Macvlan and IPvlan basics

May 29, 2016 Containers, Docker, ipvlan, macvlan

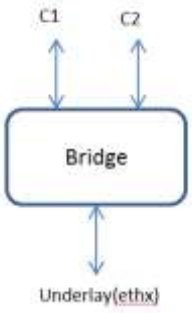
Macvlan and ipvlan are Linux network drivers that exposes underlay or host interfaces directly to VMs or Containers running in the host. In this blog, I will cover basics of macvlan and ipvlan, compare macvlan and ipvlan to Linux bridge and sub-interfaces and also show how to create these interfaces in Linux system. In the next set of blogs, I will cover how macvlan and ipvlan interfaces are used in Docker and CoreOS.

VM and Container networking

When running a baremetal server, host networking can be straightforward with few ethernet interfaces and a default gateway providing external connectivity. When we run multiple VMs in a host, it is needed to provide connectivity between VMs within the host and across hosts. On an average, the number of VMs in a single host does not exceed 15-20. When running Containers in a host, the number of Containers in a single host can easily exceed 100. It is needed to have sophisticated mechanism to interconnect Containers. Broadly, there are two ways for Containers or VMs to communicate to each other. In Underlay network approach, VMs or Containers are directly exposed to host network. Bridge, macvlan and ipvlan network drivers are examples of this approach. In Overlay network approach, there is an additional level of encapsulation like VXLAN, NVGRE between the Container/VM network and the underlay network.

Linux Bridge

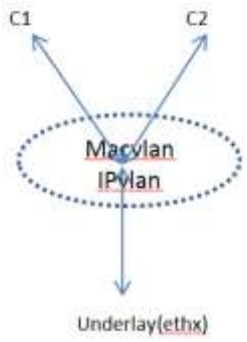
Linux Bridge acts like a regular hardware switch with learning and also supports protocols like STP for loop prevention. In linux bridge implementation, VMs or Containers will connect to bridge and bridge will connect to outside world. For external connectivity, we would need to use NAT. The following picture shows 2 Containers connected to a Linux bridge with ethx interface providing external connectivity.



Macvlan

Macvlan allows a single physical interface to have multiple mac and ip addresses using macvlan sub-interfaces. This is different from creating sub-interface on a physical interface using vlan. With vlan sub-interfaces, each sub-interface belongs to a different L2 domain using vlan and all sub-interfaces have same mac address. With macvlan, each sub-interface will get unique mac and ip address and will be exposed

directly in underlay network. Macvlan interface is typically used for virtualization applications and each macvlan interface is connected to a Container or VM. Each container or VM can directly get dhcp address from a common server as the host would do. This would help customers who want Containers to be part of their traditional network with the IP addressing scheme that they already have. Macvlan has 4 types(Private, VEPA, Bridge, Passthru). Commonly used type is Macvlan bridge which allows for endpoints in a single host to be able to talk to each other without packets leaving the host. For external connectivity, underlay network is used. Following picture shows two Containers using macvlan bridge to communicate with each other as well as to outside world. Both Containers will directly get exposed in underlay network using Macvlan sub-interfaces.



Following is an example for creating 2 macvlan bridge sub-interfaces. These examples were tried on Ubuntu 14.04.

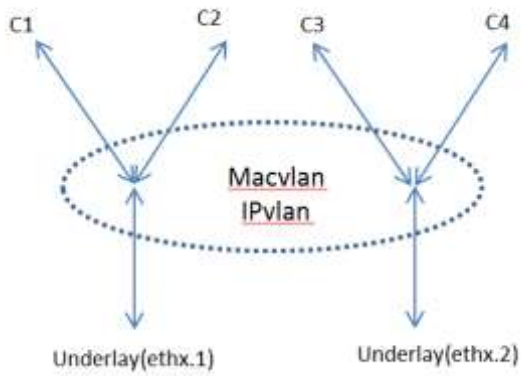
```
sudo ip link add mymacvlan1 link eth0 type macvlan mode bridge
sudo ip link add mymacvlan2 link eth0 type macvlan mode bridge
sudo ifconfig mymacvlan1 up
sudo ifconfig mymacvlan2 up
```

Following output shows the interface related details for parent interface and the two macvlan sub-interfaces. As we can see, the two macvlan sub-interfaces get unique mac address that is different from parent interface eth0.

```
$ ip link show mymacvlan1
19: mymacvlan1@eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc
noqueue state UNKNOWN mode DEFAULT group default
    link/ether 7e:33:b3:7d:f9:80 brd ff:ff:ff:ff:ff:ff
$ ip link show mymacvlan2
20: mymacvlan2@eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc
noqueue state UNKNOWN mode DEFAULT group default
    link/ether de:1a:30:bc:a3:a0 brd ff:ff:ff:ff:ff:ff
$ ifconfig eth0
```

```
eth0      Link encap:Ethernet  HWaddr 08:00:27:85:47:0a
          inet addr:192.168.0.101  Bcast:192.168.0.255
Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe85:470a/64 Scope:Link
```

Following picture shows an example where macvlan sub-interface works together with vlan sub-interface. Containers c1, c2 are connected to underlay interface ethx.1 and Containers c3, c4 are connected to underlay interface ethx.2.



Following commands creates 2 vlan sub-interfaces and creates two macvlan bridge subinterface on each vlan sub-interface as shown in the picture above:

```
sudo vconfig add eth2 10
sudo vconfig add eth2 20

macvlan with subinterface:
sudo ip link add mymacvlan1 link eth2.10 type macvlan mode bridge
sudo ip link add mymacvlan2 link eth2.10 type macvlan mode bridge
sudo ifconfig mymacvlan1 up
sudo ifconfig mymacvlan2 up

sudo ip link add mymacvlan3 link eth2.20 type macvlan mode bridge
sudo ip link add mymacvlan4 link eth2.20 type macvlan mode bridge
sudo ifconfig mymacvlan3 up
sudo ifconfig mymacvlan4 up
```

Following output shows the relevant interface details:

```
$ ip link show eth2.10
24: eth2.10@eth2: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc
```

```
noqueue state UP mode DEFAULT group default
    link/ether 08:00:27:32:b9:d3 brd ff:ff:ff:ff:ff:ff
smakam14@myubuntu:~$ ip link show eth2.20
25: eth2.20@eth2: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc
noqueue state UP mode DEFAULT group default
    link/ether 08:00:27:32:b9:d3 brd ff:ff:ff:ff:ff:ff
smakam14@myubuntu:~$ ip link show mymacvlan1
26: mymacvlan1@eth2.10: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc
noqueue state UNKNOWN mode DEFAULT group default
    link/ether 26:2a:eb:a8:19:18 brd ff:ff:ff:ff:ff:ff
smakam14@myubuntu:~$ ip link show mymacvlan2
27: mymacvlan2@eth2.10: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc
noqueue state UNKNOWN mode DEFAULT group default
    link/ether 3e:a4:35:93:90:7c brd ff:ff:ff:ff:ff:ff
smakam14@myubuntu:~$ ip link show mymacvlan3
28: mymacvlan3@eth2.20: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc
noqueue state UNKNOWN mode DEFAULT group default
    link/ether 3e:07:83:fc:72:2a brd ff:ff:ff:ff:ff:ff
smakam14@myubuntu:~$ ip link show mymacvlan4
29: mymacvlan4@eth2.20: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc
noqueue state UNKNOWN mode DEFAULT group default
    link/ether fe:92:b8:50:ef:7b brd ff:ff:ff:ff:ff:ff
```

Comparing Macvlan to Bridge

Bridges can be used for complex scenarios where it is needed to provide connectivity across hosts and be able to run L2 protocols on the bridge. Macvlan is used for more simple scenarios where it is needed to expose endpoints directly to the external network. With Bridge, it is needed to use NAT for external connectivity. With macvlan, since endpoints are directly mapped to underlay network, external connectivity can be done using same dhcp server and switch that the host uses.

ipvlan

ipvlan is similar to macvlan with the difference being that the endpoints have the same mac address. ipvlan supports L2 and L3 mode. In ipvlan l2 mode, each endpoint gets the same mac address but different ip address. In ipvlan l3 mode, packets are routed between endpoints, so this gives better scalability.

Trying out ipvlan:

ipvlan need kernel support > 4.2. I tried out ipvlan in Ubuntu 16.04.

Following commands sets up two ipvlan l2 sub-interfaces:

```
sudo ip link add myipvlan1 link enp0s3 type ipvlan mode l2
sudo ifconfig myipvlan1 up
sudo ip link add myipvlan2 link enp0s3 type ipvlan mode l2
sudo ifconfig myipvlan2 up
```

Following output shows the interface related outputs for the ipvlan interfaces. As we can see, mac address is same for both ipvlan sub-interface.

```
$ ip link show
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast
state UP
mode DEFAULT group default qlen 1000
    link/ether 08:00:27:79:da:82 brd ff:ff:ff:ff:ff:ff
3: myipvlan1@enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc
noqueue st
ate UNKNOWN mode DEFAULT group default qlen 1000
    link/ether 08:00:27:79:da:82 brd ff:ff:ff:ff:ff:ff
4: myipvlan2@enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc
noqueue st
ate UNKNOWN mode DEFAULT group default qlen 1000
    link/ether 08:00:27:79:da:82 brd ff:ff:ff:ff:ff:ff
```

Comparing ipvlan to macvlan

ipvlan should be used in cases where some switches restrict the maximum number of mac address per physical port due to port security configuration. macvlan needs to be used in cases where common dhcp server is used since dhcp server would need unique mac address which ipvlan does not have.

References:

- Bridge vs Macvlan
- Macvlan vs ipvlan
- Configuring macvlan and ipvlan Linux networking