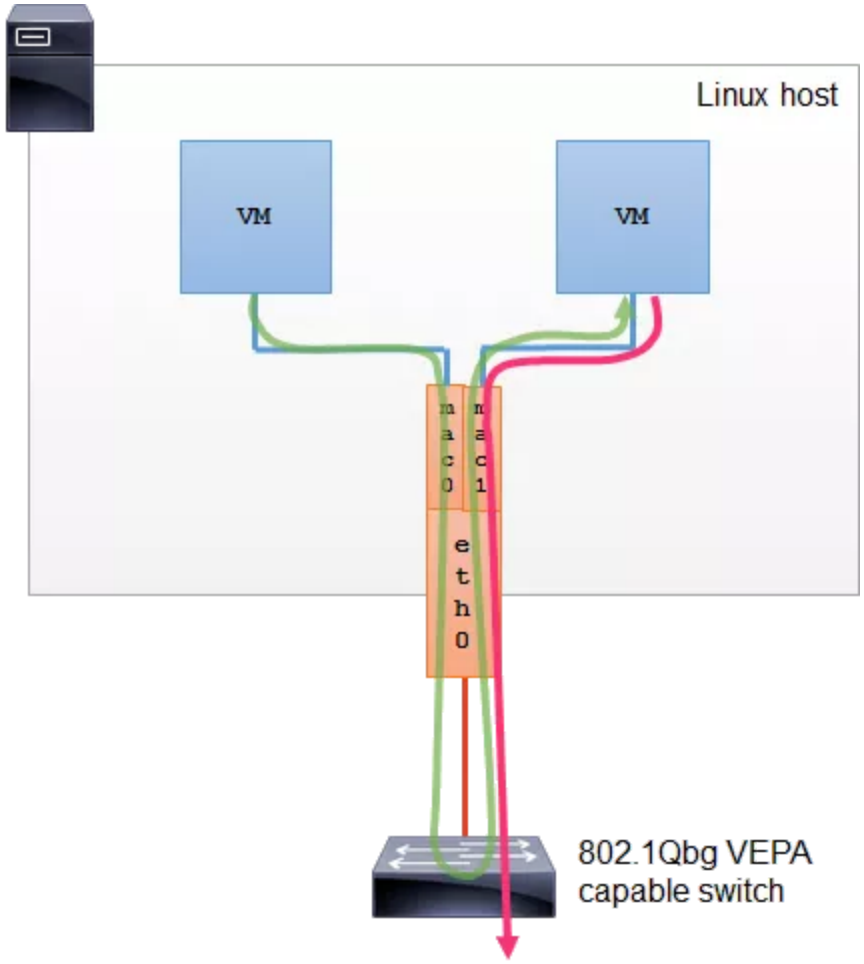
这种模式下，同一主接口下的子接口之间彼此隔离，不能通信。即使从外部的物理交换机导流，也会被无情地丢掉。



3.2 Vepa模式

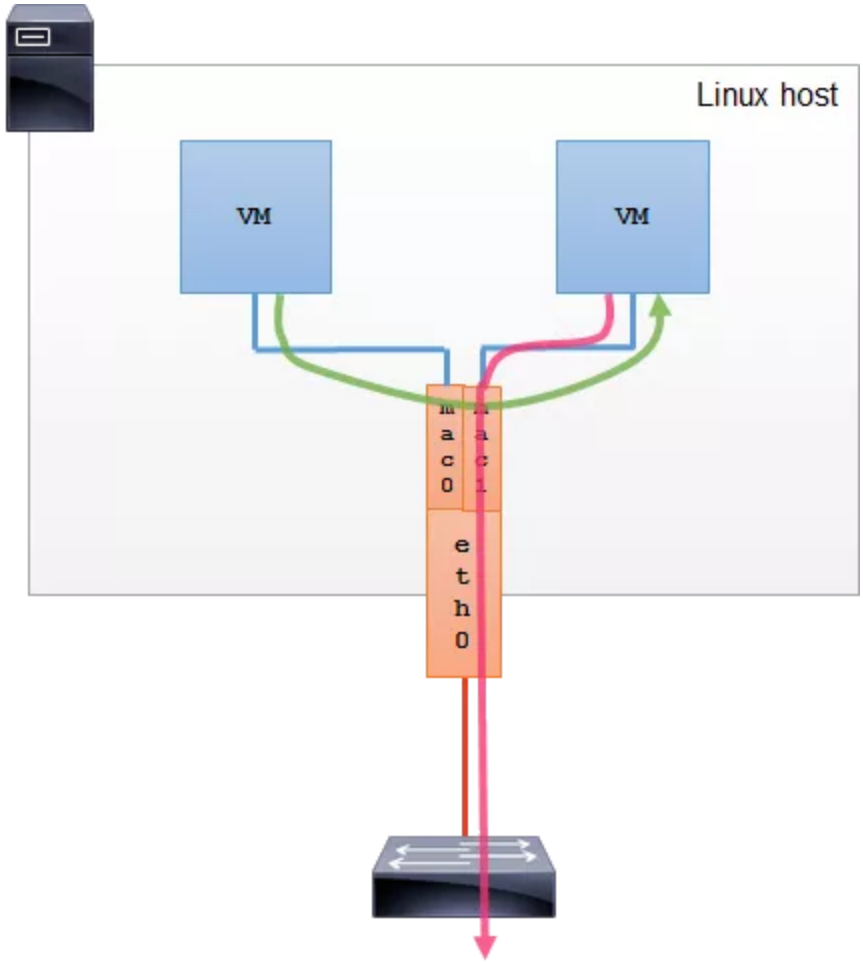
这种模式下，子接口之间的通信流量需要导到外部支持802.1Qbg/VPEA 功能的交换机上（可以是物理的或者虚拟的），经由外部交换机转发，再绕回来。

> 注：802.1Qbg/VPEA 功能简单说就是交换机要支持 **发夹（hairpin） 功能**，也就是数据包从一个接口上收上来之后还能再扔回去。



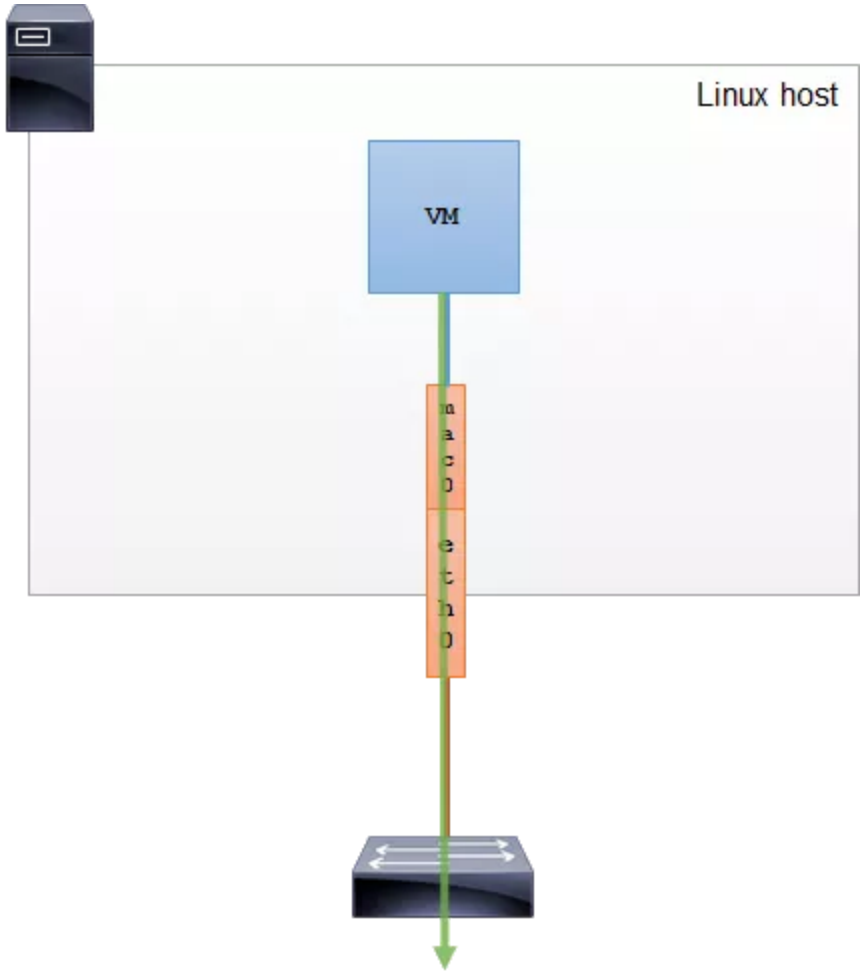
3.3 Bridge模式

这种模式下，模拟的是 Linux bridge 的功能，但比 bridge 要好的一点是每个接口的 MAC 地址是已知的，不用学习。所以，这种模式下，子接口之间就是直接可以通信的。



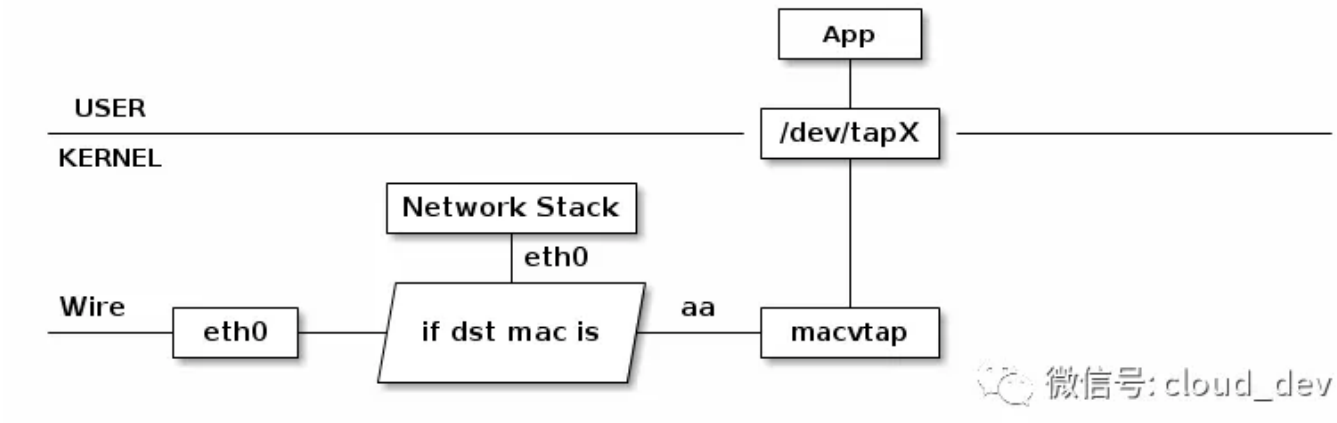
3.4 Passthrugh模式

这种模式，只允许单个子接口连接主接口，且必须设置成混杂模式，一般用于子接口桥接和创建 VLAN 子接口的场景。



4 MACTAP

和 macvlan 相似的技术还有一种是 mactap。和 macvlan 不同的是，mactap 收到包之后不是交给协议栈，而是交给一个 tapX 文件，然后通过这个文件，完成和用户态的直接通信。



5. MACVLAN实验

该实验以bridge模式为例

1. 设置网卡为混杂模式

   设置前

```shell
ens192: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.100.60  netmask 255.255.255.0  broadcast 192.168.100.255
        inet6 fe80::20c:29ff:fe78:6777  prefixlen 64  scopeid 0x20<link>
        ether 00:0c:29:78:67:77  txqueuelen 1000  (Ethernet)
        RX packets 1237  bytes 110687 (110.6 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 1234  bytes 305505 (305.5 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

```

   设置后

```shell
root@ubuntu:~# ifconfig ens192  promisc
root@ubuntu:~# ifconfig
ens192: flags=4419<UP,BROADCAST,RUNNING,PROMISC,MULTICAST>  mtu 1500
        inet 192.168.100.60  netmask 255.255.255.0  broadcast 192.168.100.255
        inet6 fe80::20c:29ff:fe78:6777  prefixlen 64  scopeid 0x20<link>
        ether 00:0c:29:78:67:77  txqueuelen 1000  (Ethernet)
        RX packets 1276  bytes 113985 (113.9 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 1261  bytes 310035 (310.0 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

```

2. 创建两个macvlan接口，其parent接口都是ens192(观察mac地址，网卡状态)

```shell
root@ubuntu:~# ip link add link ens192  name macv1 type macvlan mode bridge
root@ubuntu:~# ip link add link ens192  name macv2 type macvlan mode bridge
root@ubuntu:~# ip a
3: ens192: <BROADCAST,MULTICAST,PROMISC,UP,LOWER_UP> mtu 1500 qdisc mq state UP group default qlen 1000
    link/ether 00:0c:29:78:67:77 brd ff:ff:ff:ff:ff:ff
    altname enp11s0
    inet 192.168.100.60/24 brd 192.168.100.255 scope global ens192
       valid_lft forever preferred_lft forever
    inet6 fe80::20c:29ff:fe78:6777/64 scope link
       valid_lft forever preferred_lft forever
4: macv1@ens192: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN group default qlen 1000
    link/ether 16:4d:91:e2:26:e2 brd ff:ff:ff:ff:ff:ff
5: macv2@ens192: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN group default qlen 1000
    link/ether 52:a7:c5:dd:7b:d9 brd ff:ff:ff:ff:ff:ff
root@ubuntu:~#

```

3. 创建namespace，并将macvlan的interface插入namespace中（观察interface的变化）

```shell
root@ubuntu:~# ip netns add net1
root@ubuntu:~# ip link set macv1 netns net1
root@ubuntu:~# ip netns add net2
root@ubuntu:~# ip link set macv2 netns net2
root@ubuntu:~# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: ens160: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP group default qlen 1000
    link/ether 00:0c:29:78:67:6d brd ff:ff:ff:ff:ff:ff
    altname enp3s0
    inet 10.18.18.62/24 metric 100 brd 10.18.18.255 scope global dynamic ens160
       valid_lft 42202sec preferred_lft 42202sec
    inet6 240e:3b7:614:15d0:20c:29ff:fe78:676d/64 scope global dynamic mngtmpaddr noprefixroute
       valid_lft 234369sec preferred_lft 147969sec
    inet6 fe80::20c:29ff:fe78:676d/64 scope link
       valid_lft forever preferred_lft forever
3: ens192: <BROADCAST,MULTICAST,PROMISC,UP,LOWER_UP> mtu 1500 qdisc mq state UP group default qlen 1000
    link/ether 00:0c:29:78:67:77 brd ff:ff:ff:ff:ff:ff
    altname enp11s0
    inet 192.168.100.60/24 brd 192.168.100.255 scope global ens192
       valid_lft forever preferred_lft forever
    inet6 fe80::20c:29ff:fe78:6777/64 scope link
       valid_lft forever preferred_lft forever
root@ubuntu:~#
```

4. 设置macvlan的网卡IP,设置网卡UP状态

> 特别注意ip地址与parent的ip地址不在同一网段

```shell
root@ubuntu:~# ip netns exec net1 ip addr add 52.1.1.151/24 dev macv1
root@ubuntu:~# ip netns exec net1 ip link set macv1 up
root@ubuntu:~# ip netns exec net2 ip addr add 52.1.1.152/24 dev macv2
root@ubuntu:~# ip netns exec net2 ip link set macv2 up
root@ubuntu:~#
```

5. 进入namespace中查看网卡状态

> 此时lo网卡状态为down状态，若ping自己则ping不通

```shell
root@ubuntu:~#
root@ubuntu:~# ip netns exec net1 ip a
1: lo: <LOOPBACK> mtu 65536 qdisc noop state DOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
4: macv1@if3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default qlen 1000
    link/ether 16:4d:91:e2:26:e2 brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet 52.1.1.151/24 scope global macv1
       valid_lft forever preferred_lft forever
    inet6 fe80::144d:91ff:fee2:26e2/64 scope link
       valid_lft forever preferred_lft forever
root@ubuntu:~# ip netns exec net2 ip a
1: lo: <LOOPBACK> mtu 65536 qdisc noop state DOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
5: macv2@if3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default qlen 1000
    link/ether 52:a7:c5:dd:7b:d9 brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet 52.1.1.152/24 scope global macv2
       valid_lft forever preferred_lft forever
    inet6 fe80::50a7:c5ff:fedd:7bd9/64 scope link
       valid_lft forever preferred_lft forever
root@ubuntu:~#
root@ubuntu:~#

```

6. 在net1中ping测net2

```shell
root@ubuntu:~# ip netns exec net2 ping 52.1.1.151
PING 52.1.1.151 (52.1.1.151) 56(84) bytes of data.
64 bytes from 52.1.1.151: icmp_seq=1 ttl=64 time=0.110 ms
64 bytes from 52.1.1.151: icmp_seq=2 ttl=64 time=0.039 ms
64 bytes from 52.1.1.151: icmp_seq=3 ttl=64 time=0.055 ms
64 bytes from 52.1.1.151: icmp_seq=4 ttl=64 time=0.053 ms
^C
--- 52.1.1.151 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3049ms
rtt min/avg/max/mdev = 0.039/0.064/0.110/0.027 ms
root@ubuntu:~#
```

可以看到，能够 ping 通，如果把上面的 mode 换成其他模式就行不通了，这个就留给大家去实验了（默认是 vepa 模式）。

6. 总结

1. macvlan并不创建网络，只是创建网卡，将一张物理网卡设置多个mac地址，就是一变多，一对多；类似于鸣人的影分身之术， 注意： 需要物理网卡，打开混杂模式
2. macvlan会共享物理网卡所链接的外部网络，实现的效果跟桥接模式是一样的
3. macvlan的使用场景

- macvlan主要是用来解决效率问题
- macvlan是效率贵高的跨主机网络虚拟化解决方案之一
- 适合在对网络性能要求极高的场景下

4. macvlan是linux kernel提供的一种network driver类型， 如何查看当前内核是否加载了该driver呢?

- lsmod | grep macvlan （查看是否加载了）
- modprobe macvlan (手动加载macvlan驱动到内核)
- /drivers/net/macvlan.c (源码地址)