Contiv datapath troubleshooting

Q&A: How can I found out which mode Contiv is running?

Since CVD is using VLAN and Bridge mode, the first thing to verify is the Contiv global config mode:

```
root@ubuntu:/home/kalei# netctl global info
Fabric mode: default
Forward mode: bridge
ARP mode: proxy
Vlan Range: 1-4094
Vxlan range: 1-10000
Private subnet: 172.19.0.0/16
```

Whereas the Forward mode "bridge" is the right mode to use

Q&A: How can I use the ovs tool set to query the OVS datapath?

By default, the host doesn't have the ovs tool set (e.g. ovs-vsctl, etc) installed. The ovs tool set can be found along with the v2plugin container itself.

```
docker-runc exec <CONTAINER_ID> ovs-vsctl show
```

For example:

```
root@ubuntu:/home/kalei# docker-runc exec
2bc6bd40b7582840583966433ee0982b33263b80ef74711dbf2ce655c66a995c
ovs-vsctl show
b9aa74be-5649-46ee-8427-6712b5df5dae
    Manager "ptcp:6640"
    Bridge contivVxlanBridge
        Controller "tcp:127.0.0.1:6633"
            is_connected: true
        fail_mode: secure
        Port "vxif19216858105"
            Interface "vxif19216858105"
                type: vxlan
                options: {dst_port="8472", key=flow,
remote_ip="192.168.58.105", tos=inherit}
        Port "contivh0"
            tag: 2
            Interface "contivh0"
                type: internal
    Bridge contivVlanBridge
        Controller "tcp:127.0.0.1:6634"
            is_connected: true
        fail_mode: secure
        Port "enp0s10"
            Interface "enp0s10"
```

Whereas the **CONTAINER_ID** is v2plugin container where you can find it from the docker-runc list

Q&A: How can I find out which OVS bridge Contiv is going to program?

Contiv relies on two main OVS bridges: **contivVlanBridge** and **contivVxlanBridge**

Contiv will add OVS port on contivVlanBridge when user creates Container with VLAN network. Similarly, when user creates Container with VXLAN network, Contiv will add OVS port on contivVxlanBridge instead.

```
root@ubuntu:/home/kalei# docker-runc exec
2bc6bd40b7582840583966433ee0982b33263b80ef74711dbf2ce655c66a995c
ovs-vsctl show
2c88c724-878a-4e11-8403-998adcc2ecf9
    Manager "ptcp:6640"
    Bridge contivVlanBridge
        Controller "tcp:127.0.0.1:6634"
            is_connected: true
        fail_mode: secure
        Port uplinkPort
            Interface "eth2"
            Interface "eth3"
    Bridge contivVxlanBridge
        Controller "tcp:127.0.0.1:6633"
            is_connected: true
        fail_mode: secure
        Port "contivh0"
            tag: 2
            Interface "contivh0"
                type: internal
    ovs_version: "2.5.0"
```

Q&A: How can I correlate the OVS ports with Container ports?

Suppose user creates a container, an OVS port will be created:

```
root@ubuntu:/home/kalei# docker-runc exec
2bc6bd40b7582840583966433ee0982b33263b80ef74711dbf2ce655c66a995c
ovs-vsctl show
2c88c724-878a-4e11-8403-998adcc2ecf9
    Manager "ptcp:6640"
    Bridge contivVlanBridge
        Controller "tcp:127.0.0.1:6634"
            is_connected: true
        fail_mode: secure
        Port "vvport2"
            tag: 1
            Interface "vvport2"
        Port uplinkPort
            Interface "eth2"
            Interface "eth3"
```

Where as vvport2 will be the veth pair end connect the In you have multiple containers, there will be multiple vvport under the same contivVlanBridge.

In order to identify which vvport connect to which container, you can get the interface ID of the host and OVS port:

```
root@ubuntu:/home/kalei# docker exec -it c2 ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN
group default qlen 1
    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
16: eth0@if15: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1450 qdisc noqueue
state UP group default
    link/ether 02:02:0a:00:00:02 brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet 10.0.0.2/24 scope global eth0
       valid_lft forever preferred_lft forever
    inet6 fe80::2:aff:fe00:2/64 scope link
       valid_lft forever preferred_lft forever
```

And the OVS port will be the next one of the docker interface name (that is, if16 in the above example)

```
root@ubuntu:/home/kalei# ip a
15: vvport2@if16: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc
noqueue master ovs-system state UP
    link/ether 0e:01:b0:de:3b:93 brd ff:ff:ff:ff:ff:ff link-netnsid 2
    inet6 fe80::c01:b0ff:fede:3b93/64 scope link
       valid_lft forever preferred_lft forever
```

Q&A: I dumped the ovs flow but it shows nothing

Contiv supports policy which internally translated into OVS flows. You can dump the flow tables with the following command (Keep in mind you have to use OPENFLOW13):

```
root@ubuntu:/home/kalei# docker-runc exec
2bc6bd40b7582840583966433ee0982b33263b80ef74711dbf2ce655c66a995c
ovs-ofctl dump-flows contivVlanBridge -O OPENFLOW13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x20, duration=5665.396s, table=0, n_packets=0, n_bytes=0,
priority=102,udp,in_port=1,tp_dst=53 actions=goto_table:1
 cookie=0x22, duration=5665.395s, table=0, n_packets=0, n_bytes=0,
priority=102,udp,in_port=2,tp_dst=53 actions=goto_table:1
 cookie=0x1e, duration=5667.304s, table=0, n_packets=0, n_bytes=0,
priority=101,udp,dl_vlan=4093,dl_src=02:02:00:00:00:00/ff:ff:00:00:00:00
,tp_dst=53 actions=pop_vlan,goto_table:1
 cookie=0x1c, duration=5667.304s, table=0, n_packets=0, n_bytes=0,
priority=100,arp,arp_op=1 actions=CONTROLLER:65535
 cookie=0x1d, duration=5667.304s, table=0, n_packets=0, n_bytes=0,
priority=100,udp,dl_src=02:02:00:00:00:00/ff:ff:00:00:00:00,tp_dst=53
actions=CONTROLLER:65535
 cookie=0x1a, duration=5667.304s, table=0, n_packets=9, n_bytes=738,
priority=1 actions=goto_table:1
 cookie=0x21, duration=5665.396s, table=1, n_packets=0, n_bytes=0,
priority=100,in_port=1 actions=goto_table:6
 cookie=0x23, duration=5665.395s, table=1, n_packets=0, n_bytes=0,
priority=100,in_port=2 actions=goto_table:6
 cookie=0x26, duration=4787.918s, table=1, n_packets=8, n_bytes=648,
priority=10,in_port=4
actions=write_metadata:0x100000000/0xff00000000,goto_table:3
 cookie=0x1b, duration=5667.304s, table=1, n_packets=0, n_bytes=0,
priority=1 actions=goto_table:4
 cookie=0x19, duration=5667.304s, table=3, n_packets=9, n_bytes=738,
priority=1 actions=goto_table:4
 cookie=0x27, duration=4787.918s, table=4, n_packets=0, n_bytes=0,
priority=100,ip,metadata=0x100000000/0xff00000000,nw_dst=10.0.0.2
actions=write_metadata:0/0xfffe,goto_table:5
 cookie=0x17, duration=5667.304s, table=4, n_packets=9, n_bytes=738,
priority=1 actions=goto_table:5
 cookie=0x18, duration=5667.304s, table=5, n_packets=9, n_bytes=738,
priority=1 actions=goto_table:6
 cookie=0x16, duration=5667.304s, table=6, n_packets=9, n_bytes=738,
priority=1 actions=goto_table:9
 cookie=0x1f, duration=5667.304s, table=9, n_packets=9, n_bytes=738,
priority=1 actions=NORMAL
```

The flows are distributed across multiple Openflow tables (from table 0 to table 9)

Usually most of the troubleshooting can

Q&A: How does Contiv handle ARP request?

Contiv also handle ARP request thru ARP proxy mechanism. The following link describe how Contiv handles ARP request:

https://cisco.jiveon.com/people/kalei/blog/2017/11/07/contiv-l2-vlan-packet-flows

Q&A: How can I find out the container network namespace?

By default, container namespace is hidden. Run the following command to inspect the container network namespace

```
pid=$(docker inspect -f '{{.State.Pid}}' ${container_id})
mkdir -p /var/run/netns/
ln -sfT /proc/$pid/ns/net /var/run/netns/$container_id
```