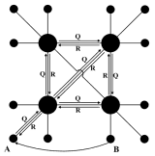


# 从Docker到Kubernetes 第6周

# 法律声明

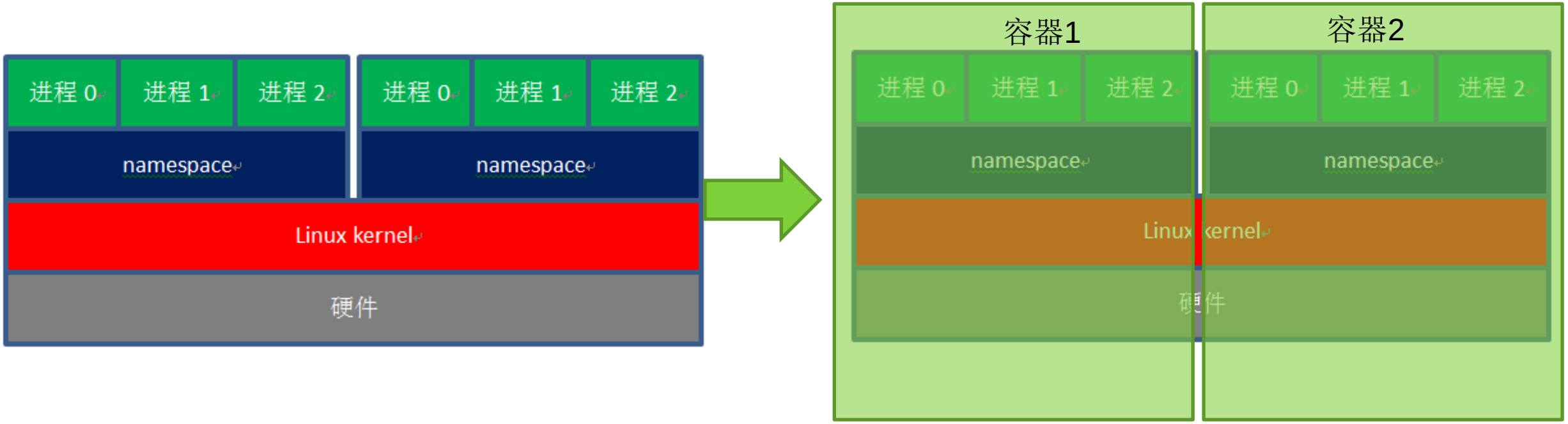
【声明】本视频和幻灯片为炼数成金网络课程的教学资料，所有资料只能在课程内使用，不得在课程以外范围散播，违者将可能被追究法律和经济责任。

课程详情访问炼数成金培训网站

http://edu.dataguru.cn

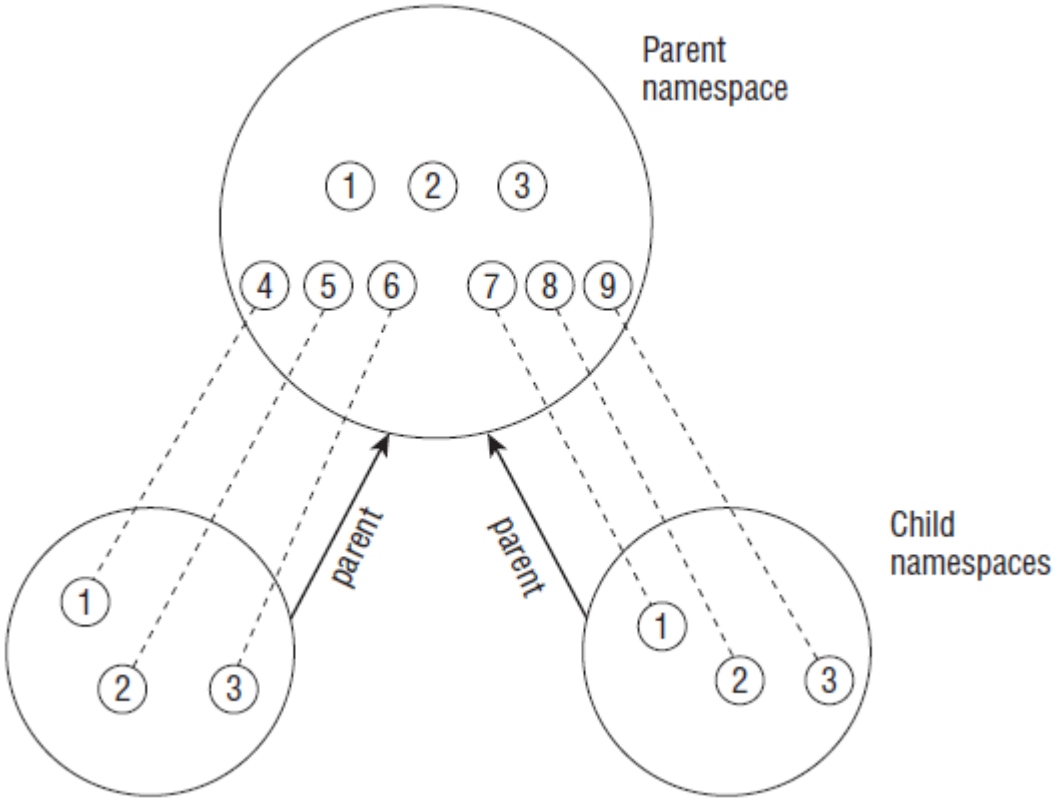
- Linux namespace详解
- ovs+Docker实战

# Linux namespace详解



# Linux namespace详解

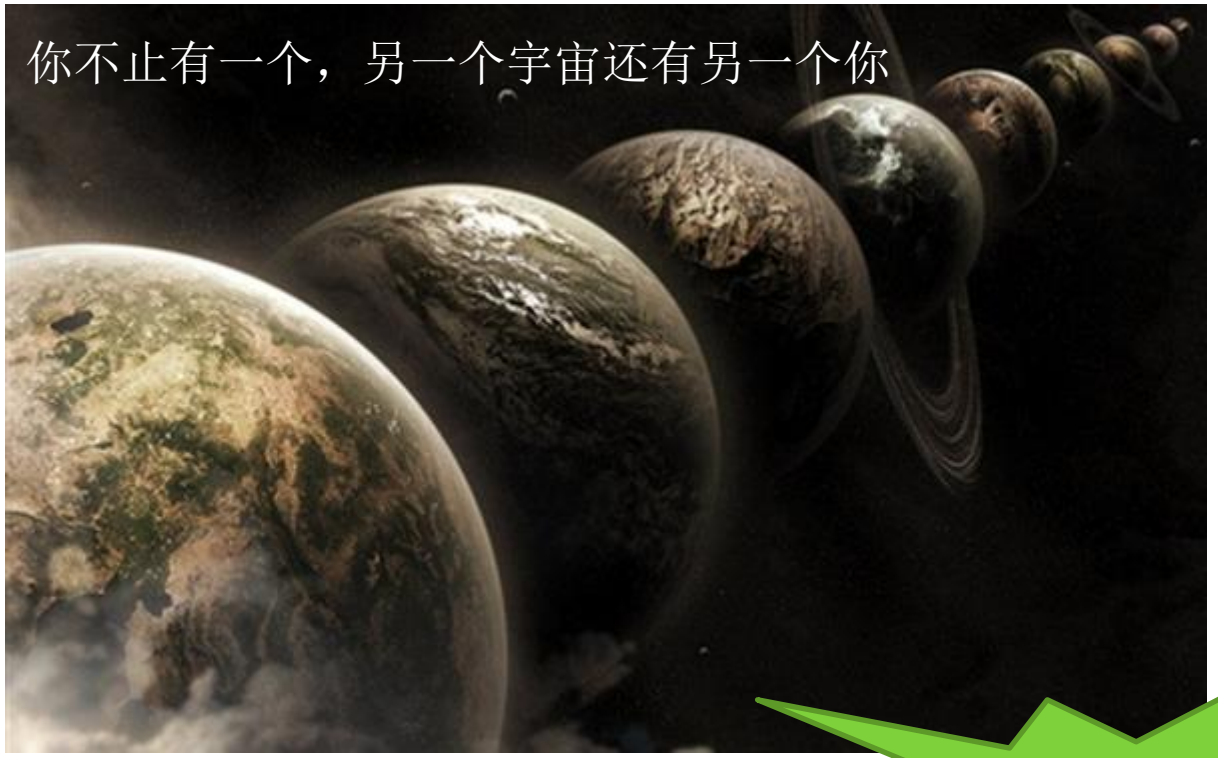
## namespace & container




namespace还拥有层次关系。图3中，一个parent namespace下有两个child namespace。parent namespace和它的两个child namespace都有三个进程号为1,2,3的进程，同时child namespace的每个进程被映射到了parent namespace中的4,5,6,7,8,9。虽然只有9个进程，但需要15个进程号来表示它们。

# Linux namespace详解

## namespace & container

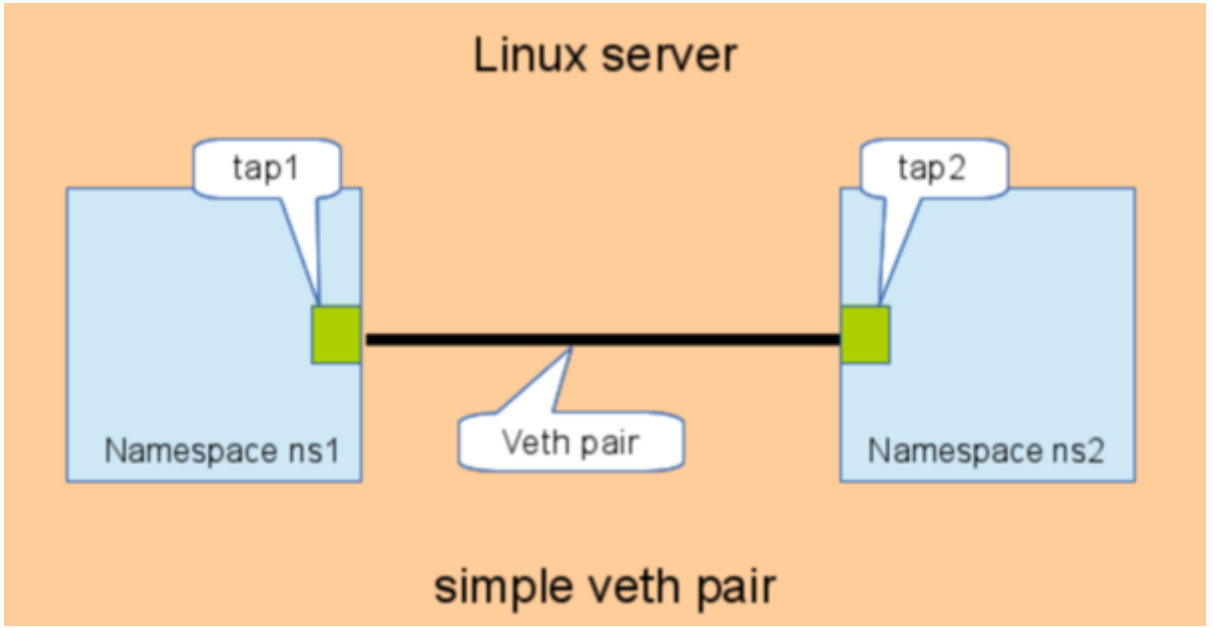



今年4月份，有人问霍金：“单向乐队（One Direction）的成员Zayn离队让全球无数少女心碎不已，这件事会产生怎样的宇宙效应呢？”

我们如穿越？

# Linux namespace详解

## namespace & container



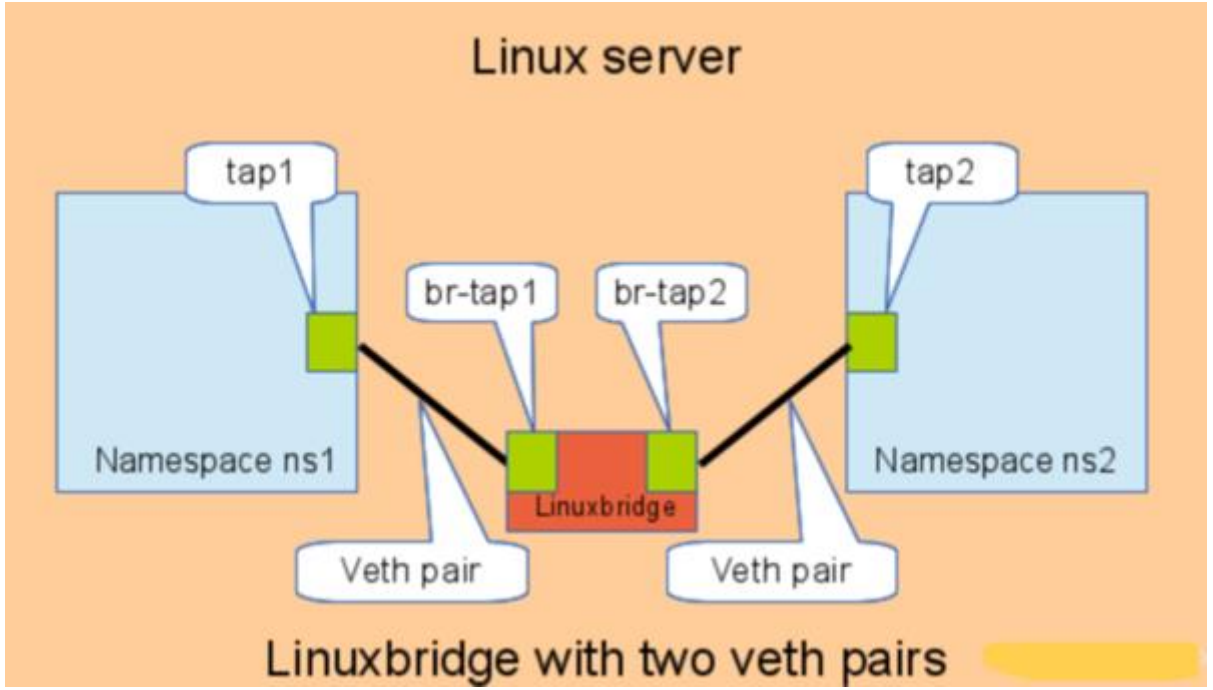

veth pair是用于不同network namespace间进行通信的方式，veth pair将一个network namespace数据发往另一个network namespace的veth。

点对点模式

交换模式如何解决？

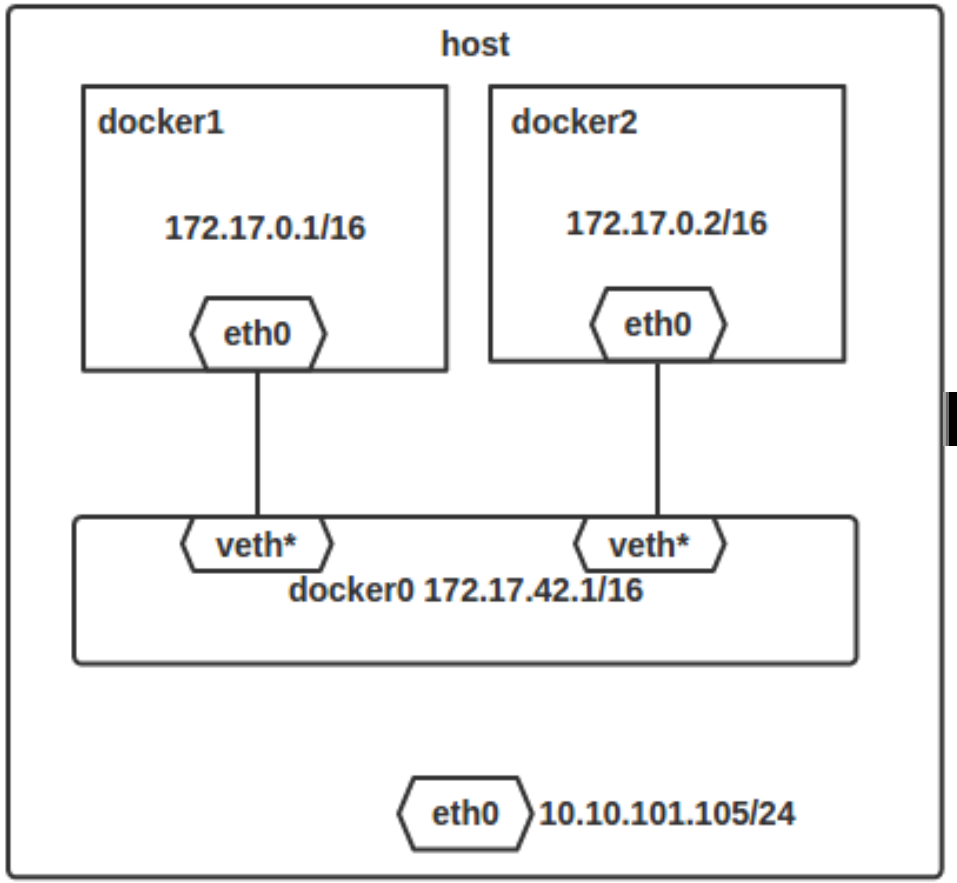
# Linux namespace详解

## namespace & container




Linux Bridge可以实现类似交换机的工作模式，将多个不同Namespace上的网卡连通

# Linux namespace详解




## namespace & container

```
[root@docker128 ~]# brctl show
bridge name     bridge id               STP enabled     interfaces
docker0         8000.02429f82646d       no
```

启动一个容器以后

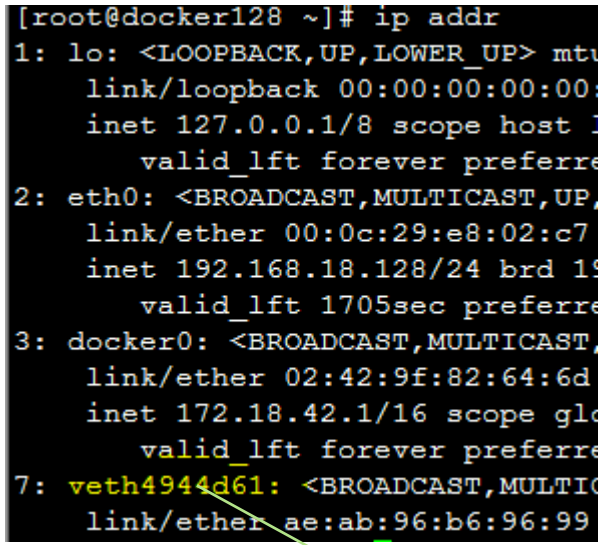
```
[root@docker128 ~]# docker run --rm=true -it java /bin/bash
```

```
bridge name     bridge id               STP enabled     interfaces
docker0         8000.02429f82646d       no              veth4944d61
```

```
[root@docker128 ~]# ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu
    link/loopback 00:00:00:00:00:
    inet 127.0.0.1/8 scope host l
       valid_lft forever preferre
2: eth0: <BROADCAST,MULTICAST,UP,
    link/ether 00:0c:29:e8:02:c7
    inet 192.168.18.128/24 brd 19
       valid_lft 1705sec preferre
3: docker0: <BROADCAST,MULTICAST,
    link/ether 02:42:9f:82:64:6d
    inet 172.18.42.1/16 scope glo
       valid_lft forever preferre
7: veth4944d61: <BROADCAST,MULTIC
    link/ether ae:ab:96:b6:96:99
```

docker inspect -f '{{.State.Pid}}' containerId 得到容器的真正pid

# Linux namespace详解




## namespace & container

docker inspect -f '{{.State.Pid}}' containerId 得到容器的真正pid 3120

```
mkdir -p /var/run/netns

ln -s /proc/3120/ns/net /var/run/netns/3120
```

```
[root@docker128 ~]# ln -s /proc/3120/ns/net /var/run/netns/3120
[root@docker128 ~]# ip netns ls
3120
[root@docker128 ~]# ip netns exec 3120 ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN
    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
6: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP
    link/ether 02:42:ac:12:00:02 brd ff:ff:ff:ff:ff:ff
    inet 172.18.0.2/16 scope global eth0
       valid_lft forever preferred_lft forever
```

# Linux namespace详解

## namespace & container

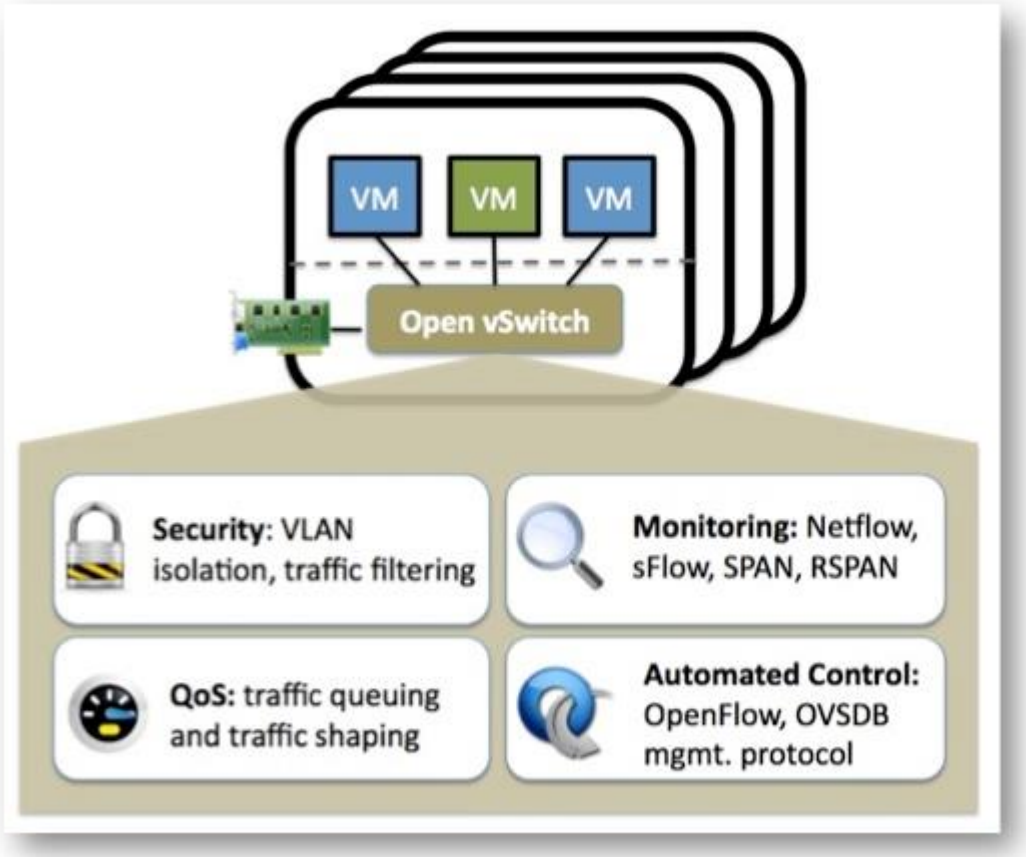
ip netns exec 3120 ethtool -S eth0

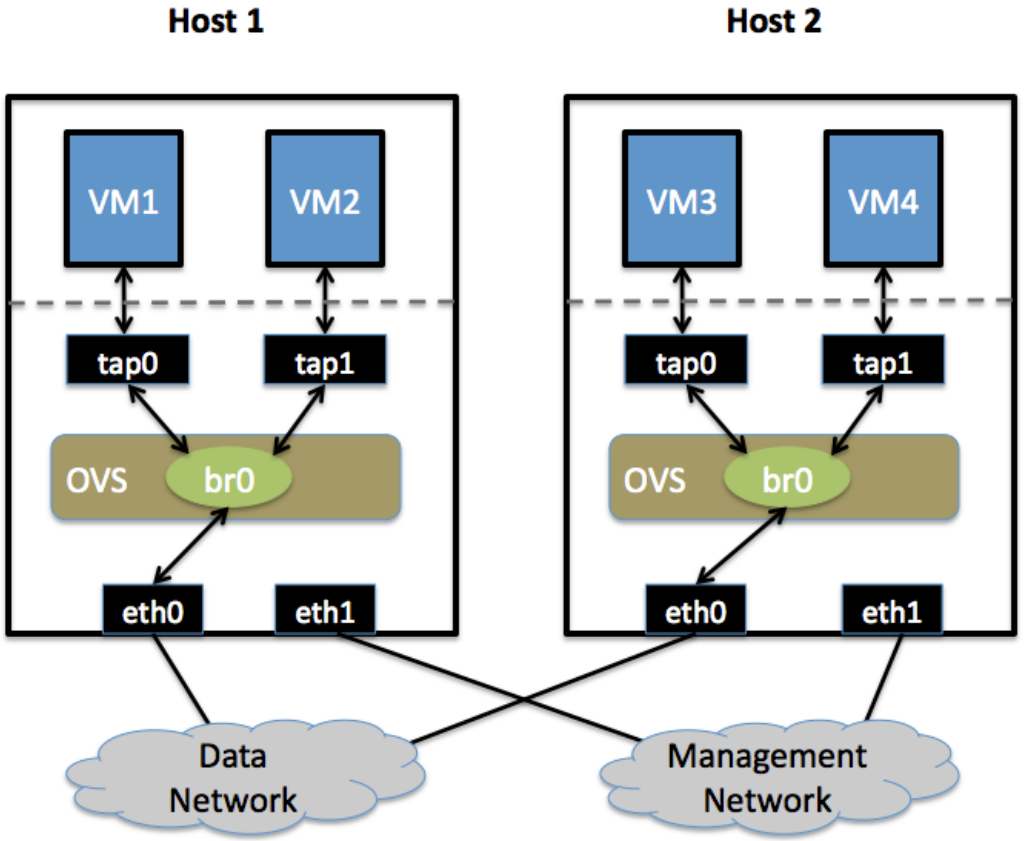
```
[root@docker128 ~]# ip netns exec 3120 ethtool -S eth0
NIC statistics:
     peer_ifindex: 7
```

在本空间执行ip a

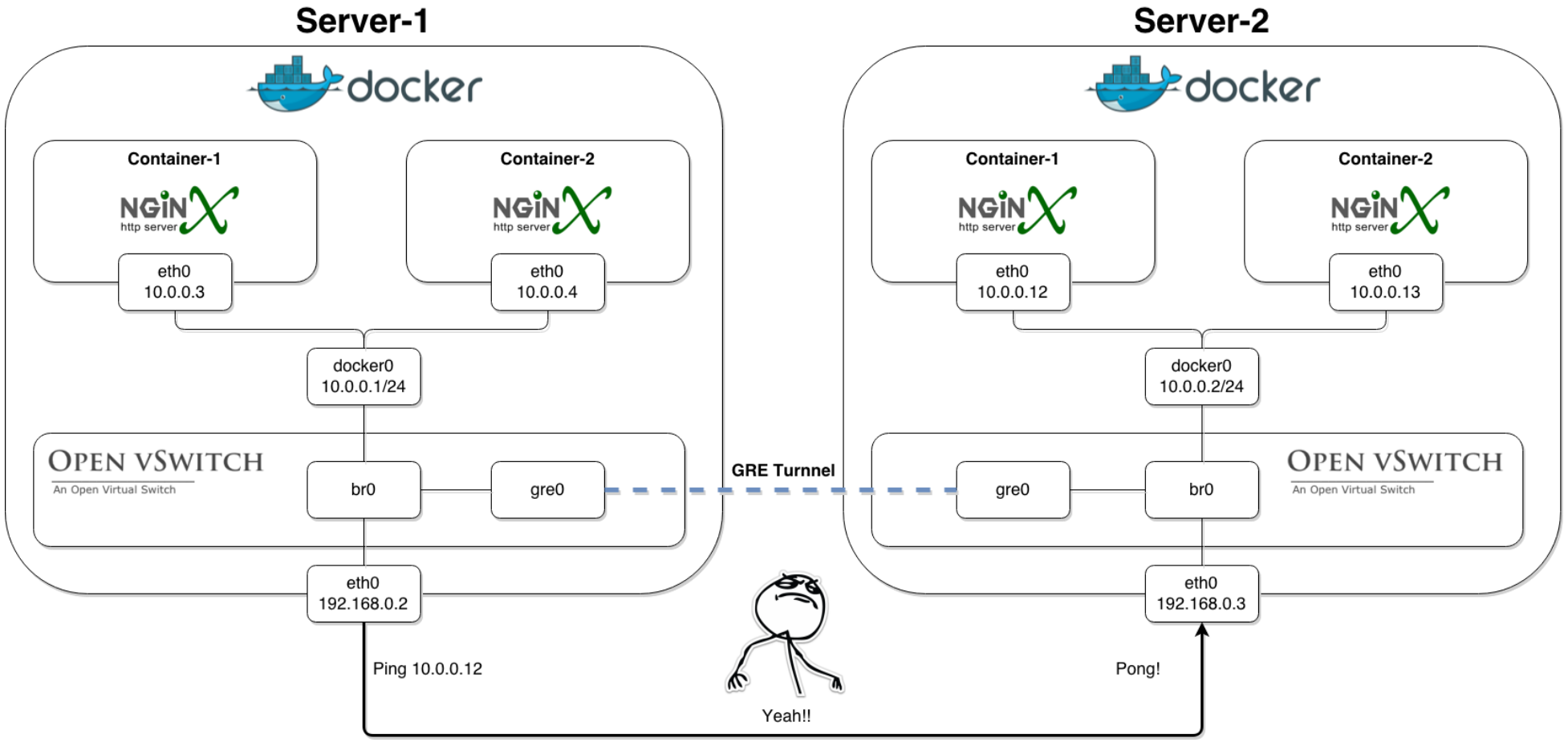
```
[root@docker128 ~]#  ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP qlen 1000
    link/ether 00:0c:29:e8:02:c7 brd ff:ff:ff:ff:ff:ff
    inet 192.168.18.128/24 brd 192.168.18.255 scope global dynamic eth0
       valid_lft 1095sec preferred_lft 1095sec
3: docker0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP
    link/ether 02:42:9f:82:64:6d brd ff:ff:ff:ff:ff:ff
    inet 172.18.42.1/16 scope global docker0
       valid_lft forever preferred_lft forever
7: veth4944d61: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue master docker0 state UP
    link/ether ae:ab:96:b6:96:99 brd ff:ff:ff:ff:ff:ff
```

# OVS +Docker

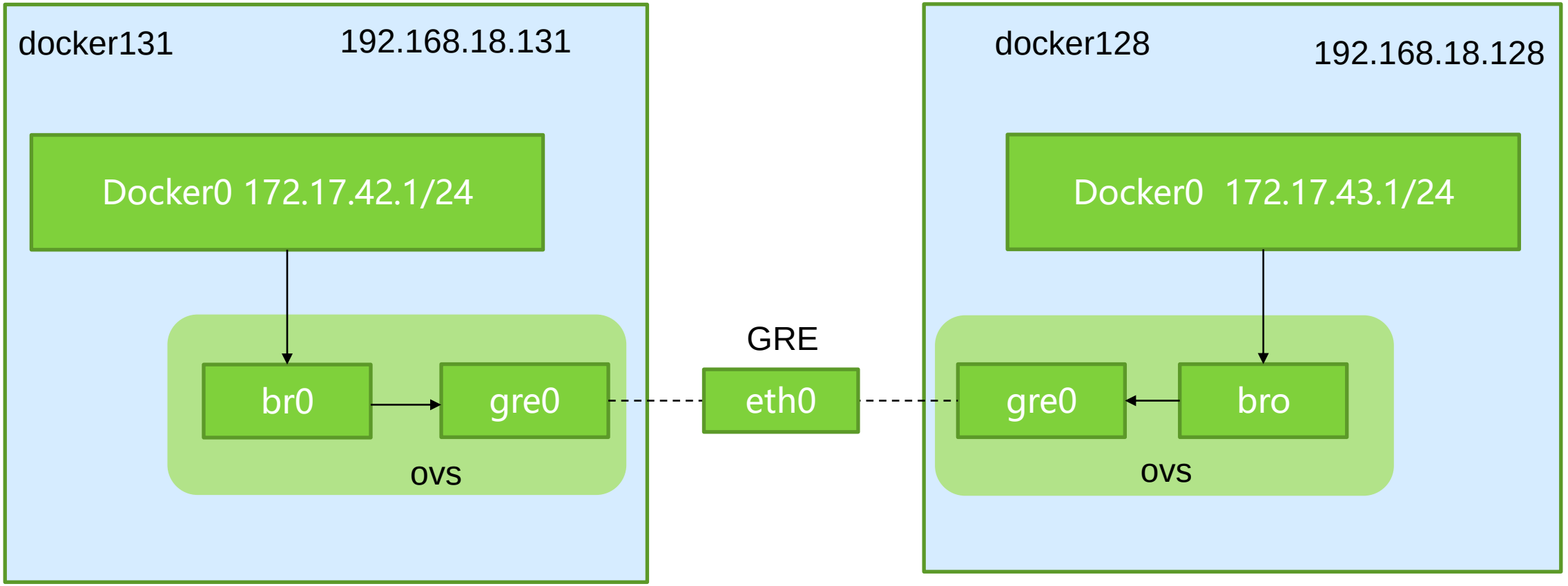

VM
VM
VM
Open vSwitch
Security: VLAN isolation, traffic filtering
Monitoring: Netflow, sFlow, SPAN, RSPAN
QoS: traffic queuing and traffic shaping
Automated Control: OpenFlow, OVSDB mgmt. protocol


Host 1
Host 2
VM1
VM2
VM3
VM4
tap0
tap1
tap0
tap1
OVS
br0
OVS
br0
eth0
eth1
eth0
eth1
Data Network
Management Network

# OVS +Docker


Server-1
docker
Container-1
NGINX
http server
eth0
10.0.0.3
Container-2
NGINX
http server
eth0
10.0.0.4
docker0
10.0.0.1/24
OPEN VSWITCH
An Open Virtual Switch
br0
gre0
eth0
192.168.0.2
Ping 10.0.0.12
GRE Turnnel
Yeah!!
Server-2
docker
Container-1
NGINX
http server
eth0
10.0.0.12
Container-2
NGINX
http server
eth0
10.0.0.13
docker0
10.0.0.2/24
gre0
br0
OPEN VSWITCH
An Open Virtual Switch
eth0
192.168.0.3
Pong!

# OVS +Docker


docker131
192.168.18.131
Docker0 172.17.42.1/24
br0
gre0
ovs
GRE
eth0
docker128
192.168.18.128
Docker0 172.17.43.1/24
gre0
bro
ovs

# OVS +Docker

vi /etc/selinux/config

SELINUX=disabled
重启

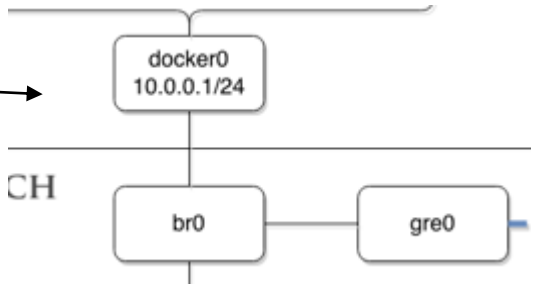
yum install openvswitch-2.4.0-1.x86_64.rpm

```
[root@docker128 ~]# service openvswitch restart
Restarting openvswitch (via systemctl):                    [  OK  ]
[root@docker128 ~]# service openvswitch status
ovsdb-server is running with pid 2429
ovs-vswitchd is running with pid 2439
```

```
[root@docker128 ~]# tail /var/log/messages
Sep 19 06:18:00 docker128 openvswitch: Killing ovs-vswitchd (882) [  OK  ]
Sep 19 06:18:00 docker128 openvswitch: Killing ovsdb-server (840) [  OK  ]
Sep 19 06:18:00 docker128 systemd: Starting LSB: Open vSwitch switch...
Sep 19 06:18:00 docker128 openvswitch: Starting ovsdb-server [  OK  ]
Sep 19 06:18:00 docker128 ovs-vsctl: ovs|00001|vsctl|INFO|Called as ovs-vsctl --no-wait -- init -- set Open_vSwitch . db-version=7.12.1
Sep 19 06:18:00 docker128 ovs-vsctl: ovs|00001|vsctl|INFO|Called as ovs-vsctl --no-wait set Open_vSwitch . ovs-version=2.4.0 "external-ids:syst
d=\"ec2be735-38df-49b0-b143-9e8ef62e7d68\"" "system-type=\"unknown\"" "system-version=\"unknown\""
Sep 19 06:18:00 docker128 openvswitch: Configuring Open vSwitch system IDs [  OK  ]
Sep 19 06:18:00 docker128 openvswitch: Starting ovs-vswitchd [  OK  ]
Sep 19 06:18:00 docker128 openvswitch: Enabling remote OVSDB managers [  OK  ]
Sep 19 06:18:00 docker128 systemd: Started LSB: Open vSwitch switch.
```

# OVS +Docker

ovs-vsctl add-br br0
ovs-vsctl add-port br0 gre1 -- set interface gre1 type=gre
option:remote_ip=192.168.18.128
#添加br0到本地docker0，使得容器流量通过OVS流经tunnel
brctl addif docker0 br0




ip link set dev br0 up
ip link set dev docker0 up

iptables -t nat  -F;iptables   -F

ip route add 172.17.0.0/16 dev docker0

# OVS +Docker

```
[root@docker131 ~]# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN
    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
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP qlen 1000
    link/ether 00:0c:29:55:5e:c3 brd ff:ff:ff:ff:ff:ff
    inet 192.168.18.131/24 brd 192.168.18.255 scope global dynamic eth0
       valid_lft 1357sec preferred_lft 1357sec
3: ovs-system: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN
    link/ether 46:0d:04:4f:04:11 brd ff:ff:ff:ff:ff:ff
10: docker0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP
    link/ether 92:8d:d0:a4:ca:45 brd ff:ff:ff:ff:ff:ff
    inet 172.17.42.1/24 scope global docker0
       valid_lft forever preferred_lft forever
11: br0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noo
    link/ether 92:8d:d0:a4:ca:45 brd ff:ff:ff:ff:ff:ff
[root@docker131 ~]# ip route
default via 192.168.18.2 dev eth0  proto static  metric 100
172.17.0.0/16 dev docker0  scope link
172.17.42.0/24 dev docker0  proto kernel  scope link  src 172
192.168.18.0/24 dev eth0  proto kernel  scope link  src 192.1
192.168.18.0/24 dev eth0  proto kernel  scope link  src 192.1
```

```
[root@docker128 ~]# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN
    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
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP qlen 1000
    link/ether 00:0c:29:e8:02:c7 brd ff:ff:ff:ff:ff:ff
    inet 192.168.18.128/24 brd 192.168.18.255 scope global dynamic eth0
       valid_lft 1209sec preferred_lft 1209sec
3: ovs-system: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN
    link/ether e2:e2:97:83:c6:35 brd ff:ff:ff:ff:ff:ff
16: docker0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP
    link/ether ba:89:14:e0:7f:43 brd ff:ff:ff:ff:ff:ff
    inet 172.17.43.1/24 scope global docker0
       valid_lft forever preferred_lft forever
17: br0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue master docker0 state UNKNOWN
    link/ether ba:89:14:e0:7f:43 brd ff:ff:ff:ff:ff:ff
[root@docker128 ~]# ip route
default via 192.168.18.2 dev eth0  proto static  metric 100
172.17.0.0/16 dev docker0  scope link
172.17.43.0/24 dev docker0  proto kernel  scope link  src 172.17.43.1
192.168.18.0/24 dev eth0  proto kernel  scope link  src 192.168.18.128
192.168.18.0/24 dev eth0  proto kernel  scope link  src 192.168.18.128  metric 100
```

# OVS +Docker

```
[root@docker128 ~]# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN
    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
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP qlen 1000
    link/ether 00:0c:29:e8:02:c7 brd ff:ff:ff:ff:ff:ff
    inet 192.168.18.128/24 brd 192.168.18.255 scope global dynamic eth0
       valid_lft 1126sec preferred_lft 1126sec
3: ovs-system: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN
    link/ether e2:e2:97:83:c6:35 brd ff:ff:ff:ff:ff:ff
16: docker0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP
    link/ether ba:89:14:e0:7f:43 brd ff:ff:ff:ff:ff:ff
    inet 172.17.43.1/24 scope global docker0
       valid_lft forever preferred_lft forever
17: br0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue master docker0 state UNKNOWN
    link/ether ba:89:14:e0:7f:43 brd ff:ff:ff:ff:ff:ff
[root@docker128 ~]# ping 172.17.42.1
PING 172.17.42.1 (172.17.42.1) 56(84) bytes of data.
64 bytes from 172.17.42.1: icmp_seq=1 ttl=64 time=3.61 ms
64 bytes from 172.17.42.1: icmp_seq=2 ttl=64 time=1.37 ms
```

```
[root@docker131 ~]# tshark -i br0 -R ip proto gre
tshark: -R without -2 is deprecated. For single-pass filtering use -Y.
Running as user "root" and group "root". This could be dangerous.
Capturing on 'br0'
^C
```

没有报文

```
[root@docker128 ~]# tshark -i eth0 ip proto gre
Running as user "root" and group "root". This could be dangerous.
Capturing on 'eth0'
  1    0.000000  172.17.43.1 -> 172.17.42.1  ICMP 136 Echo (ping) request  id=0x10db, seq=106/27136, ttl=64
  2    0.001602  172.17.42.1 -> 172.17.43.1  ICMP 136 Echo (ping) reply    id=0x10db, seq=106/27136, ttl=64 (request in 1)
```

# Thanks

## FAQ时间