Team:

Khantil Choksi - khchoksi	Shubhankar Reddy - skatta2

Staging Server IP			VM IP		
152.14.83.156	ece792	EcE792net!	192.168.124.15	ece792	EcE792net!

Index:

Problem 1:

Note: All the code and readme is inside q1 folder.
Readme file attached q1/Q1_README.txt

```
ece792@ece792-Standard-PC-i440FX-PIIX-1996:/etc/libvirt/qemu$ virsh domifaddr khchoksi4
 Name       MAC address          Protocol     Address
-------------------------------------------------------------------------------
 vnet5      52:54:00:62:24:60    ipv4         192.168.118.228/24
 vnet6      52:54:00:10:37:b6    ipv4         192.168.100.28/24
```

```
ece792@ece792-Standard-PC-i440FX-PIIX-1996:/etc/libvirt/qemu$ virsh domifaddr khchoksiq3
 Name       MAC address          Protocol     Address
-------------------------------------------------------------------------------
 vnet3      52:54:00:d5:61:94    ipv4         192.168.122.41/24
 vnet4      52:54:00:10:37:b6    ipv4         192.168.100.28/24
```

```
ece792@ece792-Standard-PC-i440FX-PIIX-1996:~/Downloads$ sudo python q1_1.py
PART 1 : All MACs and IPs
Domain name khchoksi
IP:     192.168.100.5
IP:     192.168.122.109
MAC:    52:54:00:01:f0:cc
MAC:    52:54:00:b9:0d:c7
MAC:    52:54:00:d1:66:f6


Domain name khchoksi4
IP:     192.168.100.28
MAC:    52:54:00:62:24:60
MAC:    52:54:00:10:37:b6


Domain name khchoksiq3
IP:     192.168.100.28
MAC:    52:54:00:d5:61:94
MAC:    52:54:00:10:37:b6


Domain name khchoksi5
MAC:    52:54:00:41:19:16
MAC:    52:54:00:a0:70:0a


PART 2 : List of conflicting MAC addresses
Conflicting mac found for khchoksiq3 with mac 52:54:00:10:37:b6
```

```
ece792@ece792-Standard-PC-i440FX-PIIX-1996:~/Downloads$ sudo python q1_1.py
PART 1 : All MACs and IPs
Domain name khchoksi
IP:     192.168.100.5
IP:     192.168.122.109
MAC:    52:54:00:01:f0:cc
MAC:    52:54:00:b9:0d:c7
MAC:    52:54:00:d1:66:f6


Domain name khchoksi4
IP:     192.168.100.28
MAC:    52:54:00:62:24:60
MAC:    52:54:00:10:37:b6


Domain name khchoksiq3
IP:     192.168.100.28
MAC:    52:54:00:d5:61:94
MAC:    52:54:00:10:37:b6


Domain name khchoksi5
MAC:    52:54:00:41:19:16
MAC:    52:54:00:a0:70:0a


PART 2 : List of conflicting MAC addresses
Conflicting mac found for khchoksiq3 with mac 52:54:00:10:37:b6
PART 3 : Resolving MAC conflicts
```

```
PART 4 - After resolving conflicts
Domain name khchoksi
IP:     192.168.100.5
IP:     192.168.122.109
IP:     192.168.100.5
IP:     192.168.122.109
MAC:    52:54:00:01:f0:cc
MAC:    52:54:00:b9:0d:c7
MAC:    52:54:00:d1:66:f6


Domain name khchoksi4
IP:     192.168.100.28
IP:     192.168.100.28
MAC:    52:54:00:62:24:60
MAC:    52:54:00:10:37:b6


Domain name khchoksiq3
IP:     192.168.100.28
IP:     192.168.100.28
MAC:    52:54:00:10:37:b6
MAC:    52:54:00:d5:61:94
MAC:    52:54:00:10:37:b6


Domain name khchoksi5
MAC:    52:54:00:41:19:16
MAC:    52:54:00:a0:70:0a
```

After solving conflicts:

```
ece792@ece792-Standard-PC-i440FX-PIIX-1996:~$ virsh domiflist khchoksiq3
Interface  Type       Source     Model       MAC
-------------------------------------------------------
vnet3      network    default    virtio      52:54:00:d5:61:94
vnet4      network    khchoksiNETWORK3 virtio      52:54:00:da:ef:c3

ece792@ece792-Standard-PC-i440FX-PIIX-1996:~$ virsh domiflist khchoksi4
Interface  Type       Source     Model       MAC
-------------------------------------------------------
vnet5      network    private_net1 virtio      52:54:00:62:24:60
vnet6      network    khchoksiNETWORK3 virtio      52:54:00:10:37:b6
```

Reference:

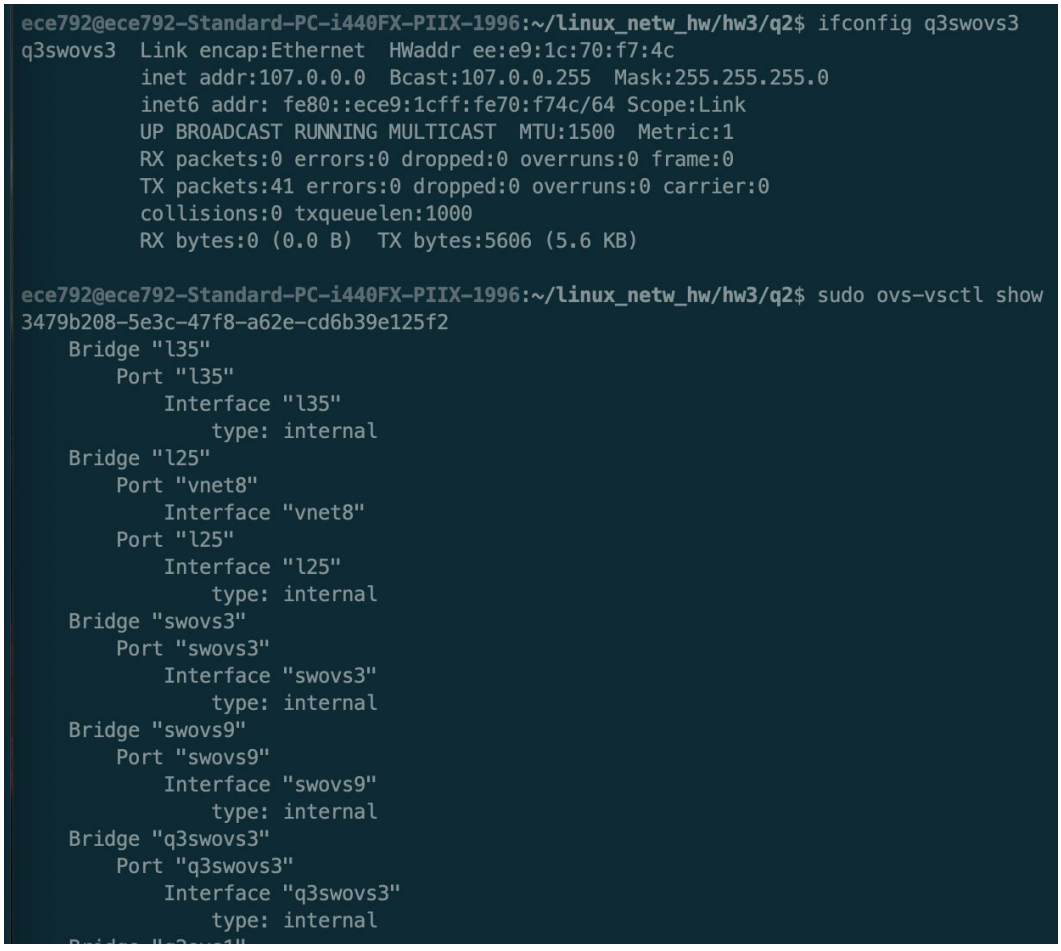
- https://wiki.libvirt.org/page/VirtualNetworking
- https://stackoverflow.com/questions/20499074/run-local-python-script-on-remote-server
-

Problem 2:

Note: All the code and readme is inside q2 folder.
Readme file attached q2/Q2_README.txt
Example:
For following network (defined in networks_vars.yml):

```
---
networks:
 l32:
   network_name: l32
   bridge_name: q3ovs3
   bridge_type: routed
   ip_address: 109.0.0.0/24
```

Following is the screenshot showing OVS bridge created in routed mode:



It has also added route:

```
ece792@ece792-Standard-PC-i440FX-PIIX-1996:~$ route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         192.168.124.1   0.0.0.0         UG    100    0        0 ens3
0.0.0.0         192.168.123.1   0.0.0.0         UG    101    0        0 ens5
29.0.0.0        0.0.0.0         255.255.255.0   U     0      0        0 ens4
54.0.0.0        0.0.0.0         255.255.255.0   U     0      0        0 veth13
107.0.0.0       0.0.0.0         255.255.255.0   U     0      0        0 q3swovs3
117.0.0.0       29.0.0.4        255.255.255.0   UG    0      0        0 ens4
```

References:

- https://jamielinux.com/docs/libvirt-networking-handbook/routed-network.html
- https://docs.ansible.com/ansible/2.4/xml_module.html
- https://docs.ansible.com/ansible/2.5/modules/lineinfile_module.html

Problem 3:

Design 1: All VMs from all tenants are connected to same bridge (in bridge mode).
Hypervisor 1:
sudo ip route add 192.168.12.0/24(Tenant subnet) via 192.168.123.66(Hypervisor's IP)
sudo ip route add 192.168.11.0/24 () via 192.168.123.123 (Hypervisor's IP)

Configuration:
Guest VM Config:

```
[root@localhost ~]# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         192.168.122.1   0.0.0.0         UG    0      0        0 eth0
19.0.0.0        0.0.0.0         255.255.255.0   U     0      0        0 eth1
192.168.122.0   0.0.0.0         255.255.255.0   U     0      0        0 eth0
[root@localhost ~]# ifconfig eth1
eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 19.0.0.2  netmask 255.255.255.0  broadcast 19.0.0.255
        ether 52:54:00:01:2a:60  txqueuelen 1000  (Ethernet)
        RX packets 2077  bytes 356490 (348.1 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 215  bytes 33474 (32.6 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Hypervisor:
The ens4 interface is connected to OVS bridge swovs2.

```
ece792@ece792-Standard-PC-i440FX-PIIX-1996:~$ sudo ovs-vsctl show
3479b208-5e3c-47f8-a62e-cd6b39e125f2
    Bridge "swovs3"
        Port "swovs3"
            Interface "swovs3"
                type: internal
    Bridge "swovs9"
        Port "swovs9"
            Interface "swovs9"
                type: internal
    Bridge "swovs2"
        Port "vnet11"
            Interface "vnet11"
        Port "swovs2"
            Interface "swovs2"
                type: internal
        Port "vnet12"
            Interface "vnet12"
        Port "ens4"
            Interface "ens4"
```

```
[root@localhost ~]# ping 19.0.0.1
PING 19.0.0.1 (19.0.0.1) 56(84) bytes of data.
64 bytes from 19.0.0.1: icmp_seq=1 ttl=64 time=5.08 ms
64 bytes from 19.0.0.1: icmp_seq=2 ttl=64 time=1.26 ms
64 bytes from 19.0.0.1: icmp_seq=3 ttl=64 time=1.44 ms
64 bytes from 19.0.0.1: icmp_seq=4 ttl=64 time=1.69 ms
^C
--- 19.0.0.1 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3005ms
rtt min/avg/max/mdev = 1.269/2.371/5.080/1.571 ms
```

1. **Design 1: All VMs from all tenants are connected to same bridge (in bridge mode).**
   a. **What are the disadvantages for tenants? Is a tenant's traffic isolated from other tenants?**
      i. All the tenants' VMs will be in the same subnet inside one hypervisor.
      ii. It will not provide the security and isolated environment to the packets sent by one tenant. e.g. The VM of tenant1 will be able to ping and send packets to VM of tenant2 inside the same hypervisor. So, it will violate the isolated traffic promise for each tenant.
   b. **What, if anything, breaks if two tenants in the same hypervisor host use the same IP address?**
      i. Explanation:
      We have configured other team's VMs to be of same IP address 19.0.0.1
      The ping is successful as one of the VMs on the other end responds to the ARP reply.
      We lost connectivity to the VM with duplicate IP, unless we flush the ARP tables, and tried the ping again.
      - Consider that we have following configuration:

	Hypervisor1	Hypervisor2
VM1	Tenant1 (19.0.0.2)	Tenant1 (19.0.0.1)
VM2	Tenant2 (19.0.0.4)	Tenant2 (19.0.0.1)

Below is the tcpdump capture when we tried to ping the other team (with duplicate IPs).

```
23:57:00.340229 IP 19.0.0.2 > 19.0.0.1: ICMP echo request, id 2308
, seq 3, length 64
23:57:00.343062 IP 19.0.0.1 > 19.0.0.2: ICMP echo reply, id 2308,
seq 3, length 64
```

Screenshot for `ifconfig on eth1`, of tenant 1 on our side (Hypervisor 1)

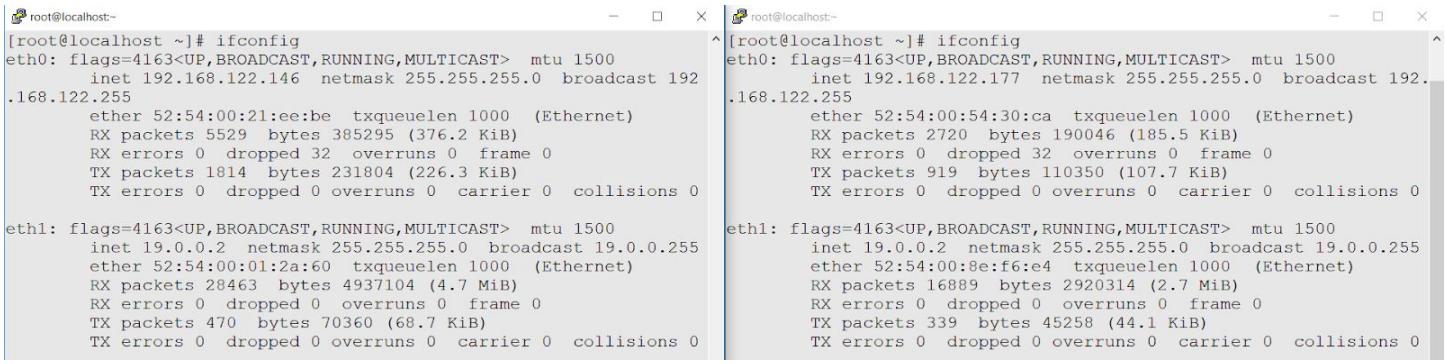
```
[root@localhost ~]# ifconfig eth1
eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 19.0.0.4  netmask 255.255.255.0  broadcast 19.0.0.255
        ether 52:54:00:8e:f6:e4  txqueuelen 1000  (Ethernet)
        RX packets 12206  bytes 2105056 (2.0 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 327  bytes 41154 (40.1 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Consider the above scenario, where the main issue over here is, the ping request from tenant2 - hypervisor1 - vm2 will not be able to decide whether it pinging to tenant2-hypervisor2-VM2 or tenant1-hypervisor2-VM1. So this is the drawback of Design 1 of attaching all VMs to single bridge.

- Now, we have configured as following:

	Hypervisor1	Hypervisor2
VM1	Tenant1 (19.0.0.2)	Tenant1 (19.0.0.1)
VM2	Tenant2 (19.0.0.2)	Tenant2 (19.0.0.3)

Below is the screenshot when we configured duplicate IPs on our (hypervisor 1) VMs.



Below is the wireshark (tshark) capture of the ping request and reply from our VM.

```
19.0.0.1        52:54:00:6c:17:ca      19.0.0.2        52:54:00:01:2a:60
19.0.0.2        52:54:00:01:2a:60      19.0.0.1        52:54:00:6c:17:ca
```

Conclusion: So L2 datapath from 3->2 breaks if the VMs with same IPs are on our hypervisor, datapath 8->9 breaks if they are on our peer hypervisor.

c. What, if anything, breaks if two tenants in a different hypervisor host use the same IP address?

We have configured as following:

	Hypervisor1	Hypervisor2
VM1	Tenant1 (19.0.0.3)	Tenant1 (19.0.0.1)
VM2	Tenant2 (19.0.0.2)	Tenant2 (19.0.0.3)

From hypervisor1 VM2 tenant2 is trying to ping to Hypervisor2 Tenant2 **(**his / her own VM)
$ ping 19.0.0.3
Datapath from (2) to onwards will not work as ARP finds the IP 19.0.0.3 in the same hypervisor.

i. Connection to the tenant2's VM on hypervisor 2 is lost.
ii. Below is the tcpdump of ping on the ens4 of hypervisor1, as we can observe, there are no ping packets from 19.0.0.2 exiting hypervisor1. As the switch has mapped its MAC address to tenant1's VM1 which is connected to the same ovs switch.

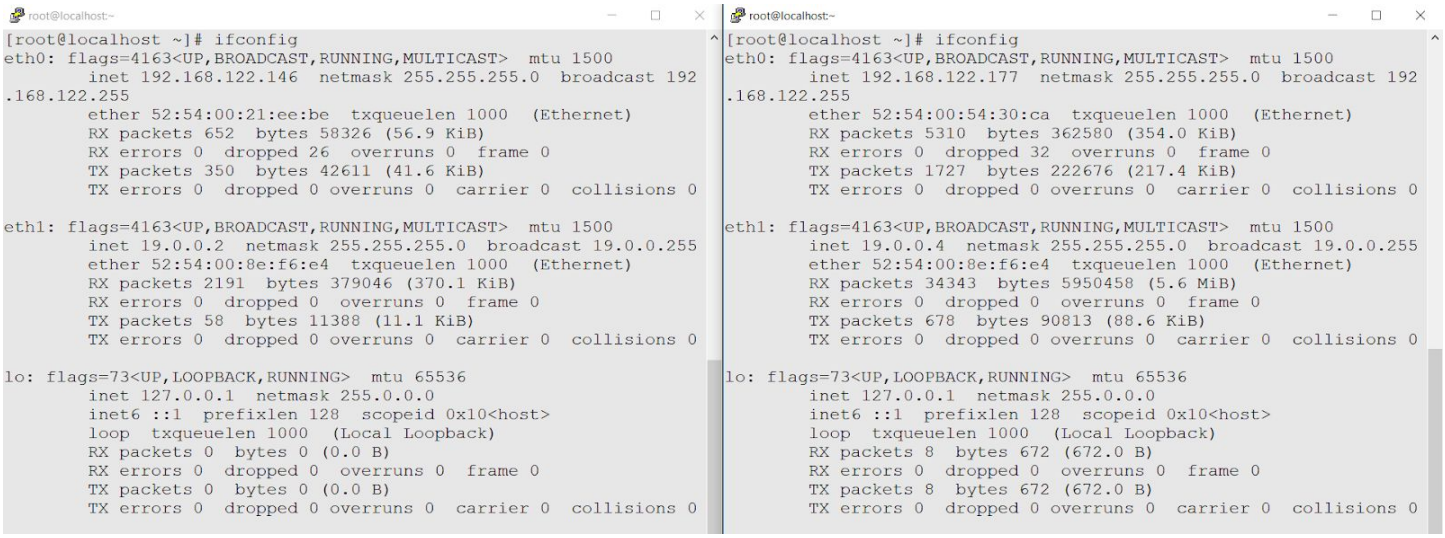
Conclusion: So L3 datapath 3->4 breaks as the ARP is resolved inside switch2 and the packets are transmitted to the VM of tenant2 from VM1 of tenant1 in the same hypervisor.

```
ece792@ece792-Standard-PC-i440FX-PIIX-1996:~$ sudo tcpdump -i ens4 | grep "ICMP"
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on ens4, link-type EN10MB (Ethernet), capture size 262144 bytes
00:35:23.501311 IP6 :: > ff02::16: HBH ICMP6, multicast listener report v2, 2 group record(s), length 48
00:35:23.525339 IP6 :: > ff02::16: HBH ICMP6, multicast listener report v2, 2 group record(s), length 48
00:35:24.468810 IP6 :: > ff02::16: HBH ICMP6, multicast listener report v2, 2 group record(s), length 48
00:35:24.780795 IP6 :: > ff02::16: HBH ICMP6, multicast listener report v2, 2 group record(s), length 48
00:35:28.902520 IP6 :: > ff02::16: HBH ICMP6, multicast listener report v2, 1 group record(s), length 28
00:35:28.903170 IP6 :: > ff02::1:ff53:156e: ICMP6, neighbor solicitation, who has fe80::5054:ff:fe53:156e, length 24
00:35:29.583357 IP6 :: > ff02::16: HBH ICMP6, multicast listener report v2, 1 group record(s), length 28
00:35:29.903407 IP6 fe80::5054:ff:fe53:156e > ff02::16: HBH ICMP6, multicast listener report v2, 1 group record(s), length 28
00:35:29.905421 IP6 fe80::5054:ff:fe53:156e > ip6-allrouters: ICMP6, router solicitation, length 8
```

d. What, if anything, breaks if two tenants in the same hypervisor host use the same MAC address?

Consider the following configuration:

	Hypervisor1	Hypervisor2
VM1	Tenant1 (19.0.0.2) (52:54:00:8e:f6:e4)	Tenant1 (19.0.0.1)
VM2	Tenant2 (19.0.0.4) (52:54:00:8e:f6:e4)	Tenant2 (19.0.0.3)



The above screenshot displays the configuration on VMs on Hypervisor1
Ping to 19.0.0.4 from VM1 of hypervisor1 will fail.
L2 Datapath from (2) to VM2 fails.

- Now, consider the scenario that, having same MAC address on Hypervisor2 and VM from Hypervisor1 wants to ping to VMs on Hypervisor2. Following is the screenshot for that:

	Hypervisor1	Hypervisor2
VM1	Tenant1 (19.0.0.2)	Tenant1 (19.0.0.1) (52:54:00:6c:17:ca)
VM2	Tenant2 (19.0.0.4)	Tenant2 (19.0.0.3) (52:54:00:6c:17:ca)

```
[root@localhost ~]# ping 19.0.0.3
PING 19.0.0.3 (19.0.0.3) 56(84) bytes of data.
64 bytes from 19.0.0.3: icmp_seq=19 ttl=64 time=2.00 ms
64 bytes from 19.0.0.3: icmp_seq=20 ttl=64 time=1.53 ms
^C
--- 19.0.0.3 ping statistics ---
20 packets transmitted, 2 received, 90% packet loss, time 19005ms
rtt min/avg/max/mdev = 1.532/1.770/2.009/0.242 ms
[root@localhost ~]# ping 19.0.0.1
PING 19.0.0.1 (19.0.0.1) 56(84) bytes of data.
^C
--- 19.0.0.1 ping statistics ---
10 packets transmitted, 0 received, 100% packet loss, time 9002ms

[root@localhost ~]# ping 19.0.0.1
PING 19.0.0.1 (19.0.0.1) 56(84) bytes of data.
64 bytes from 19.0.0.1: icmp_seq=1 ttl=64 time=2.08 ms
64 bytes from 19.0.0.1: icmp_seq=2 ttl=64 time=1.54 ms
^C
--- 19.0.0.1 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1001ms
rtt min/avg/max/mdev = 1.541/1.813/2.086/0.275 ms
[root@localhost ~]# ping 19.0.0.3
PING 19.0.0.3 (19.0.0.3) 56(84) bytes of data.
^C
--- 19.0.0.3 ping statistics ---
3 packets transmitted, 0 received, 100% packet loss, time 2000ms

[root@localhost ~]# ping 19.0.0.3
PING 19.0.0.3 (19.0.0.3) 56(84) bytes of data.
64 bytes from 19.0.0.3: icmp_seq=1 ttl=64 time=4.59 ms
```

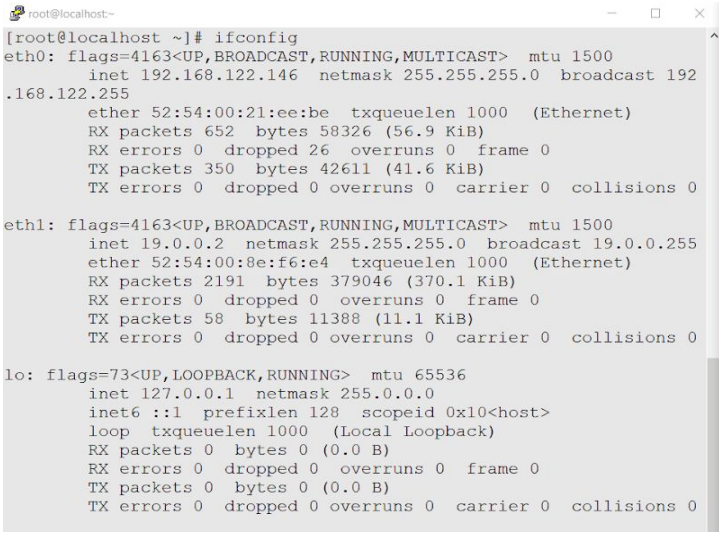
We observed intermediate connectivity issues while trying to ping from vm on hypervisor1 to another vm on hypervisor2. Both had connectivity individually, but while switching the ping from one destination vm to another, there was a momentary loss of connectivity. We believe that the datapath from 9 to 10 breaks, as 2 same MAC addresses cannot be mapped to different ports on a switch.

Conclusion: L2 datapath from 9->10 breaks. (assuming we are trying to communicate with VM2 in hypervisor 2 from VM2 in hypervisor1, and VM1 and VM2 in hypervisor2 have the same MAC). VM2 in hypervisor 1 will be able to communicate with either of the VMs but not both simultaneously.(The switch takes time to update its MAC table entries)

e. What, if anything, breaks if two tenants in a different hypervisor host use the same MAC address?

	Hypervisor1	Hypervisor2
VM1	Tenant1 (19.0.0.2)	Tenant1 (19.0.0.1) (52:54:00:6c:17:ca)
VM2	Tenant2 (19.0.0.4) (52:54:00:6c:17:ca)	Tenant2 (19.0.0.3)

We observe that there is no connectivity between the VMs which have the same MAC addresses. Only one of the VMs with the same MAC addresses can participate in a conversation with the other VM.



Conclusion: If two VMs with same MAC address communicates, then ARP won't be resolved. So L2 datapath from 3 -> 4 will break.

f. What about a VLAN based solution? Will it work to provide isolation? What are the limitations of this solution? No need to perform experiments for this question.

- It will work, using VLANs we can effectively provide L2 isolation between the VMs of two tenants. As the switch attaches the VLAN tag to the packet entering the switch from a particular port, it will work. We should still ensure that VMs of the same tenant do not have the same MAC/IP addresses.

2. **<u>Design 2: Each tenant has its own bridge (in bridge mode)</u>**
   According to design 2, we need to connect the different tenant's VMs to the tenant's own bridge. But, on our hypervisor only ens4 interface is provided. So, we are not able to connect two tenant's bridge to ens4 a the same time. (Datapath from 3 -> 4 for one tenant will break while for other tenant is working)

   a. **What are the disadvantages for the provider? Which resource in the hypervisor hosts will be a bottleneck?**
      i. The design is not feasible. As the number of switches may scale in each hypervisor, it is not possible for the provider to provision as many interfaces as the switches to each hypervisor. The number of interfaces that the provider can attach will become the bottleneck.
   b. **What, if anything, breaks if two tenants in the same hypervisor host use the same IP address?**
      i. If they are connected to different switches beneath the hypervisor by using 2 different network interfaces, then nothing breaks, as both the tenant's VMs are completely isolated from each other.
   c. **What, if anything, breaks if two tenants in the different hypervisor host use the same IP address?**
      i. As having different bridges for different tenants, it will provide L2 network isolation between two tenants. So, everything will work and two tenants will be able to have same IP subnets across different hypervisor.
   d. **What, if anything, breaks if two tenants in the same hypervisor host use the same MAC address?**
      i. Even if two tenants use the same MAC address, nothing breaks as the ARP broadcast packets are isolated from each other by the two switches. There won't be any MAC address conflicts between the tenants.
   e. **What, if anything, breaks if two tenants in a different hypervisor host use the same MAC address?**
      i. As having different bridges for different tenants, it will provide network isolation between two tenants. So, everything will work even if two tenants in different hypervisor host use the same MAC address. The ARP broadcast packets are still isolated from two tenants' different bridges. Therefore, there won't be any conflict.
   f. **Do we need VLANs in the hypervisor bridge or do VLANs in Physical L2 network suffice? No need to perform experiments for this question.**
      i. If there is a single L2 switch beneath the hypervisor which is capable of detecting and transmitting tagged frames, then VLANs in the hypervisor will suffice.

3. **Design 1 vs. Design 2:**
   a. **Admin Hat: List trade-offs with Design 1 and Design 2.**
      i. **Design 1:**
         - The admin has to make sure that, they don't use same subnet address with other tenants inside the hypervisor, as well as across the hypervisor.
         - The admin has to make sure that, the ping packets from their own tenant VM is reaching to their own other VMs. So, they have to make sure that, they are not sending packets to other tenant or receiving packets from other tenant.
         - The admin also has to configure unique MAC address or resolve MAC address conflicts inside and across the hypervisor VMs.

ii. **Design 2:**
    - The admin doesn't have to worry about the subnet address conflicts, MAC address conflicts across different tenants.

b. Provider hat (hypervisor host's configuration point of view): List trade-offs with Design 1 and Design 2.

i. Design 1:
- Easier for the provider to configure, but doesn't provide isolation between tenants. A lot of the bandwidth will be wasted on ARP broadcasts, as there is no L2 isolation.

ii. Design 2:
- More expensive to configure, in terms of providing different interfaces. But provides isolation and more available bandwidth compared to design 1 to each tenant.

Reference:

For Problem 3 and 4,

We have performed experiment with other Team (Staging Server IP:152.14.18.157 and VM ip: 192.168.122.38) by setting the infrastructure as explained.

Problem 4:

Basic topology setup:

	Hypervisor1	Hypervisor2
VM1	Tenant1 (119.0.0.2)	Tenant1 (117.0.0.50)
VM2	Tenant2 (119.0.0.3)	Tenant2 (117.0.0.51)

Tshark capture of our VM pinging the neighbor VM in neighbor hypervisor

```
ece792@ece792-Standard-PC-i440FX-PIIX-1996:/etc/libvirt/qemu$ sudo tshark -i ens4
Running as user "root" and group "root". This could be dangerous.
tshark: Lua: Error during loading:
 [string "/usr/share/wireshark/init.lua"]:32: dofile has been disabled due to running Wireshark as superuser. See https
k.org/CaptureSetup/CapturePrivileges for help in running Wireshark as an unprivileged user.
Capturing on 'ens4'
    1 0.000000000    119.0.0.2 → 117.0.0.51   ICMP 98 Echo (ping) request  id=0x095c, seq=3/768, ttl=63
    2 0.001160625   117.0.0.51 → 119.0.0.2    ICMP 98 Echo (ping) reply    id=0x095c, seq=3/768, ttl=63 (request in 1)
```

Screenshot of Tenant1VM1

```
[root@localhost ~]# ifconfig eth1
eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 119.0.0.2  netmask 255.255.255.0  broadcast 119.0.0.255
        ether 52:54:00:b0:68:a7  txqueuelen 1000  (Ethernet)
        RX packets 548  bytes 63711 (62.2 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 487  bytes 57382 (56.0 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

[root@localhost ~]# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         192.168.122.1   0.0.0.0         UG    0      0        0 eth0
117.0.0.0       119.0.0.1       255.255.255.0   UG    0      0        0 eth1
119.0.0.0       0.0.0.0         255.255.255.0   U     0      0        0 eth1
192.168.122.0   0.0.0.0         255.255.255.0   U     0      0        0 eth0
```

Screenshot of Tenant1VM1 trying to ping Tenant1VM2 (on the peer hypervisor)

```
[root@localhost ~]# ping 117.0.0.50
PING 117.0.0.50 (117.0.0.50) 56(84) bytes of data.
64 bytes from 117.0.0.50: icmp_seq=1 ttl=62 time=1.59 ms
64 bytes from 117.0.0.50: icmp_seq=2 ttl=62 time=13.0 ms
64 bytes from 117.0.0.50: icmp_seq=3 ttl=62 time=1.42 ms
64 bytes from 117.0.0.50: icmp_seq=4 ttl=62 time=1.50 ms
64 bytes from 117.0.0.50: icmp_seq=5 ttl=62 time=1.68 ms
64 bytes from 117.0.0.50: icmp_seq=6 ttl=62 time=8.87 ms
64 bytes from 117.0.0.50: icmp_seq=7 ttl=62 time=1.61 ms
```

1. Design 1:

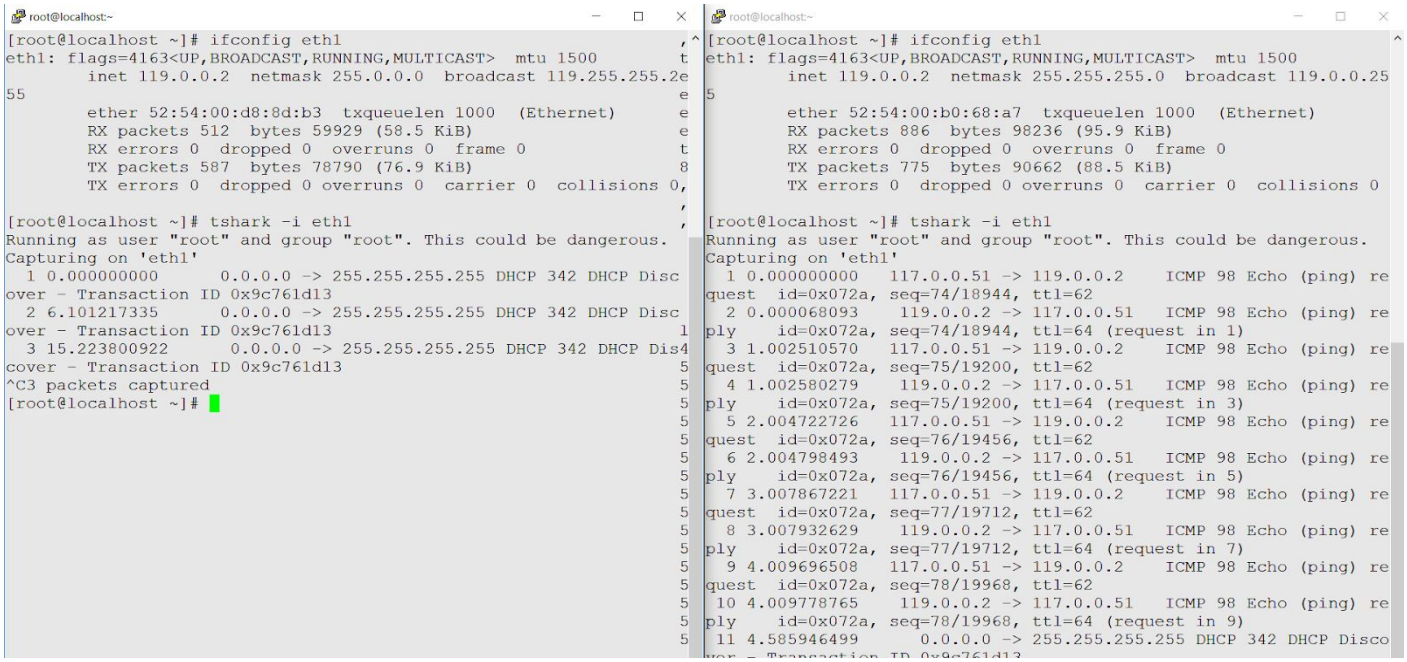
(a) What are the disadvantages for tenants? Is a tenant's traffic isolated from other tenants?

- The tenant's traffic isn't isolated from other tenants.
- And as they use a common bridge, the tenant1 and tenant2 should use the same subnet IP addresses within the same hypervisor.
- All the tenants' VMs will be in the same subnet inside one hypervisor.

(b) What, if anything, breaks if two tenants in the same hypervisor host use the same IP address?

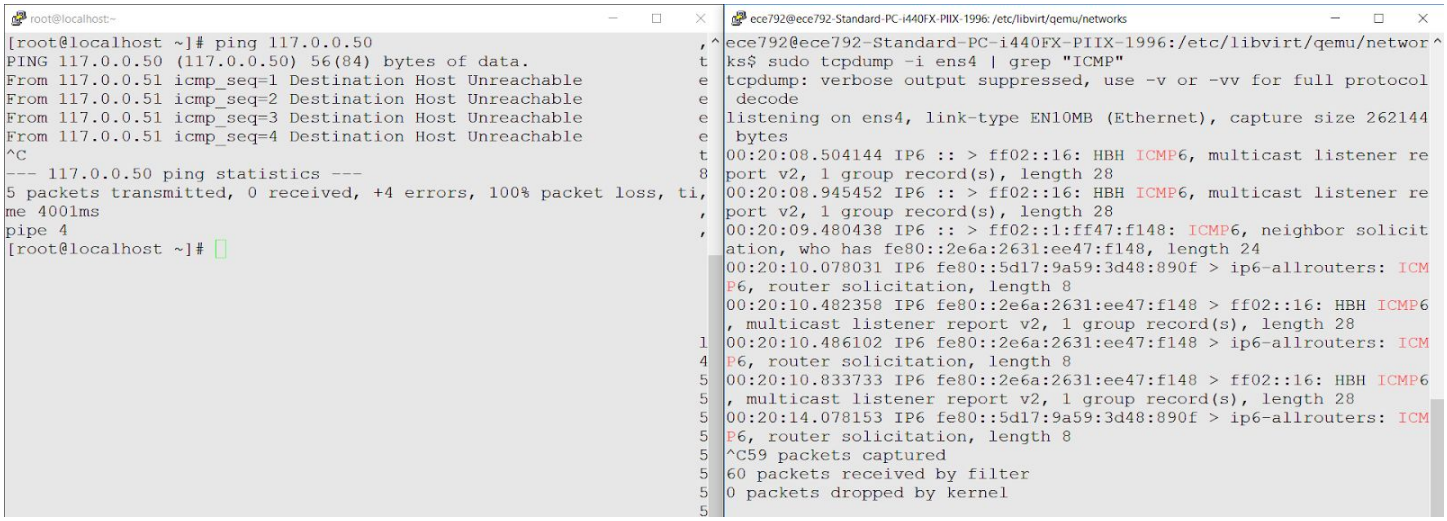
- L2 datapath breaks from 2->1, as only one of the VMs is reachable.
- Case 1:

	Hypervisor1	Hypervisor2
VM1	Tenant1 (119.0.0.2)	Tenant1 (117.0.0.50)
VM2	Tenant2 (119.0.0.2)	Tenant2 (117.0.0.51)



- In the above we can observe that only one of the VMs is connected to the network at a point in time. When the ARP reply comes for 119.0.0.2, the last machine replying to ARP takes preference and the gateway uses that MAC address to send the reply. L2 datapath breaks from the gateway to the VM(with the duplicate IP). Though the MAC field is filled correctly, it will be mapped to a different port due to the other IP.

(c) What, if anything, breaks if two tenants in a different hypervisor host use the same IP address?



Connectivity is lost to the VM on the other hypervisor having the same IP as the ARP broadcast never extends beyond the OVS switch. I.e. L3 datapath breaks at the gateway (3 -> 4), as the gateway is in the subnet, it doesn't transmit the packet out of it. Hence L2 encap decap does not take place at the gateway.

The hypervisor2 has a VM with duplicate IP (tenant 2). When tenant1's VM on hypervisor2 tries to contact the VM on hypervisor1, the packets end up at Teannt2 in the Hypervisor2.

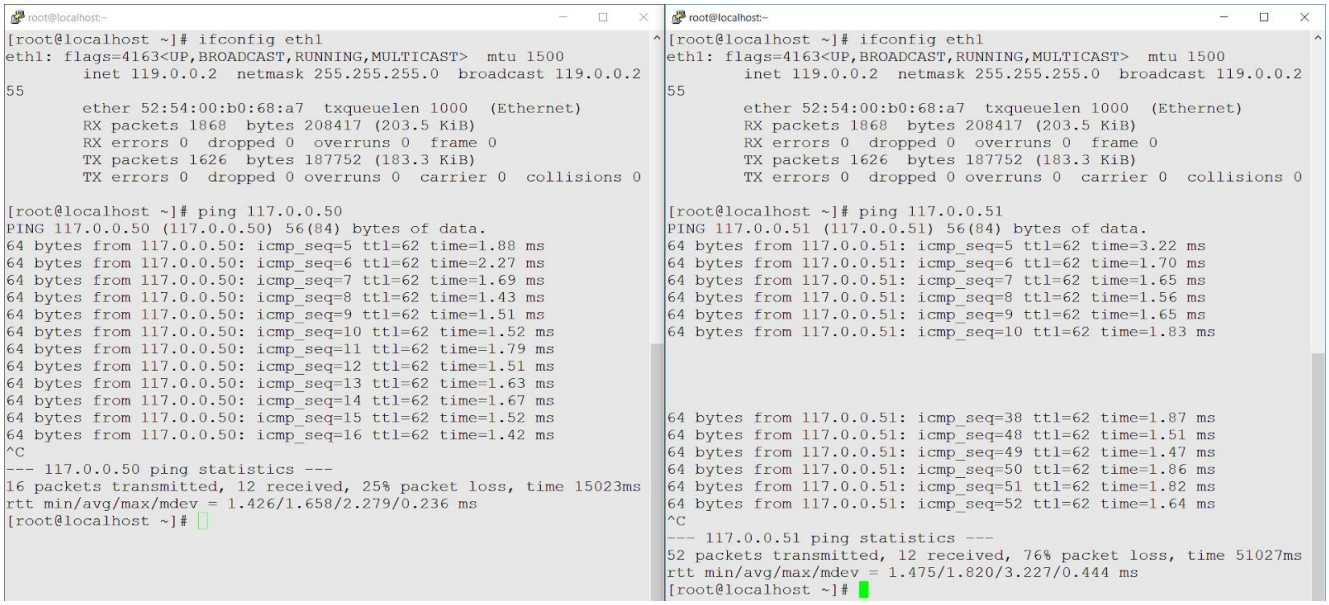
-

	Hypervisor1	Hypervisor2
VM1	Tenant1 (119.0.0.2)	Tenant1 (117.0.0.50)
VM2	Tenant2 (119.0.0.3)	Tenant2 (119.0.0.2)

(d) What, if anything, breaks if two tenants in the same hypervisor host use the same MAC address?

- There is no communication between 2 IPs having the same MAC within the same hypervisor. The MAC is resolved to the port of the sending IP and the packet is sent back on the sending VM's output interface. L2 breaks inside the OVS bridge.

	Hypervisor1	Hypervisor2
VM1	Tenant1 (119.0.0.2)	Tenant1 (117.0.0.50) (52:54:00:b0:68:a7)
VM2	Tenant2 (119.0.0.3)	Tenant2 (117.0.0.51) (52:54:00:b0:68:a7)



This above screenshot displays the scenario when we try to ping the peer VM having duplicate IPs. Only one of the 2 peer devices is reachable at a point in time, if any other VM tries to communicate the original connection is lost. The data path from switch on peer hypervisor to the peer VM i.e. the L2 breaks (8) -> (9) datapath will break.

(e) What, if anything, breaks if two tenants in a different hypervisor host use the same MAC address?

-

	Hypervisor1	Hypervisor2
VM1	Tenant1 (119.0.0.2) (52:54:00:b0:68:a7)	Tenant1 (117.0.0.50)
VM2	Tenant2 (119.0.0.3)	Tenant2 (117.0.0.51) (52:54:00:b0:68:a7)

Ping Successful of two VMs having same mac address

```
[root@localhost ~]# ifconfig eth1
eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 119.0.0.2  netmask 255.255.255.0  broadcast 119.0.0.255
        ether 52:54:00:b0:68:a7  txqueuelen 1000  (Ethernet)
        RX packets 2440  bytes 284642 (277.9 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 2141  bytes 245834 (240.0 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

[root@localhost ~]# ping 117.0.0.50
PING 117.0.0.50 (117.0.0.50) 56(84) bytes of data.
64 bytes from 117.0.0.50: icmp_seq=1 ttl=62 time=1.74 ms
64 bytes from 117.0.0.50: icmp_seq=2 ttl=62 time=2.93 ms
64 bytes from 117.0.0.50: icmp_seq=3 ttl=62 time=1.19 ms
64 bytes from 117.0.0.50: icmp_seq=4 ttl=62 time=1.36 ms
64 bytes from 117.0.0.50: icmp_seq=5 ttl=62 time=1.50 ms
64 bytes from 117.0.0.50: icmp_seq=6 ttl=62 time=1.49 ms
64 bytes from 117.0.0.50: icmp_seq=7 ttl=62 time=1.43 ms
64 bytes from 117.0.0.50: icmp_seq=8 ttl=62 time=1.47 ms
64 bytes from 117.0.0.50: icmp_seq=9 ttl=62 time=1.37 ms
64 bytes from 117.0.0.50: icmp_seq=10 ttl=62 time=1.65 ms
64 bytes from 117.0.0.50: icmp_seq=11 ttl=62 time=1.88 ms
64 bytes from 117.0.0.50: icmp_seq=12 ttl=62 time=2.12 ms
64 bytes from 117.0.0.50: icmp_seq=13 ttl=62 time=1.60 ms
64 bytes from 117.0.0.50: icmp_seq=14 ttl=62 time=1.58 ms
```

Wireshark capture at ens4 of hypervisor

```
  231 32.582002029   117.0.0.50 → 119.0.0.3    ICMP 98 Echo (ping) request  id=0x069a, seq=284/7169, ttl=63
  232 32.582352111    119.0.0.3 → 117.0.0.50   ICMP 98 Echo (ping) reply    id=0x069a, seq=284/7169, ttl=63 (request in 231)
  235 32.710310094 fe80::11cc:36b6:16d:adb2 → ff02::16     ICMPv6 90 Multicast Listener Report Message v2
  240 33.100623689          :: → ff02::16     ICMPv6 90 Multicast Listener Report Message v2
  247 33.584151602   117.0.0.50 → 119.0.0.3    ICMP 98 Echo (ping) request  id=0x069a, seq=285/7425, ttl=63
  248 33.584482000    119.0.0.3 → 117.0.0.50   ICMP 98 Echo (ping) reply    id=0x069a, seq=285/7425, ttl=63 (request in 247)
277   250 33.637822984          :: → ff02::16     ICMPv6 90 Multicast Listener Report Message v2
  253 33.857761499          :: → ff02::1:ff7b:c772 ICMPv6 78 Neighbor Solicitation for fe80::2a67:ad78:fc7b:c772
  261 34.586793847   117.0.0.50 → 119.0.0.3    ICMP 98 Echo (ping) request  id=0x069a, seq=286/7681, ttl=63
  262 34.587263517    119.0.0.3 → 117.0.0.50   ICMP 98 Echo (ping) reply    id=0x069a, seq=286/7681, ttl=63 (request in 261)
281   264 34.859960237 fe80::2a67:ad78:fc7b:c772 → ff02::16     ICMPv6 90 Multicast Listener Report Message v2
  266 34.944465411 fe80::2a67:ad78:fc7b:c772 → ff02::2      ICMPv6 62 Router Solicitation
  268 35.345209993 fe80::2a67:ad78:fc7b:c772 → ff02::16     ICMPv6 90 Multicast Listener Report Message v2
  269 35.588925898   117.0.0.50 → 119.0.0.3    ICMP 98 Echo (ping) request  id=0x069a, seq=287/7937, ttl=63
```

Conclusion: Nothing breaks.

(f) What about a VLAN based solution for providing L3 connectivity to each VM? Will it work? What are the limitations of this solution?

It would not work as VLAN provides L2 isolation. Even if we use VLANs here the gateway would remove the VLAN tag before transmitting the packet which defeats the purpose of attaching it.

2. Design 2:

Topology Setup:

	Hypervisor1	Hypervisor2
VM1	Tenant1 (119.0.0.2) (L25 is the OVS routed bridge)	Tenant1 (117.0.0.50)
VM2	Tenant2 (121.0.0.2) (L35 is the OVS routed bridge)	Tenant2 (118.0.0.51)

At our hypervisor:

Tenant1 VM:

```
[root@localhost ~]# ifconfig eth1
eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 119.0.0.2  netmask 255.255.255.0  broadcast 119.0.0.255
        ether 52:54:00:b0:68:a7  txqueuelen 1000  (Ethernet)
        RX packets 3108  bytes 497853 (486.1 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 2825  bytes 466158 (455.2 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

[root@localhost ~]# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
117.0.0.0       119.0.0.1       255.255.255.0   UG    0      0        0 eth1
118.0.0.0       119.0.0.1       255.255.255.0   UG    0      0        0 eth1
119.0.0.0       0.0.0.0         255.255.255.0   U     0      0        0 eth1
192.168.122.0   0.0.0.0         255.255.255.0   U     0      0        0 eth0
```

Tenant2 VM:

```
[root@localhost ~]# ifconfig eth1
eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 121.0.0.2  netmask 255.255.255.0  broadcast 121.0.0.255
        ether 52:54:00:32:5e:f9  txqueuelen 1000  (Ethernet)
        RX packets 37  bytes 2853 (2.7 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 98  bytes 16888 (16.4 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

[root@localhost ~]# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
29.0.0.0        121.0.0.1       255.255.255.0   UG    0      0        0 eth1
117.0.0.0       121.0.0.1       255.255.255.0   UG    0      0        0 eth1
118.0.0.0       121.0.0.1       255.255.255.0   UG    0      0        0 eth1
121.0.0.0       0.0.0.0         255.255.255.0   U     0      0        0 eth1
192.168.122.0   0.0.0.0         255.255.255.0   U     0      0        0 eth0
```

Hypervisor Configuration and Route

```
ece792@ece792-Standard-PC-i440FX-PIIX-1996:~$ ifconfig ens4
ens4      Link encap:Ethernet  HWaddr 52:54:00:b7:0b:df
          inet addr:29.0.0.2  Bcast:29.0.0.255  Mask:255.255.255.0
          inet6 addr: fe80::5054:ff:feb7:bdf/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:828805 errors:0 dropped:0 overruns:0 frame:0
          TX packets:26945 errors:0 dropped:0 overruns:0 carrier:0
          collisions:300 txqueuelen:1000
          RX bytes:138309230 (138.3 MB)  TX bytes:4428301 (4.4 MB)

ece792@ece792-Standard-PC-i440FX-PIIX-1996:~$ route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         192.168.124.1   0.0.0.0         UG    100    0        0 ens3
0.0.0.0         192.168.123.1   0.0.0.0         UG    101    0        0 ens5
29.0.0.0        0.0.0.0         255.255.255.0   U     0      0        0 ens4
54.0.0.0        0.0.0.0         255.255.255.0   U     0      0        0 veth13
117.0.0.0       29.0.0.4        255.255.255.0   UG    0      0        0 ens4
118.0.0.0       29.0.0.4        255.255.255.0   UG    0      0        0 ens4
119.0.0.0       0.0.0.0         255.255.255.0   U     0      0        0 l25
121.0.0.0       0.0.0.0         255.255.255.0   U     0      0        0 l35
169.254.0.0     0.0.0.0         255.255.0.0     U     1000   0        0 ens5
192.168.100.0   0.0.0.0         255.255.255.0   U     0      0        0 routedbr
192.168.118.0   0.0.0.0         255.255.255.0   U     0      0        0 sw6
192.168.122.0   0.0.0.0         255.255.255.0   U     0      0        0 virbr0
192.168.123.0   0.0.0.0         255.255.255.0   U     0      0        0 ens5
192.168.123.0   0.0.0.0         255.255.255.0   U     100    0        0 ens5
192.168.124.0   0.0.0.0         255.255.255.0   U     100    0        0 ens3
```

(a) What are the disadvantages for the provider? Which resource in the hypervisor hosts will be a Bottleneck?

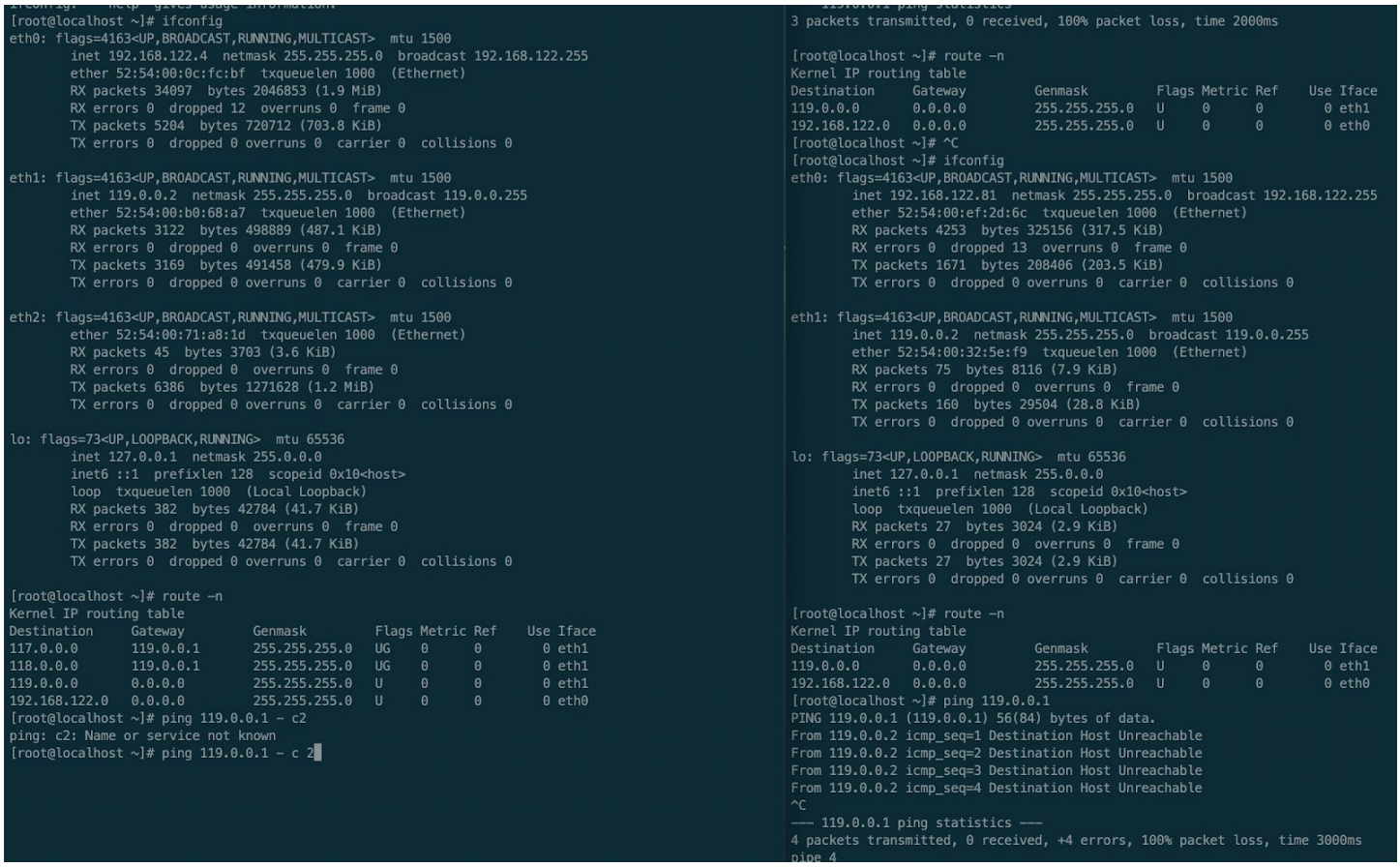
- The provider needs to ensure that the subnets provisioned are mutually exclusive.
- The same tenant can not have the same subnet across different hypervisors, i.e. this design doesn't provide L3 isolation.
- The physical interface ens4 of the hypervisor will be the bottleneck as all the tenants will share the bandwidth on it.

(b) What, if anything, breaks if two tenants in the same hypervisor host use the same IP address?

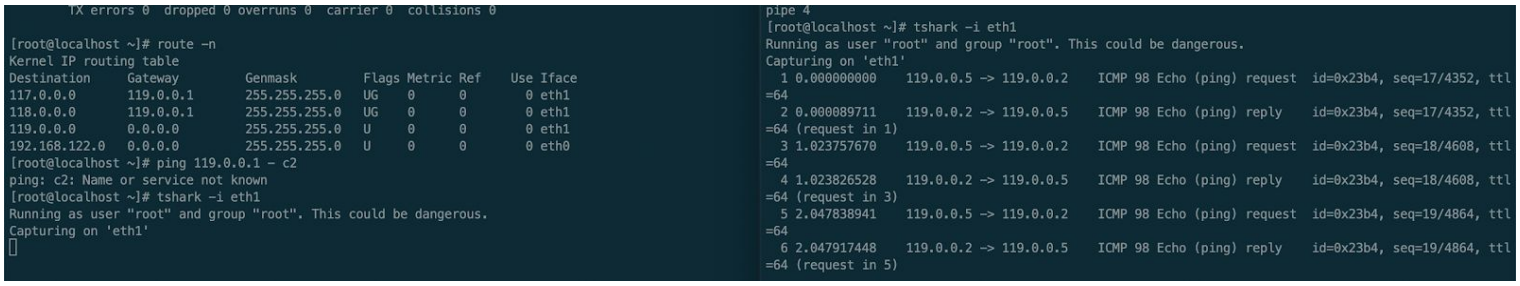
	Hypervisor1	Hypervisor2
VM1	Tenant1 (119.0.0.2)	Tenant1 (117.0.0.50)
VM2	Tenant2 (119.0.0.2)	Tenant2 (118.0.0.51)

- Case 1: (Gateway's ip is not changed)
  The L2 datapath from 1 -> 2 breaks as VM(119.0.0.2) is in a different subnet(121.0.0.1/24). As they are in different subnets, the gateway doesn't respond to the ARP requests from the VM.

- Case 2: When the gateway is also assigned an IP of 119.0.0.1, the L2 datapath between the VM(with the duplicate IP) and the gateway(119.0.0.1) breaks.
  Screenshot for case 2



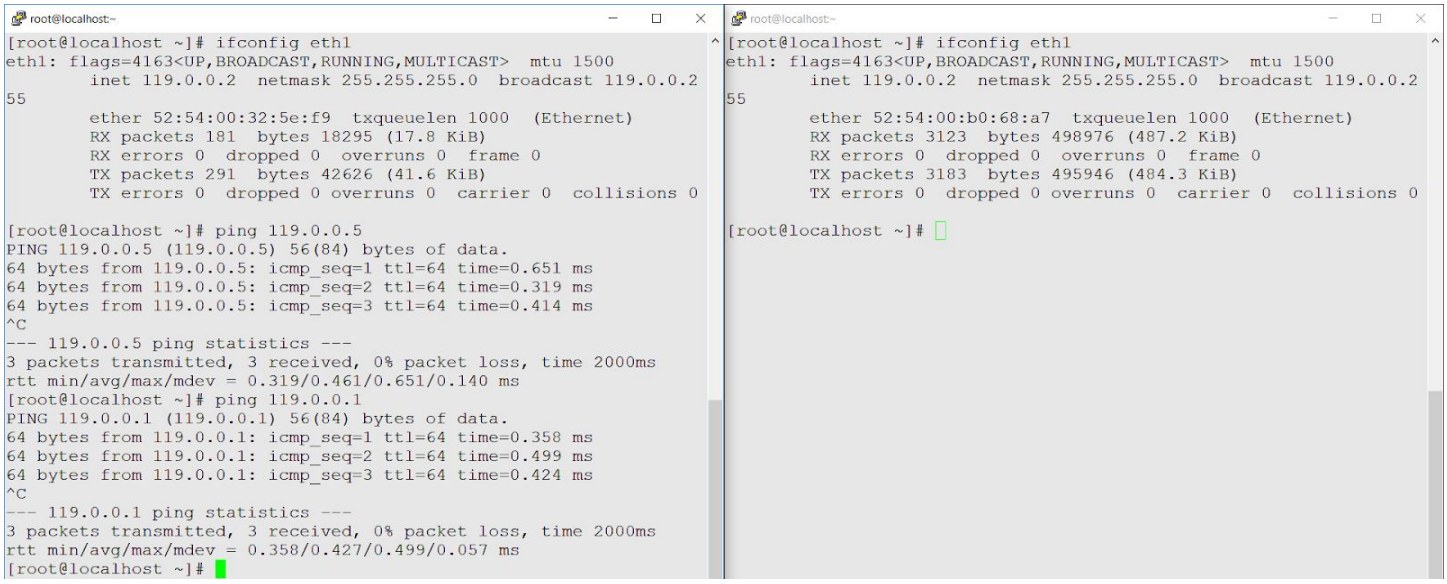
- Case 3: When we assign the gateway of other tenant as different IP of 119.0.0.10, the peer hypervisor loses connectivity to one of the subnets completely. The screenshot below displays the tshark capture on eth1 (of VM with duplicate IP) and tcpdump on ens4 when our peer hypervisor tries to reach us.
  Case 3 Screenshot:



Screenshot of tcpdump:

```
ece792@ece792-Standard-PC-i440FX-PIIX-1996:~$ sudo tcpdump -i ens4

tcpdump: verbose output suppressed, use -v or -vv for full protoco
l decode
listening on ens4, link-type EN10MB (Ethernet), capture size 26214
4 bytes
12:54:38.456942 IP localhost > 119.0.0.2: ICMP echo request, id 17
22, seq 5, length 64
12:54:38.516252 IP 0.0.0.0.bootpc > 255.255.255.255.bootps: BOOTP/
DHCP, Request from 52:54:00:fa:d7:68 (oui Unknown), length 301
12:54:38.799237 IP 0.0.0.0.bootpc > 255.255.255.255.bootps: BOOTP/
DHCP, Request from 52:54:00:77:34:30 (oui Unknown), length 300
12:54:38.862207 IP 0.0.0.0.bootpc > 255.255.255.255.bootps: BOOTP/
DHCP, Request from 52:54:00:b5:34:15 (oui Unknown), length 301
12:54:39.036737 STP 802.1d, Config, Flags [none], bridge-id 8001.0
0:1b:2b:22:c9:80.8009, length 43
12:54:39.117669 IP6 fe80::a8f9:5583:79:df4.mdns > ff02::fb.mdns: 0
*- [0q] 2/0/0 (Cache flush) PTR ece792-Standard-PC-i440FX-PIIX-201
6.local., (Cache flush) AAAA fe80::a8f9:5583:79:df4 (167)
12:54:39.456681 IP localhost > 119.0.0.2: ICMP echo request, id 17
22, seq 6, length 64
12:54:39.538870 ARP, Request who-has 245.245.245.3 tell 245.245.24
5.5, length 46
12:54:40.302050 IP6 fe80::69ba:ff02:afd3:19c7.mdns > ff02::fb.mdns
: 0 [2q] PTR (QM)? _ipps._tcp.local. PTR (QM)? _ipp._tcp.local. (4
5)
```

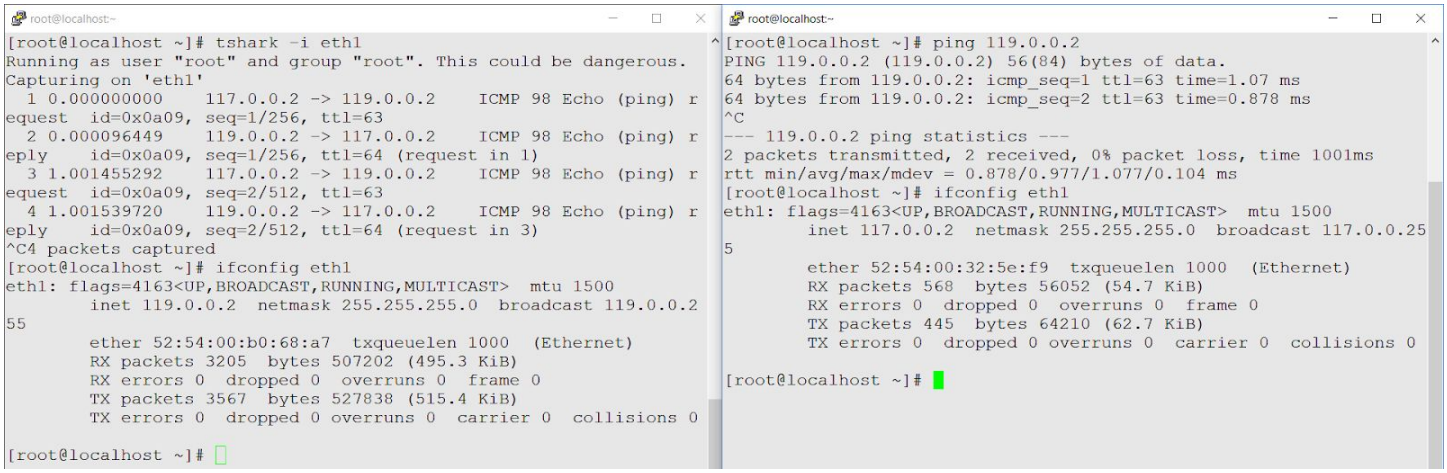
- In this below we can observe, if the gateway IPs are different then the VM with duplicate IP is able to ping both the gateways.



(c) What, if anything, breaks if two tenants in a different hypervisor host use the same IP address?

	Hypervisor1	Hypervisor2
VM1	Tenant1 (119.0.0.2)	Tenant1 (117.0.0.2)
VM2	Tenant2 (117.0.0.2)	Tenant2 (118.0.0.51)

- The L3 datapath breaks from 3 -> 4, as the IP subnet is connected locally the ping packet never crosses the gateway in the hypervisor and ends up in the local subnet.
- So basically Tenant1 wants the packet to send out to Tenant1's other VM running on different hypervisor. But, the ICMP packet reaches to Tenant2's VM on same hypervisor.

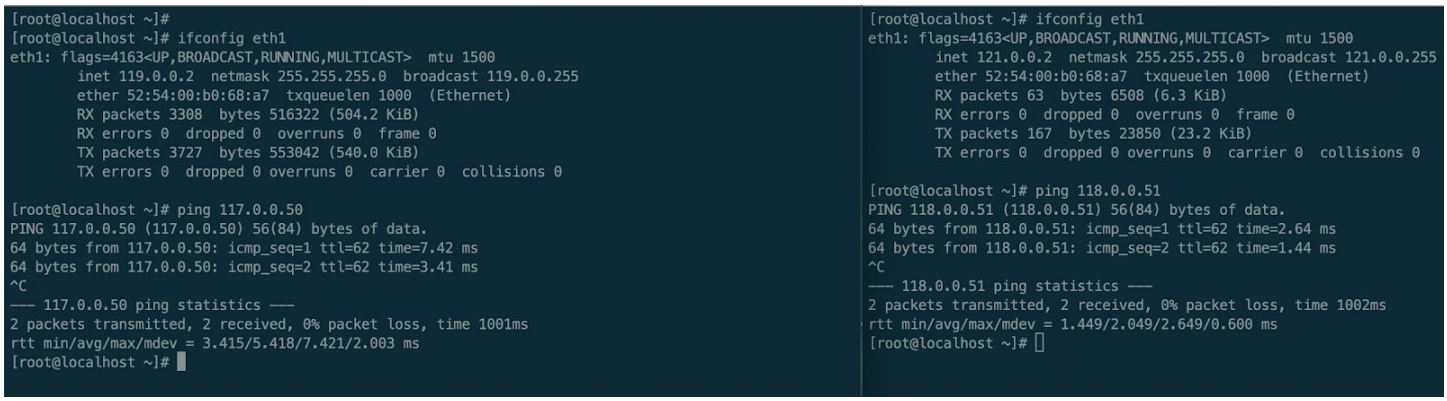


(d) What, if anything, breaks if two tenants in the same hypervisor host use the same MAC address?

	Hypervisor1	Hypervisor2
VM1	Tenant1 (119.0.0.2) (MAC Address: 52:54:00:b0:68:a7)	Tenant1 (117.0.0.2)
VM2	Tenant2 (121.0.0.2) (MAC Address: 52:54:00:b0:68:a7)	Tenant2 (118.0.0.51)

Nothing breaks, as both the VMs are connected to different switches(in routed mode). Even when they have the same MAC address there is no conflict in the MAC table.

Screenshot shows that both the tenants are able to ping to their respective VMs on different hypervisor.



(e) What, if anything, breaks if two tenants in a different hypervisor host use the same MAC address?

	Hypervisor1	Hypervisor2
VM1	Tenant1 (119.0.0.2) (MAC Address: 52:54:00:b0:68:a7)	Tenant1 (117.0.0.2)
VM2	Tenant2 (121.0.0.2)	Tenant2 (118.0.0.51) (MAC Address: 52:54:00:b0:68:a7)

Nothing breaks, as both the VMs are connected to different switches(in routed mode). Even when they have the same MAC address there is no conflict in the MAC table.
And here we have L3 datapath so MAC conflicts will not break any datapaths.

Tenant1 across different hypervisors are able to ping each other:

```
[root@localhost ~]# ifconfig eth1
eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 119.0.0.2  netmask 255.255.255.0  broadcast 119.0.0.255
        ether 52:54:00:b0:68:a7  txqueuelen 1000  (Ethernet)
        RX packets 3313  bytes 516644 (504.5 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 3741  bytes 555386 (542.3 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

[root@localhost ~]# ping 117.0.0.50
PING 117.0.0.50 (117.0.0.50) 56(84) bytes of data.
64 bytes from 117.0.0.50: icmp_seq=1 ttl=62 time=2.35 ms
64 bytes from 117.0.0.50: icmp_seq=2 ttl=62 time=2.02 ms
64 bytes from 117.0.0.50: icmp_seq=3 ttl=62 time=1.38 ms
^C
--- 117.0.0.50 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2003ms
rtt min/avg/max/mdev = 1.385/1.922/2.354/0.404 ms
[root@localhost ~]#
```

Tenant2 across different hypervisors are able to ping each other:

```
[root@localhost ~]# ifconfig eth1
eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 121.0.0.2  netmask 255.255.255.0  broadcast 121.0.0.255
        ether 52:54:00:b7:ae:d4  txqueuelen 1000  (Ethernet)
        RX packets 31  bytes 2299 (2.2 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 246  bytes 21532 (21.0 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

[root@localhost ~]# ping 118.0.0.51
PING 118.0.0.51 (118.0.0.51) 56(84) bytes of data.
64 bytes from 118.0.0.51: icmp_seq=1 ttl=62 time=1.30 ms
64 bytes from 118.0.0.51: icmp_seq=2 ttl=62 time=1.47 ms
c64 bytes from 118.0.0.51: icmp_seq=3 ttl=62 time=1.39 ms
64 bytes from 118.0.0.51: icmp_seq=4 ttl=62 time=1.53 ms
^C
--- 118.0.0.51 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3005ms
rtt min/avg/max/mdev = 1.305/1.427/1.534/0.089 ms
```

(f) Does a VLAN based solution in this design overcome any limitations of the VLAN based solution used in design 1?

- It would not work as VLAN provides L2 isolation. Even if we use VLANs here the gateway would remove the VLAN tag before transmitting the packet which defeats the purpose of attaching it.

3. Design 1 vs. Design 2:

(a) Admin Hat: List trade-offs with Design 1 and Design 2.

Design 1:

- And as they use a common bridge, the tenant1 and tenant2 should use the same subnet IP addresses within the same hypervisor.
- All the tenants' VMs will be in the same subnet inside one hypervisor.
- Doesn't provide L2 isolation among tenants within the same subnet.

Design 2:

- To provide L2 isolation every new tenant should be provisioned with a new bridge in routed mode.
- Two tenants won't be able to use the same subnet inside and across hypervisor/s.

(b) Provider hat (hypervisor host's configuration point of view): List trade-offs with Design 1 and Design 2.

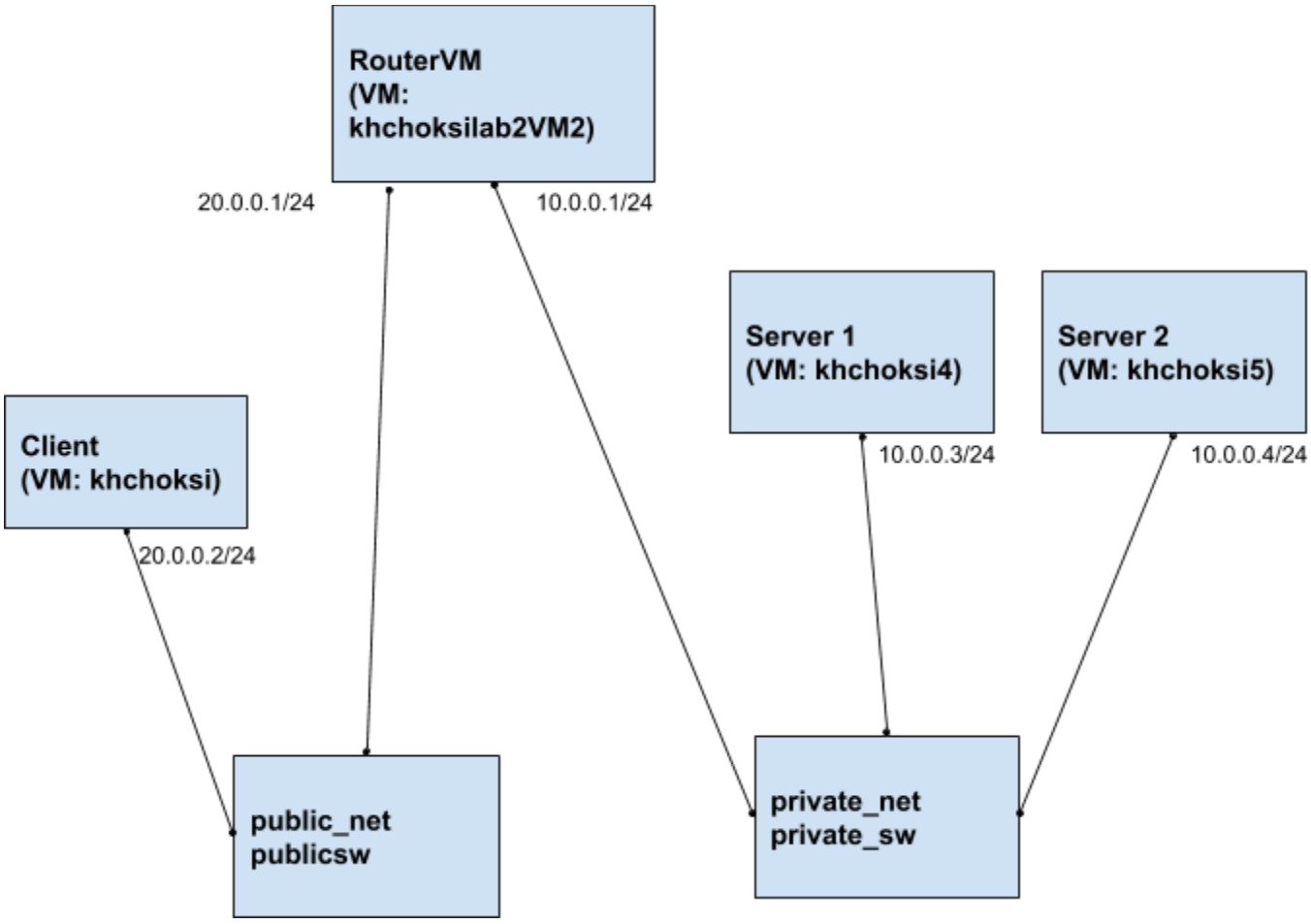
Design 1:

- Easier configuration, provider will just add a single route for all packets entering into hypervisor to the common single bridge.

Design 2:

- Provider has configure route table entries for every subnet / tenant existing in the hypervisor for connectivity with the Internet or other VMs.

Problem 5:



Prerequisite: To enable IP forwarding in Router VM:
In the file /etc/sysctl.conf, change the line:

```
net.ipv4.ip_forward = 0
```

to

```
net.ipv4.ip_forward = 1
```

1. **Set the topology as given in Figure. List IP/subnet plan for each L3 interfaces and provide Forwarding Table output for each VM.**
    a. **For router VM (khchoksilab2VM2)**

```
[root@localhost ~]# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         192.168.122.1   0.0.0.0         UG    0      0        0 eth0
10.0.0.0        0.0.0.0         255.255.255.0   U     0      0        0 eth1
20.0.0.0        0.0.0.0         255.255.255.0   U     0      0        0 eth2
192.168.122.0   0.0.0.0         255.255.255.0   U     0      0        0 eth0
[root@localhost ~]# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.122.51  netmask 255.255.255.0  broadcast 192.168.122.255
        ether 52:54:00:01:8a:4c  txqueuelen 1000  (Ethernet)
        RX packets 5474  bytes 423354 (413.4 KiB)
        RX errors 0  dropped 56  overruns 0  frame 0
        TX packets 2335  bytes 359588 (351.1 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.0.0.1  netmask 255.255.255.0  broadcast 10.0.0.255
        ether 52:54:00:30:95:89  txqueuelen 1000  (Ethernet)
        RX packets 370  bytes 44082 (43.0 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 195  bytes 29570 (28.8 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

eth2: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 20.0.0.1  netmask 255.255.255.0  broadcast 20.0.0.255
        ether 52:54:00:3f:35:6b  txqueuelen 1000  (Ethernet)
        RX packets 604  bytes 122163 (119.2 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 234  bytes 40672 (39.7 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 73  bytes 8908 (8.6 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 73  bytes 8908 (8.6 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

b. For client VM (khchoksi)

```
[root@localhost ~]# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         192.168.122.1   0.0.0.0         UG    0      0        0 eth0
20.0.0.0        0.0.0.0         255.255.255.0   U     0      0        0 eth1
192.168.122.0   0.0.0.0         255.255.255.0   U     0      0        0 eth0
[root@localhost ~]# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.122.109  netmask 255.255.255.0  broadcast 192.168.122.255
        ether 52:54:00:01:f0:cc  txqueuelen 1000  (Ethernet)
        RX packets 2640  bytes 170880 (166.8 KiB)
        RX errors 0  dropped 60  overruns 0  frame 0
        TX packets 536  bytes 66265 (64.7 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 20.0.0.2  netmask 255.255.255.0  broadcast 20.0.0.255
        ether 52:54:00:88:17:38  txqueuelen 1000  (Ethernet)
        RX packets 667  bytes 126500 (123.5 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 160  bytes 34072 (33.2 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

c. **For ServerVM1(khchoksi4)**

```
[root@localhost ~]# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         192.168.118.1   0.0.0.0         UG    0      0        0 eth1
0.0.0.0         192.168.118.1   0.0.0.0         UG    100    0        0 eth1
10.0.0.0        0.0.0.0         255.255.255.0   U     0      0        0 eth0
20.0.0.0        10.0.0.1        255.255.255.0   UG    0      0        0 eth0
192.168.118.0   0.0.0.0         255.255.255.0   U     0      0        0 eth1
192.168.118.0   0.0.0.0         255.255.255.0   U     100    0        0 eth1
[root@localhost ~]# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.0.0.3  netmask 255.255.255.0  broadcast 10.0.0.255
        ether 52:54:00:2d:eb:45  txqueuelen 1000  (Ethernet)
        RX packets 197  bytes 30724 (30.0 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 360  bytes 42288 (41.2 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.118.228  netmask 255.255.255.0  broadcast 192.168.118.255
        inet6 fe80::18cd:10b4:9cda:dca4  prefixlen 64  scopeid 0x20<link>
        ether 52:54:00:62:24:60  txqueuelen 1000  (Ethernet)
        RX packets 5262  bytes 409173 (399.5 KiB)
        RX errors 0  dropped 22  overruns 0  frame 0
        TX packets 2735  bytes 349051 (340.8 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 134  bytes 15008 (14.6 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 134  bytes 15008 (14.6 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

d. For ServerVM2 (khchoksi5)

```
[root@localhost ~]# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         192.168.118.1   0.0.0.0         UG    0      0        0 eth0
10.0.0.0        0.0.0.0         255.255.255.0   U     0      0        0 eth1
20.0.0.0        10.0.0.1        255.255.255.0   UG    0      0        0 eth1
192.168.118.0   0.0.0.0         255.255.255.0   U     0      0        0 eth0
[root@localhost ~]# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.118.27  netmask 255.255.255.0  broadcast 192.168.118.255
        ether 52:54:00:f4:b6:f5  txqueuelen 1000  (Ethernet)
        RX packets 3390  bytes 232020 (226.5 KiB)
        RX errors 0  dropped 65  overruns 0  frame 0
        TX packets 860  bytes 94836 (92.6 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.0.0.4  netmask 255.255.255.0  broadcast 10.0.0.255
        ether 52:54:00:ab:99:e9  txqueuelen 1000  (Ethernet)
        RX packets 321  bytes 75837 (74.0 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 475  bytes 91566 (89.4 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 16  bytes 1792 (1.7 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 16  bytes 1792 (1.7 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

[root@localhost ~]#
```

2. Ping from server to client

```
sudo iptables -t nat -I PREROUTING 1 -p icmp -s 20.0.0.2 -j DNAT --to-destination 10.0.0.3
sudo iptables -t nat -I PREROUTING 1 -p icmp -s 20.0.0.2 -j DNAT --to-destination 10.0.0.4
sudo iptables -t nat -I PREROUTING 1 -p icmp -d 20.0.0.2 -j SNAT --to 20.0.0.1
```

```
[root@localhost ~]# sudo iptables -t nat -L --line-number
Chain PREROUTING (policy ACCEPT)
num  target     prot opt source               destination
1    DNAT       icmp --  20.0.0.2             anywhere             to:10.0.0.4
2    DNAT       icmp --  20.0.0.2             anywhere             to:10.0.0.3
3    PREROUTING_direct  all  --  anywhere             anywhere
4    PREROUTING_ZONES_SOURCE  all  --  anywhere             anywhere
5    PREROUTING_ZONES  all  --  anywhere             anywhere

Chain INPUT (policy ACCEPT)
num  target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
num  target     prot opt source               destination
1    OUTPUT_direct  all  --  anywhere             anywhere

Chain POSTROUTING (policy ACCEPT)
num  target     prot opt source               destination
1    SNAT       icmp --  anywhere             20.0.0.2             to:20.0.0.1
2    POSTROUTING_direct  all  --  anywhere             anywhere
3    POSTROUTING_ZONES_SOURCE  all  --  anywhere             anywhere
4    POSTROUTING_ZONES  all  --  anywhere             anywhere
```

Following are wireshark captures if do ping from Server VM1 (10.0.0.3) to Client (20.0.0.2):

- Wireshark at Client VM

```
[root@localhost ~]# sudo tshark -i eth1 -T fields -e ip.src -e eth.src -e ip.dst -e eth.dst -e col.Protocol
Running as user "root" and group "root". This could be dangerous.
Capturing on 'eth1'
20.0.0.1        52:54:00:3f:35:6b       20.0.0.2        52:54:00:88:17:38       ICMP
20.0.0.2        52:54:00:88:17:38       20.0.0.1        52:54:00:3f:35:6b       ICMP
20.0.0.1        52:54:00:3f:35:6b       20.0.0.2        52:54:00:88:17:38       ICMP
20.0.0.2        52:54:00:88:17:38       20.0.0.1        52:54:00:3f:35:6b       ICMP
20.0.0.1        52:54:00:3f:35:6b       20.0.0.2        52:54:00:88:17:38       ICMP
20.0.0.2        52:54:00:88:17:38       20.0.0.1        52:54:00:3f:35:6b       ICMP
20.0.0.1        52:54:00:3f:35:6b       20.0.0.2        52:54:00:88:17:38       ICMP
20.0.0.2        52:54:00:88:17:38       20.0.0.1        52:54:00:3f:35:6b       ICMP
20.0.0.1        52:54:00:3f:35:6b       20.0.0.2        52:54:00:88:17:38       ICMP
20.0.0.2        52:54:00:88:17:38       20.0.0.1        52:54:00:3f:35:6b       ICMP
        52:54:00:3f:35:6b               52:54:00:88:17:38       ARP
        52:54:00:88:17:38               52:54:00:3f:35:6b       ARP
20.0.0.1        52:54:00:3f:35:6b       20.0.0.2        52:54:00:88:17:38       ICMP
20.0.0.2        52:54:00:88:17:38       20.0.0.1        52:54:00:3f:35:6b       ICMP
20.0.0.1        52:54:00:3f:35:6b       20.0.0.2        52:54:00:88:17:38       ICMP
20.0.0.2        52:54:00:88:17:38       20.0.0.1        52:54:00:3f:35:6b       ICMP
20.0.0.1        52:54:00:3f:35:6b       20.0.0.2        52:54:00:88:17:38       ICMP
20.0.0.2        52:54:00:88:17:38       20.0.0.1        52:54:00:3f:35:6b       ICMP
20.0.0.1        52:54:00:3f:35:6b       20.0.0.2        52:54:00:88:17:38       ICMP
20.0.0.2        52:54:00:88:17:38       20.0.0.1        52:54:00:3f:35:6b       ICMP
20.0.0.1        52:54:00:3f:35:6b       20.0.0.2        52:54:00:88:17:38       ICMP
20.0.0.2        52:54:00:88:17:38       20.0.0.1        52:54:00:3f:35:6b       ICMP
        52:54:00:88:17:38               52:54:00:3f:35:6b       ARP
        52:54:00:3f:35:6b               52:54:00:88:17:38       ARP
20.0.0.1        52:54:00:3f:35:6b       20.0.0.2        52:54:00:88:17:38       ICMP
20.0.0.2        52:54:00:88:17:38       20.0.0.1        52:54:00:3f:35:6b       ICMP
20.0.0.1        52:54:00:3f:35:6b       20.0.0.2        52:54:00:88:17:38       ICMP
20.0.0.2        52:54:00:88:17:38       20.0.0.1        52:54:00:3f:35:6b       ICMP
```

- Wireshark at RouterVM eth1 (connected to public net)

```
[root@localhost ~]# tshark -i eth1 -T fields -e ip.src -e eth.src -e ip.dst -e eth.src -e col.Protocol
Running as user "root" and group "root". This could be dangerous.
Capturing on 'eth1'
0.0.0.0         255.255.255.255 52:54:00:30:95:89       DHCP
0.0.0.0         255.255.255.255 52:54:00:30:95:89       DHCP
0.0.0.0         255.255.255.255 52:54:00:30:95:89       DHCP
0.0.0.0         255.255.255.255 52:54:00:30:95:89       DHCP
0.0.0.0         255.255.255.255 52:54:00:30:95:89       DHCP
10.0.0.3                20.0.0.2        52:54:00:2d:eb:45       ICMP
20.0.0.2                10.0.0.3        52:54:00:30:95:89       ICMP
10.0.0.3                20.0.0.2        52:54:00:2d:eb:45       ICMP
20.0.0.2                10.0.0.3        52:54:00:30:95:89       ICMP
10.0.0.3                20.0.0.2        52:54:00:2d:eb:45       ICMP
20.0.0.2                10.0.0.3        52:54:00:30:95:89       ICMP
10.0.0.3                20.0.0.2        52:54:00:2d:eb:45       ICMP
20.0.0.2                10.0.0.3        52:54:00:30:95:89       ICMP
10.0.0.3                20.0.0.2        52:54:00:2d:eb:45       ICMP
20.0.0.2                10.0.0.3        52:54:00:30:95:89       ICMP
```

- Wireshark at RouterVM eth2 (connected to private net)

```
[root@localhost ~]# sudo tshark -i eth2 -T fields -e ip.src -e eth.src -e ip.dst -e eth.dst -e col.Protocol
Running as user "root" and group "root". This could be dangerous.
Capturing on 'eth2'
20.0.0.1        52:54:00:3f:35:6b       20.0.0.2        52:54:00:88:17:38       ICMP
20.0.0.2        52:54:00:88:17:38       20.0.0.1        52:54:00:3f:35:6b       ICMP
20.0.0.1        52:54:00:3f:35:6b       20.0.0.2        52:54:00:88:17:38       ICMP
20.0.0.2        52:54:00:88:17:38       20.0.0.1        52:54:00:3f:35:6b       ICMP
20.0.0.1        52:54:00:3f:35:6b       20.0.0.2        52:54:00:88:17:38       ICMP
20.0.0.2        52:54:00:88:17:38       20.0.0.1        52:54:00:3f:35:6b       ICMP
20.0.0.1        52:54:00:3f:35:6b       20.0.0.2        52:54:00:88:17:38       ICMP
20.0.0.2        52:54:00:88:17:38       20.0.0.1        52:54:00:3f:35:6b       ICMP
20.0.0.1        52:54:00:3f:35:6b       20.0.0.2        52:54:00:88:17:38       ICMP
20.0.0.2        52:54:00:88:17:38       20.0.0.1        52:54:00:3f:35:6b       ICMP
        52:54:00:3f:35:6b               52:54:00:88:17:38       ARP
        52:54:00:88:17:38               52:54:00:3f:35:6b       ARP
20.0.0.1        52:54:00:3f:35:6b       20.0.0.2        52:54:00:88:17:38       ICMP
20.0.0.2        52:54:00:88:17:38       20.0.0.1        52:54:00:3f:35:6b       ICMP
20.0.0.1        52:54:00:3f:35:6b       20.0.0.2        52:54:00:88:17:38       ICMP
20.0.0.2        52:54:00:88:17:38       20.0.0.1        52:54:00:3f:35:6b       ICMP
20.0.0.1        52:54:00:3f:35:6b       20.0.0.2        52:54:00:88:17:38       ICMP
20.0.0.2        52:54:00:88:17:38       20.0.0.1        52:54:00:3f:35:6b       ICMP
20.0.0.1        52:54:00:3f:35:6b       20.0.0.2        52:54:00:88:17:38       ICMP
20.0.0.2        52:54:00:88:17:38       20.0.0.1        52:54:00:3f:35:6b       ICMP
20.0.0.1        52:54:00:3f:35:6b       20.0.0.2        52:54:00:88:17:38       ICMP
20.0.0.2        52:54:00:88:17:38       20.0.0.1        52:54:00:3f:35:6b       ICMP
        52:54:00:88:17:38               52:54:00:3f:35:6b       ARP
        52:54:00:3f:35:6b               52:54:00:88:17:38       ARP
20.0.0.1        52:54:00:3f:35:6b       20.0.0.2        52:54:00:88:17:38       ICMP
20.0.0.2        52:54:00:88:17:38       20.0.0.1        52:54:00:3f:35:6b       ICMP
20.0.0.1        52:54:00:3f:35:6b       20.0.0.2        52:54:00:88:17:38       ICMP
20.0.0.2        52:54:00:88:17:38       20.0.0.1        52:54:00:3f:35:6b       ICMP
20.0.0.1        52:54:00:3f:35:6b       20.0.0.2        52:54:00:88:17:38       ICMP
20.0.0.2        52:54:00:88:17:38       20.0.0.1        52:54:00:3f:35:6b       ICMP
20.0.0.1        52:54:00:3f:35:6b       20.0.0.2        52:54:00:88:17:38       ICMP
20.0.0.2        52:54:00:88:17:38       20.0.0.1        52:54:00:3f:35:6b       ICMP
20.0.0.1        52:54:00:3f:35:6b       20.0.0.2        52:54:00:88:17:38       ICMP
20.0.0.2        52:54:00:88:17:38       20.0.0.1        52:54:00:3f:35:6b       ICMP
```

- Wireshark at ServerVM (10.0.0.3)

```
[root@localhost ~]# ping 20.0.0.2
PING 20.0.0.2 (20.0.0.2) 56(84) bytes of data.
64 bytes from 20.0.0.2: icmp_seq=1 ttl=63 time=4.95 ms
64 bytes from 20.0.0.2: icmp_seq=2 ttl=63 time=3.75 ms
64 bytes from 20.0.0.2: icmp_seq=3 ttl=63 time=1.46 ms
64 bytes from 20.0.0.2: icmp_seq=4 ttl=63 time=1.89 ms
64 bytes from 20.0.0.2: icmp_seq=5 ttl=63 time=2.20 ms
64 bytes from 20.0.0.2: icmp_seq=6 ttl=63 time=1.40 ms
64 bytes from 20.0.0.2: icmp_seq=7 ttl=63 time=2.27 ms
64 bytes from 20.0.0.2: icmp_seq=8 ttl=63 time=2.44 ms
64 bytes from 20.0.0.2: icmp_seq=9 ttl=63 time=3.00 ms
64 bytes from 20.0.0.2: icmp_seq=10 ttl=63 time=4.39 ms
64 bytes from 20.0.0.2: icmp_seq=11 ttl=63 time=3.12 ms
64 bytes from 20.0.0.2: icmp_seq=12 ttl=63 time=4.27 ms
64 bytes from 20.0.0.2: icmp_seq=13 ttl=63 time=2.77 ms
```

```
[root@localhost ~]# tshark -i eth0 -T fields -e ip.src -e eth.src
-e ip.dst -e eth.src -e col.Protocol
Running as user "root" and group "root". This could be dangerous.
Capturing on 'eth0'
10.0.0.3                20.0.0.2        52:54:00:2d:eb:45       IC
MP
20.0.0.2                10.0.0.3        52:54:00:30:95:89       IC
MP
10.0.0.3                20.0.0.2        52:54:00:2d:eb:45       IC
MP
20.0.0.2                10.0.0.3        52:54:00:30:95:89       IC
MP
10.0.0.3                20.0.0.2        52:54:00:2d:eb:45       IC
MP
20.0.0.2                10.0.0.3        52:54:00:30:95:89       IC
MP
10.0.0.3                20.0.0.2        52:54:00:2d:eb:45       IC
MP
20.0.0.2                10.0.0.3        52:54:00:30:95:89       IC
MP
10.0.0.3                20.0.0.2        52:54:00:2d:eb:45       IC
MP
20.0.0.2                10.0.0.3        52:54:00:30:95:89       IC
MP
10.0.0.3                20.0.0.2        52:54:00:2d:eb:45       IC
```

3. Configure NAT / PAT proxy settings

```
iptables -t nat -I PREROUTING 1 -p tcp -s 20.0.0.2 --dport 2000 -j DNAT --to 10.0.0.3:22
iptables -t nat -I PREROUTING 1 -p tcp -s 20.0.0.2 --dport 2001 -j DNAT --to 10.0.0.4:22
iptables -I FORWARD 1 -i eth2 -o eth1 -p tcp -s 20.0.0.2 --dport 22 -j ACCEPT
iptables -t nat -I POSTROUTING 1 -p tcp -d 10.0.0.3 --dport 22 -j SNAT --to-source 10.0.0.1
iptables -t nat -I POSTROUTING 1 -p tcp -d 10.0.0.4 --dport 22 -j SNAT --to-source 10.0.0.1
```

```
[root@localhost ~]# iptables -t nat -L
Chain PREROUTING (policy ACCEPT)
target     prot opt source               destination
DNAT       tcp  --  20.0.0.2             anywhere             tcp dpt:dc to:10.0.0.4:22
DNAT       tcp  --  20.0.0.2             anywhere             tcp dpt:sieve-filter to:10.0.0.3:22
PREROUTING_direct  all  --  anywhere             anywhere
PREROUTING_ZONES_SOURCE  all  --  anywhere             anywhere
PREROUTING_ZONES  all  --  anywhere             anywhere

Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
OUTPUT_direct  all  --  anywhere             anywhere

Chain POSTROUTING (policy ACCEPT)
target     prot opt source               destination
SNAT       tcp  --  anywhere             10.0.0.4             tcp dpt:ssh to:10.0.0.1
SNAT       tcp  --  anywhere             10.0.0.3             tcp dpt:ssh to:10.0.0.1
POSTROUTING_direct  all  --  anywhere             anywhere
POSTROUTING_ZONES_SOURCE  all  --  anywhere             anywhere
POSTROUTING_ZONES  all  --  anywhere             anywhere
```

```
[root@localhost ~]# iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
ACCEPT     all  --  anywhere             anywhere             ctstate RELATED,ESTABLISHED
ACCEPT     all  --  anywhere             anywhere
INPUT_direct  all  --  anywhere             anywhere
INPUT_ZONES_SOURCE  all  --  anywhere             anywhere
INPUT_ZONES  all  --  anywhere             anywhere
DROP       all  --  anywhere             anywhere             ctstate INVALID
REJECT     all  --  anywhere             anywhere             reject-with icmp-host-prohibited

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination
ACCEPT     tcp  --  20.0.0.2             anywhere             tcp dpt:ssh
ACCEPT     all  --  anywhere             anywhere             ctstate RELATED,ESTABLISHED
ACCEPT     all  --  anywhere             anywhere
FORWARD_direct  all  --  anywhere             anywhere
FORWARD_IN_ZONES_SOURCE  all  --  anywhere             anywhere
FORWARD_IN_ZONES  all  --  anywhere             anywhere
FORWARD_OUT_ZONES_SOURCE  all  --  anywhere             anywhere
FORWARD_OUT_ZONES  all  --  anywhere             anywhere
DROP       all  --  anywhere             anywhere             ctstate INVALID
REJECT     all  --  anywhere             anywhere             reject-with icmp-host-prohibited
```

SSH from Client to Server 1

```
[root@localhost ~]# ssh root@20.0.0.1 -p 2000
root@20.0.0.1's password:
Last login: Wed Oct 24 00:11:53 2018 from 10.0.0.1
[root@localhost ~]# ifconfig eth0
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.0.0.3  netmask 255.255.255.0  broadcast 10.0.0.255
        ether 52:54:00:2d:eb:45  txqueuelen 1000  (Ethernet)
        RX packets 1025  bytes 190194 (185.7 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 1369  bytes 216888 (211.8 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

SSH from Client to Server 2

```
[root@localhost ~]# ssh root@20.0.0.1 -p 2001
root@20.0.0.1's password:
Last login: Wed Oct 24 00:08:33 2018 from 10.0.0.1
[root@localhost ~]# ifconfig eth1
eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.0.0.4  netmask 255.255.255.0  broadcast 10.0.0.255
        ether 52:54:00:ad:7a:5d  txqueuelen 1000  (Ethernet)
        RX packets 176  bytes 25779 (25.1 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 175  bytes 33053 (32.2 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Wireshark Captures for ssh from client vm to Server 2:
i.e. ssh root@20.0.0.1 -p 2001

Following are screenshots for the same:
Wireshark capture at eth2 (connected to publicnet) of RouterVM

```
[root@localhost ~]# sudo tshark -i eth2 -T fields -e ip.src -e tcp.srcport -e ip.dst -e tcp.dstport -e col.Protocol
Running as user "root" and group "root". This could be dangerous.
Capturing on 'eth2'
20.0.0.2        60100   20.0.0.1        2001    TCP
20.0.0.1        2001    20.0.0.2        60100   TCP
20.0.0.2        60100   20.0.0.1        2001    TCP
20.0.0.2        60100   20.0.0.1        2001    TCP
20.0.0.1        2001    20.0.0.2        60100   TCP
20.0.0.1        2001    20.0.0.2        60100   TCP
20.0.0.2        60100   20.0.0.1        2001    TCP
20.0.0.2        60100   20.0.0.1        2001    TCP
20.0.0.1        2001    20.0.0.2        60100   TCP
20.0.0.1        2001    20.0.0.2        60100   TCP
20.0.0.2        60100   20.0.0.1        2001    TCP
20.0.0.1        2001    20.0.0.2        60100   TCP
20.0.0.2        60100   20.0.0.1        2001    TCP
20.0.0.1        2001    20.0.0.2        60100   TCP
20.0.0.2        60100   20.0.0.1        2001    TCP
20.0.0.1        2001    20.0.0.2        60100   TCP
20.0.0.2        60100   20.0.0.1        2001    TCP
```

Wireshark capture at eth1 (connected to privatenet) of RouterVM

```
[root@localhost ~]# sudo tshark -i eth1 -T fields -e ip.src -e tcp.srcport  -e ip.dst -e tcp.dstport -e col.Protocol
Running as user "root" and group "root". This could be dangerous.
Capturing on 'eth1'
0.0.0.0         255.255.255.255         DHCP
10.0.0.1        60098   10.0.0.4        22      TCP
10.0.0.4        22      10.0.0.1        60098   TCP
10.0.0.1        60098   10.0.0.4        22      TCP
10.0.0.1        60098   10.0.0.4        22      SSH
10.0.0.4        22      10.0.0.1        60098   TCP
10.0.0.4        22      10.0.0.1        60098   SSHv2
10.0.0.1        60098   10.0.0.4        22      TCP
10.0.0.1        60098   10.0.0.4        22      SSHv2
10.0.0.4        22      10.0.0.1        60098   TCP
10.0.0.4        22      10.0.0.1        60098   SSHv2
10.0.0.1        60098   10.0.0.4        22      SSHv2
10.0.0.4        22      10.0.0.1        60098   SSHv2
```

Wireshark capture at eth1 (connected to privatenet) of Server2VM (10.0.0.4)

```
[root@localhost ~]# sudo tshark -i eth1 -T fields -e ip.src -e tcp.srcport  -e ip.dst -e tcp.dstport -e col.Protocol
Running as user "root" and group "root". This could be dangerous.
Capturing on 'eth1'
0.0.0.0         255.255.255.255         DHCP
10.0.0.1        60098   10.0.0.4        22      TCP
10.0.0.4        22      10.0.0.1        60098   TCP
10.0.0.1        60098   10.0.0.4        22      TCP
10.0.0.1        60098   10.0.0.4        22      SSH
10.0.0.4        22      10.0.0.1        60098   TCP
10.0.0.4        22      10.0.0.1        60098   SSHv2
10.0.0.1        60098   10.0.0.4        22      TCP
10.0.0.1        60098   10.0.0.4        22      SSHv2
10.0.0.4        22      10.0.0.1        60098   TCP
10.0.0.4        22      10.0.0.1        60098   SSHv2
10.0.0.1        60098   10.0.0.4        22      SSHv2
10.0.0.4        22      10.0.0.1        60098   SSHv2
10.0.0.1        60098   10.0.0.4        22      SSHv2
10.0.0.4        22      10.0.0.1        60098   TCP
10.0.0.1        60098   10.0.0.4        22      SSHv2
10.0.0.4        22      10.0.0.1        60098   TCP
10.0.0.4        22      10.0.0.1        60098   TCP
10.0.0.1        60098   10.0.0.4        22      SSHv2
10.0.0.4        22      10.0.0.1        60098   TCP
10.0.0.1        60098   10.0.0.4        22      TCP
10.0.0.1        60098   10.0.0.4        22      SSHv2
10.0.0.4        22      10.0.0.1        60098   TCP
10.0.0.4        22      10.0.0.1        60098   TCP
```

4.

(a) Defining Load: Send TCP packets for port range 2000:2499 to Server 1(10.0.0.3) and 2500:3000 to Server 2(10.0.0.4). (e.g. in a real-life scenario, let's say we have many clients who wants to send TCP packets to our servers' 22 port. Also, assume that server 1,2 are running same application on port 22. So we will load balance based on port number ranges from clients. But client will not know from which server their data is being served.)

(b) Configure the balancing knob at the RouterVM.

```
iptables -t nat -I PREROUTING 1 -p tcp -s 20.0.0.0/24 --dport 2000:2499 -j DNAT --to
```

```
10.0.0.3:22
iptables -t nat -I PREROUTING 1 -p tcp -s 20.0.0.0/24 --dport 2500:3000 -j DNAT --to
10.0.0.4:22
iptables -I FORWARD 1 -i eth2 -o eth1 -p tcp -s 20.0.0.0/24 --dport 22 -j ACCEPT
iptables -t nat -I POSTROUTING 1 -p tcp -d 10.0.0.3 --dport 22 -j SNAT --to-source 10.0.0.1
iptables -t nat -I POSTROUTING 1 -p tcp -d 10.0.0.4 --dport 22 -j SNAT --to-source 10.0.0.1
```

(c) Verify that the load balancing mechanism is working.

From client machine:

```
$ssh root@20.0.0.1 -p 2100 (so it will redirect to Server1)
```

```
[root@localhost ~]# ssh root@20.0.0.1 -p 2100
root@20.0.0.1's password:
Last login: Fri Oct 26 20:08:42 2018 from 10.0.0.1
[root@localhost ~]# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.0.0.3  netmask 255.255.255.0  broadcast 10.0.0.255
        ether 52:54:00:2d:eb:45  txqueuelen 1000  (Ethernet)
        RX packets 216  bytes 45927 (44.8 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 143  bytes 33672 (32.8 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.118.228  netmask 255.255.255.0  broadcast 192.168.118.255
        inet6 fe80::18cd:10b4:9cda:dca4  prefixlen 64  scopeid 0x20<link>
        ether 52:54:00:62:24:60  txqueuelen 1000  (Ethernet)
        RX packets 2384  bytes 153683 (150.0 KiB)
        RX errors 0  dropped 23  overruns 0  frame 0
        TX packets 657  bytes 74160 (72.4 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

```
$ssh root@20.0.0.1 -p 2600 (so it will redirect to Server2)
```

```
[root@localhost ~]# ssh root@20.0.0.1 -p 2600
root@20.0.0.1's password:
Last login: Fri Oct 26 20:15:03 2018 from 10.0.0.1
[root@localhost ~]# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.118.59  netmask 255.255.255.0  broadcast 192.168.118.255
        ether 52:54:00:41:19:16  txqueuelen 1000  (Ethernet)
        RX packets 143645  bytes 31906125 (30.4 MiB)
        RX errors 0  dropped 20  overruns 0  frame 0
        TX packets 18123  bytes 1324127 (1.2 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.0.0.4  netmask 255.255.255.0  broadcast 10.0.0.255
        ether 52:54:00:ad:7a:5d  txqueuelen 1000  (Ethernet)
        RX packets 8957  bytes 2873583 (2.7 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 4640  bytes 1490972 (1.4 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

IP Tables configuration:

```
[root@localhost ~]# iptables -t nat -L
Chain PREROUTING (policy ACCEPT)
target     prot opt source               destination
DNAT       tcp  --  20.0.0.0/24         anywhere             tcp dpts:rtsserv:hbci to:10.0.0.4:22
DNAT       tcp  --  20.0.0.0/24         anywhere             tcp dpts:sieve-filter:2499 to:10.0.0.3:22
PREROUTING_direct  all  --  anywhere             anywhere
PREROUTING_ZONES_SOURCE  all  --  anywhere             anywhere
PREROUTING_ZONES  all  --  anywhere             anywhere

Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
OUTPUT_direct  all  --  anywhere             anywhere

Chain POSTROUTING (policy ACCEPT)
target     prot opt source               destination
SNAT       tcp  --  anywhere             10.0.0.4             tcp dpt:ssh to:10.0.0.1
SNAT       tcp  --  anywhere             10.0.0.3             tcp dpt:ssh to:10.0.0.1
POSTROUTING_direct  all  --  anywhere             anywhere
POSTROUTING_ZONES_SOURCE  all  --  anywhere             anywhere
POSTROUTING_ZONES  all  --  anywhere             anywhere
```

```
[root@localhost ~]# iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
ACCEPT     all  --  anywhere             anywhere             ctstate RELATED,ESTABLISHED
ACCEPT     all  --  anywhere             anywhere
INPUT_direct  all  --  anywhere             anywhere
INPUT_ZONES_SOURCE  all  --  anywhere             anywhere
INPUT_ZONES  all  --  anywhere             anywhere
DROP       all  --  anywhere             anywhere             ctstate INVALID
REJECT     all  --  anywhere             anywhere             reject-with icmp-host-prohibited

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination
ACCEPT     tcp  --  20.0.0.0/24          anywhere             tcp dpt:ssh
ACCEPT     all  --  anywhere             anywhere             ctstate RELATED,ESTABLISHED
ACCEPT     all  --  anywhere             anywhere
FORWARD_direct  all  --  anywhere             anywhere
FORWARD_IN_ZONES_SOURCE  all  --  anywhere             anywhere
FORWARD_IN_ZONES  all  --  anywhere             anywhere
FORWARD_OUT_ZONES_SOURCE  all  --  anywhere             anywhere
FORWARD_OUT_ZONES  all  --  anywhere             anywhere
DROP       all  --  anywhere             anywhere             ctstate INVALID
REJECT     all  --  anywhere             anywhere             reject-with icmp-host-prohibited
```

Wireshark Captures:

Ex1: Traffic forwarded to Server1

`$ssh root@20.0.0.1 -p 2100 (so it will redirect to Server1)`

Wireshark capture at eth0 (connected to privatenet) of Server2VM (10.0.0.3)

```
[root@localhost ~]# sudo tshark -i eth0 -T fields -e ip.src -e tcp.srcport -e ip.dst -e tcp.dstport -e col.Protocol
Running as user "root" and group "root". This could be dangerous.
Capturing on 'eth0'
10.0.0.1        37172   10.0.0.3        22      TCP
10.0.0.3        22      10.0.0.1        37172   TCP
10.0.0.1        37172   10.0.0.3        22      TCP
10.0.0.1        37172   10.0.0.3        22      SSH
10.0.0.3        22      10.0.0.1        37172   TCP
10.0.0.3        22      10.0.0.1        37172   SSHv2
10.0.0.1        37172   10.0.0.3        22      TCP
10.0.0.1        37172   10.0.0.3        22      SSHv2
10.0.0.3        22      10.0.0.1        37172   TCP
10.0.0.3        22      10.0.0.1        37172   SSHv2
10.0.0.1        37172   10.0.0.3        22      SSHv2
10.0.0.3        22      10.0.0.1        37172   SSHv2
```

Wireshark packet capture at eth1 (connected to privatenet) of the router VM.

```
[root@localhost ~]# sudo tshark -i eth1 -T fields -e ip.src -e tcp.srcport -e ip.dst -e tcp.dstport -e col.Protocol
Running as user "root" and group "root". This could be dangerous.
Capturing on 'eth1'
10.0.0.1        37172   10.0.0.3        22      TCP
10.0.0.3        22      10.0.0.1        37172   TCP
10.0.0.1        37172   10.0.0.3        22      TCP
10.0.0.1        37172   10.0.0.3        22      SSH
10.0.0.3        22      10.0.0.1        37172   TCP
10.0.0.3        22      10.0.0.1        37172   SSHv2
10.0.0.1        37172   10.0.0.3        22      TCP
10.0.0.1        37172   10.0.0.3        22      SSHv2
10.0.0.3        22      10.0.0.1        37172   TCP
10.0.0.3        22      10.0.0.1        37172   SSHv2
10.0.0.1        37172   10.0.0.3        22      SSHv2
```

Wireshark packet capture at eth2 (connected to publicnet) of the router VM.

```
[root@localhost ~]# sudo tshark -i eth2 -T fields -e ip.src -e tcp.srcport -e ip.dst -e tcp.dstport -e col.Protocol
Running as user "root" and group "root". This could be dangerous.
Capturing on 'eth2'
20.0.0.1        2100    20.0.0.2        37162   TCP
20.0.0.2        37162   20.0.0.1        2100    TCP
20.0.0.2        37162   20.0.0.1        2100    TCP
20.0.0.1        2100    20.0.0.2        37162   TCP
20.0.0.1        2100    20.0.0.2        37162   TCP
20.0.0.2        37162   20.0.0.1        2100    TCP
20.0.0.1        2100    20.0.0.2        37162   TCP
20.0.0.2        37162   20.0.0.1        2100    TCP
20.0.0.2        37162   20.0.0.1        2100    TCP
20.0.0.1        2100    20.0.0.2        37162   TCP
20.0.0.2        37162   20.0.0.1        2100    TCP
20.0.0.1        2100    20.0.0.2        37162   TCP
20.0.0.1        2100    20.0.0.2        37162   TCP
20.0.0.2        37162   20.0.0.1        2100    TCP
20.0.0.2        37162   20.0.0.1        2100    TCP
20.0.0.1        2100    20.0.0.2        37162   TCP
20.0.0.2        37162   20.0.0.1        2100    TCP
20.0.0.2        37162   20.0.0.1        2100    TCP
20.0.0.1        2100    20.0.0.2        37162   TCP
20.0.0.1        2100    20.0.0.2        37162   TCP
20.0.0.2        37162   20.0.0.1        2100    TCP
20.0.0.2        37162   20.0.0.1        2100    TCP
20.0.0.1        2100    20.0.0.2        37162   TCP
20.0.0.1        2100    20.0.0.2        37162   TCP
20.0.0.2        37162   20.0.0.1        2100    TCP
20.0.0.1        2100    20.0.0.2        37162   TCP
```

Ex2: Traffic forwarded to Server2

```
$ssh root@20.0.0.1 -p 2600 (so it will redirect to Server2)
```

Wireshark capture at eth1 (connected to privatenet) of Server2VM (10.0.0.4)

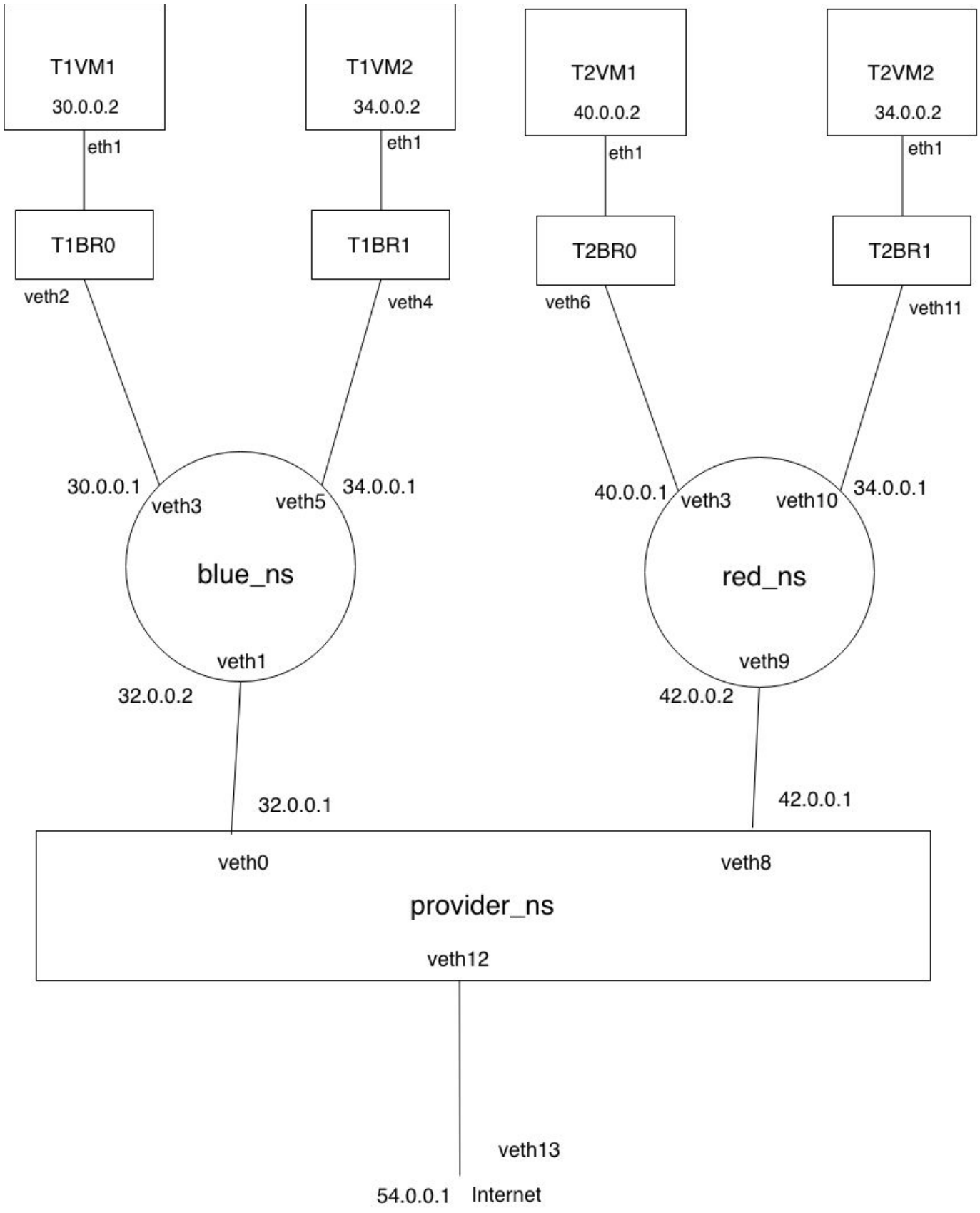
```
[root@localhost ~]# sudo tshark -i eth1 -T fields -e ip.src -e
otocol
Running as user "root" and group "root". This could be dangero
Capturing on 'eth1'
10.0.0.1        36298   10.0.0.4        22      TCP
10.0.0.4        22      10.0.0.1        36298   TCP
10.0.0.1        36298   10.0.0.4        22      TCP
10.0.0.1        36298   10.0.0.4        22      SSH
10.0.0.4        22      10.0.0.1        36298   TCP
10.0.0.4        22      10.0.0.1        36298   SSHv2
10.0.0.1        36298   10.0.0.4        22      TCP
10.0.0.1        36298   10.0.0.4        22      SSHv2
10.0.0.4        22      10.0.0.1        36298   TCP
10.0.0.4        22      10.0.0.1        36298   SSHv2
10.0.0.1        36298   10.0.0.4        22      SSHv2
10.0.0.4        22      10.0.0.1        36298   SSHv2
10.0.0.1        36298   10.0.0.4        22      SSHv2
10.0.0.4        22      10.0.0.1        36298   TCP
10.0.0.1        36298   10.0.0.4        22      SSHv2
10.0.0.4        22      10.0.0.1        36298   TCP
10.0.0.4        22      10.0.0.1        36298   TCP
10.0.0.1        36298   10.0.0.4        22      SSHv2
10.0.0.4        22      10.0.0.1        36298   TCP
```

Wireshark packet capture at eth1 (connected to privatenet) of the router VM.

```
[root@localhost ~]# sudo tshark -i eth1 -T fields -e ip.src -e tcp.srcport -e ip.dst -e tcp.dstport -e col.Protocol
Running as user "root" and group "root". This could be dangerous.
Capturing on 'eth1'
10.0.0.1        36296   10.0.0.4        22      TCP
10.0.0.4        22      10.0.0.1        36296   TCP
10.0.0.1        36296   10.0.0.4        22      TCP
10.0.0.1        36296   10.0.0.4        22      SSH
10.0.0.4        22      10.0.0.1        36296   TCP
10.0.0.4        22      10.0.0.1        36296   SSHv2
10.0.0.1        36296   10.0.0.4        22      TCP
10.0.0.1        36296   10.0.0.4        22      SSHv2
10.0.0.4        22      10.0.0.1        36296   TCP
10.0.0.4        22      10.0.0.1        36296   SSHv2
10.0.0.1        36296   10.0.0.4        22      SSHv2
```

Problem 6:

Topology setup:



Setup commands:
At Tenant1 Subnet 1

```
sudo iptables -t nat -A POSTROUTING -s 30.0.0.0/24 ! -d 30.0.0.0/24 -j MASQUERADE
```

For Tenant1 Subnet 2

```
sudo iptables -t nat -A POSTROUTING -s 32.0.0.0/24 ! -d 32.0.0.0/24 -j MASQUERADE
```

1. Demonstrate the L2 isolation between two subnets of the same tenant. (Hint: Broadcast should be restricted and VMs can have same MAC addresses).

To prove L2 isolation, we show that the ARP broadcasts from one subnet of tenant1 are not transmitted on the other subnet of tenant1, and having same MAC addresses doesn't break anything.

Ex. If T1VM2 (34.0.0.2) tries to ARP for 34.0.0.3 which not actually present in the subnet2 (t1br1), it will not send any of the ARP broadcasts to subnet1 (t1br0).

```
[root@localhost ~]# ping 34.0.0.3
PING 34.0.0.3 (34.0.0.3) 56(84) bytes of data.
From 34.0.0.2 icmp_seq=1 Destination Host Unreachable
From 34.0.0.2 icmp_seq=2 Destination Host Unreachable
From 34.0.0.2 icmp_seq=3 Destination Host Unreachable
From 34.0.0.2 icmp_seq=4 Destination Host Unreachable
^C
--- 34.0.0.3 ping statistics ---
6 packets transmitted, 0 received, +4 errors, 100% packet loss, time 5001ms
pipe 4
```

Wireshark on T1VM1 port eth2

```
[root@localhost ~]# ifconfig eth1
eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 30.0.0.2  netmask 255.255.255.0  broadcast 30.0.0.255
        ether 52:54:00:c5:5c:53  txqueuelen 1000  (Ethernet)
        RX packets 690  bytes 69213 (67.5 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 1203  bytes 215425 (210.3 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

[root@localhost ~]# tcpdump -i eth1
tcpdump: verbose output suppressed, use -v or -vv for full protocol
 decode
listening on eth1, link-type EN10MB (Ethernet), capture size 262144
 bytes
^C
0 packets captured
0 packets received by filter
0 packets dropped by kernel
```

Ex. Now, we set the same MAC address for both T1VM1 and T1VM2.

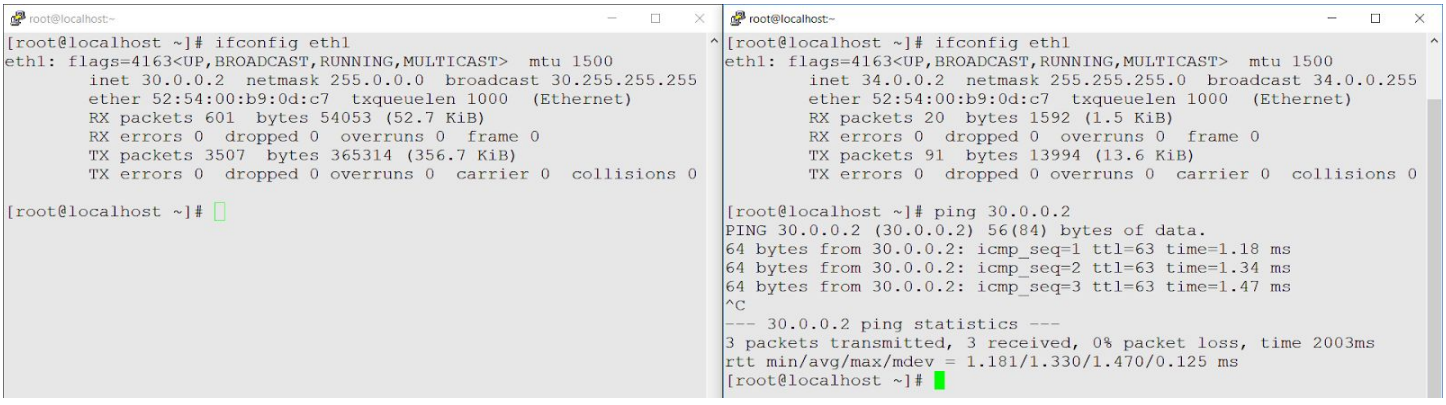
T1VM1

```
[root@localhost ~]# ifconfig eth1
eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 30.0.0.2  netmask 255.0.0.0  broadcast 30.255.255.255
        ether 52:54:00:b9:0d:c7  txqueuelen 1000  (Ethernet)
        RX packets 581  bytes 52333 (51.1 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 3491  bytes 361978 (353.4 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

T1VM2

```
[root@localhost ~]# ifconfig eth1
eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 34.0.0.2  netmask 255.255.255.0  broadcast 34.0.0.255
        ether 52:54:00:b9:0d:c7  txqueuelen 1000  (Ethernet)
        RX packets 6  bytes 404 (404.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 66  bytes 11736 (11.4 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Ping from Tenant1's one VM to another VM is successful even though tenant1 has same MAC address in his VMs. This proves that they have L2 isolation among themselves.



2. Demonstrate the L3 isolation between two tenants.

Setup commands: (We assume 8.8.8.8 to be the internet in this scenario)

Inside blue_ns

```
sudo iptables -t nat -A POSTROUTING -s 30.0.0.0/24 ! -d 30.0.0.0/24 -j MASQUERADE
sudo iptables -t nat -A POSTROUTING -s 34.0.0.0/24 ! -d 34.0.0.0/24 -j MASQUERADE
```

Inside red_ns

```
sudo iptables -t nat -A POSTROUTING -s 40.0.0.0/24 ! -d 40.0.0.0/24 -j MASQUERADE
sudo iptables -t nat -A POSTROUTING -s 34.0.0.0/24 ! -d 34.0.0.0/24 -j MASQUERADE
```

Inside hypervisor

```
sudo iptables -t nat -A POSTROUTING -s 32.0.0.0/24 ! -d 32.0.0.0/24 -j MASQUERADE
sudo iptables -t nat -A POSTROUTING -s 42.0.0.0/24 ! -d 42.0.0.0/24 -j MASQUERADE
```

Routes in hypervisor

```
ip route add 32.0.0.0/24 via 32.0.0.2 dev veth0
ip route add 42.0.0.0/24 via 42.0.0.2 dev veth8
```

```
ece792@ece792-Standard-PC-i440FX-PIIX-1996:~$ route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         192.168.124.1   0.0.0.0         UG    100    0        0 ens3
0.0.0.0         192.168.123.1   0.0.0.0         UG    101    0        0 ens5
10.0.0.0        0.0.0.0         255.255.255.0   U     0      0        0 ens5
19.0.0.0        0.0.0.0         255.255.255.0   U     0      0        0 ens4
20.0.0.0        0.0.0.0         255.0.0.0       U     0      0        0 l25
32.0.0.0        0.0.0.0         255.255.255.0   U     0      0        0 veth0
42.0.0.0        0.0.0.0         255.255.255.0   U     0      0        0 veth8
169.254.0.0     0.0.0.0         255.255.0.0     U     1000   0        0 vnet0
192.168.110.0   0.0.0.0         255.255.255.0   U     0      0        0 swovs3
192.168.118.0   0.0.0.0         255.255.255.0   U     0      0        0 sw6
192.168.119.0   0.0.0.0         255.255.255.0   U     0      0        0 virbr1
192.168.120.0   0.0.0.0         255.255.255.0   U     0      0        0 sw3
192.168.121.0   0.0.0.0         255.255.255.0   U     0      0        0 sw4
192.168.122.0   0.0.0.0         255.255.255.0   U     0      0        0 virbr0
192.168.123.0   0.0.0.0         255.255.255.0   U     100    0        0 ens5
192.168.124.0   0.0.0.0         255.255.255.0   U     100    0        0 ens3
```

```
ece792@ece792-Standard-PC-i440FX-PIIX-1996:~$ sudo iptables -t nat -L --line-number
Chain PREROUTING (policy ACCEPT)
num  target     prot opt source               destination

Chain INPUT (policy ACCEPT)
num  target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
num  target     prot opt source               destination

Chain POSTROUTING (policy ACCEPT)
num  target     prot opt source               destination
1    RETURN     all  --  192.168.118.0/24     base-address.mcast.net/24
2    RETURN     all  --  192.168.118.0/24     255.255.255.255
3    MASQUERADE  tcp  --  192.168.118.0/24    !192.168.118.0/24     masq ports: 1024-65535
4    MASQUERADE  udp  --  192.168.118.0/24    !192.168.118.0/24     masq ports: 1024-65535
5    MASQUERADE  all  --  192.168.118.0/24    !192.168.118.0/24
6    RETURN     all  --  192.168.121.0/24     base-address.mcast.net/24
7    RETURN     all  --  192.168.121.0/24     255.255.255.255
8    MASQUERADE  tcp  --  192.168.121.0/24    !192.168.121.0/24     masq ports: 1024-65535
9    MASQUERADE  udp  --  192.168.121.0/24    !192.168.121.0/24     masq ports: 1024-65535
10   MASQUERADE  all  --  192.168.121.0/24    !192.168.121.0/24
11   RETURN     all  --  192.168.110.0/24     base-address.mcast.net/24
12   RETURN     all  --  192.168.110.0/24     255.255.255.255
13   MASQUERADE  tcp  --  192.168.110.0/24    !192.168.110.0/24     masq ports: 1024-65535
14   MASQUERADE  udp  --  192.168.110.0/24    !192.168.110.0/24     masq ports: 1024-65535
15   MASQUERADE  all  --  192.168.110.0/24    !192.168.110.0/24
16   RETURN     all  --  192.168.119.0/24     base-address.mcast.net/24
17   RETURN     all  --  192.168.119.0/24     255.255.255.255
18   MASQUERADE  tcp  --  192.168.119.0/24    !192.168.119.0/24     masq ports: 1024-65535
19   MASQUERADE  udp  --  192.168.119.0/24    !192.168.119.0/24     masq ports: 1024-65535
20   MASQUERADE  all  --  192.168.119.0/24    !192.168.119.0/24
21   RETURN     all  --  192.168.122.0/24     base-address.mcast.net/24
22   RETURN     all  --  192.168.122.0/24     255.255.255.255
23   MASQUERADE  tcp  --  192.168.122.0/24    !192.168.122.0/24     masq ports: 1024-65535
24   MASQUERADE  udp  --  192.168.122.0/24    !192.168.122.0/24     masq ports: 1024-65535
25   MASQUERADE  all  --  192.168.122.0/24    !192.168.122.0/24
26   MASQUERADE  all  --  32.0.0.0/24         !32.0.0.0/24
27   MASQUERADE  all  --  42.0.0.0/24         !42.0.0.0/24
```

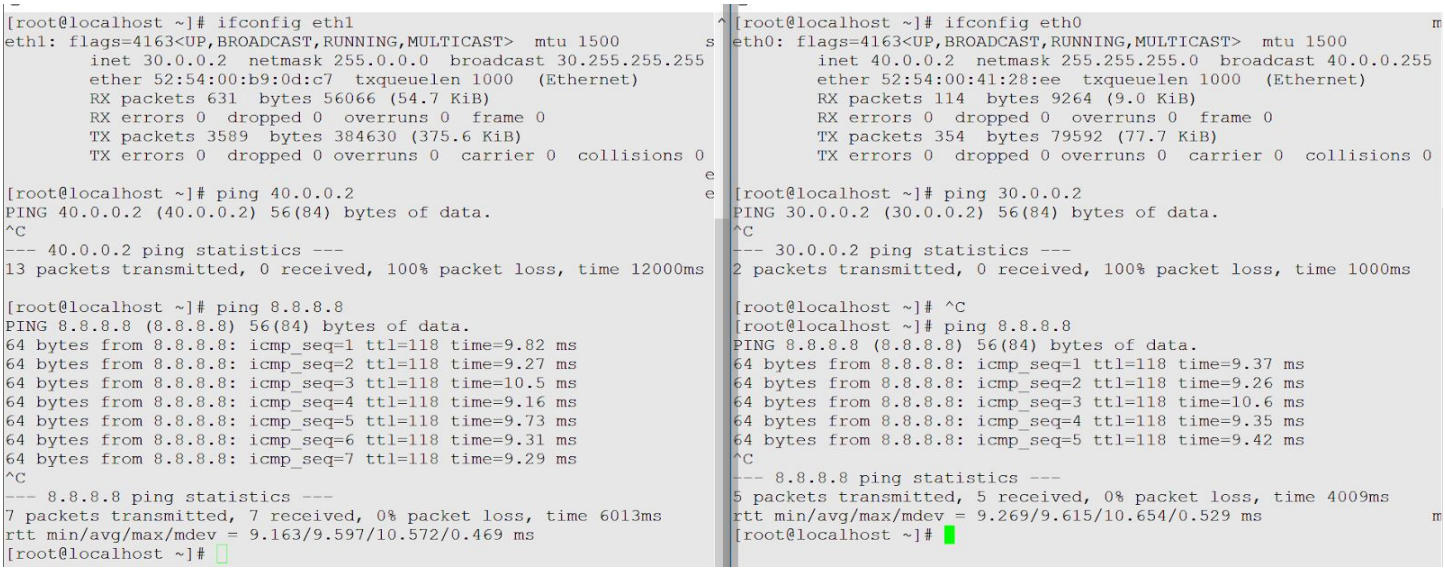
- Screenshot of iptables of blue_ns

```
ece792@ece792-Standard-PC-i440FX-PIIX-1996:~$ cat oct29/bluens.txt
# Generated by iptables-save v1.6.0 on Tue Oct 30 22:13:19 2018
*filter
:INPUT ACCEPT [4994:1635104]
:FORWARD ACCEPT [1072:96866]
:OUTPUT ACCEPT [78:6710]
COMMIT
# Completed on Tue Oct 30 22:13:19 2018
# Generated by iptables-save v1.6.0 on Tue Oct 30 22:13:19 2018
*nat
:PREROUTING ACCEPT [741:151160]
:INPUT ACCEPT [397:125336]
:OUTPUT ACCEPT [39:2948]
:POSTROUTING ACCEPT [41:3044]
-A POSTROUTING -s 30.0.0.0/24 ! -d 30.0.0.0/24 -j MASQUERADE
-A POSTROUTING -s 34.0.0.0/24 ! -d 34.0.0.0/24 -j MASQUERADE
COMMIT
# Completed on Tue Oct 30 22:13:19 2018
```

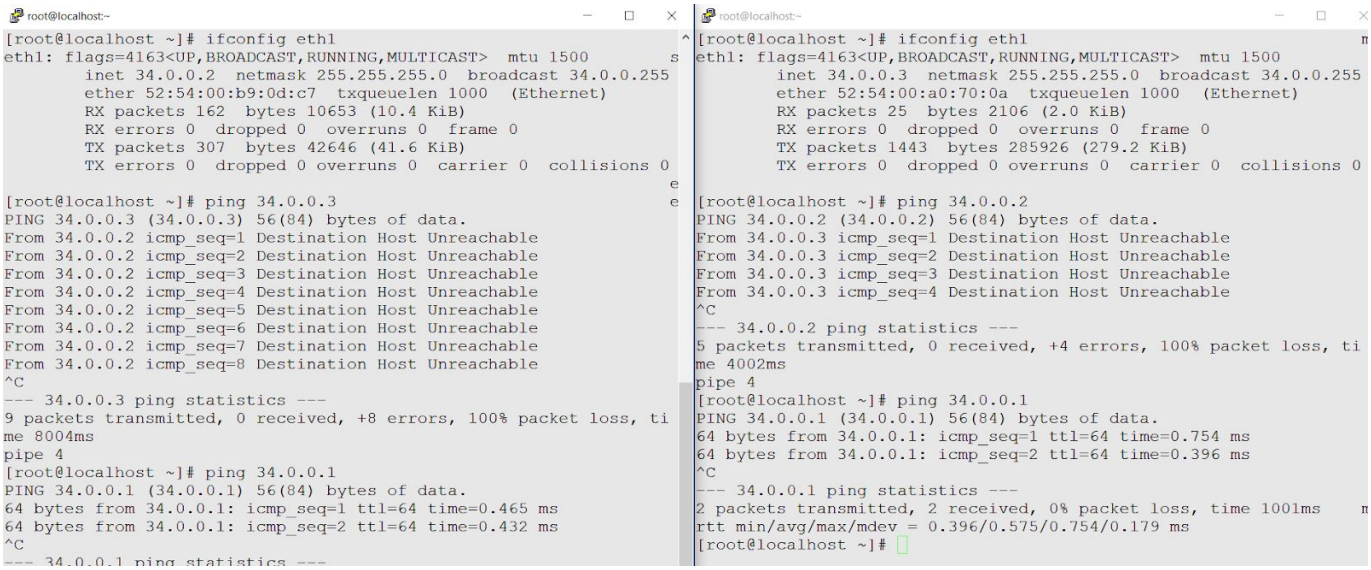
- Screenshot of iptables of red_ns

• VMs of tenant 1 should not be able to ping to tenant 2 VMs whether they have same or different IP subnets.

Screenshot of Tenant1 VM1 (30.0.0.2) tries to ping to Tenant2 VM1(40.0.0.2) (in different subnet):



Screenshot of Tenant1 VM2 (34.0.0.2) tries to ping to Tenant2 VM2(34.0.0.3) (when they are in the same subnet) but won't be able to ping:



• In another experiment, both tenants use one subnet that is common (e.g., 10.0.0.0/8) and one that is different. The hosts in the common subnet for tenant red and tenant blue should be able to ping the internet.

Tenant1VM2 (34.0.0.2) tries to ping to internet (google.com) and at the same time Tenant2VM2 (34.0.0.3) tries to ping to internet (google.com). As we have done postrouting at 2 places, the red and blue namespaces and the hypervisor, we do not observe any ambiguity when the reply comes from google.com.

```
[root@localhost ~]# ifconfig eth1
eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 34.0.0.2  netmask 255.255.255.0  broadcast 34.0.0.255
        ether 52:54:00:b9:0d:c7  txqueuelen 1000  (Ethernet)
        RX packets 176  bytes 11545 (11.2 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 337  bytes 45506 (44.4 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

[root@localhost ~]# ping googel.com
PING googel.com (172.217.7.196) 56(84) bytes of data.
64 bytes from iad30s10-in-f4.1e100.net (172.217.7.196): icmp_seq=1
 ttl=52 time=9.15 ms
64 bytes from iad30s10-in-f4.1e100.net (172.217.7.196): icmp_seq=2
 ttl=52 time=9.16 ms
^C
--- googel.com ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1001ms
rtt min/avg/max/mdev = 9.150/9.155/9.161/0.095 ms
[root@localhost ~]# ping google.com
PING google.com (172.217.15.78) 56(84) bytes of data.
64 bytes from iad23s63-in-f14.1e100.net (172.217.15.78): icmp_seq=
1 ttl=52 time=9.42 ms
64 bytes from iad23s63-in-f14.1e100.net (172.217.15.78): icmp_seq=
2 ttl=52 time=9.99 ms
64 bytes from iad23s63-in-f14.1e100.net (172.217.15.78): icmp_seq=
3 ttl=52 time=9.36 ms
64 bytes from iad23s63-in-f14.1e100.net (172.217.15.78): icmp_seq=
4 ttl=52 time=9.36 ms
64 bytes from iad23s63-in-f14.1e100.net (172.217.15.78): icmp_seq=
5 ttl=52 time=9.26 ms
^C
--- google.com ping statistics ---
```

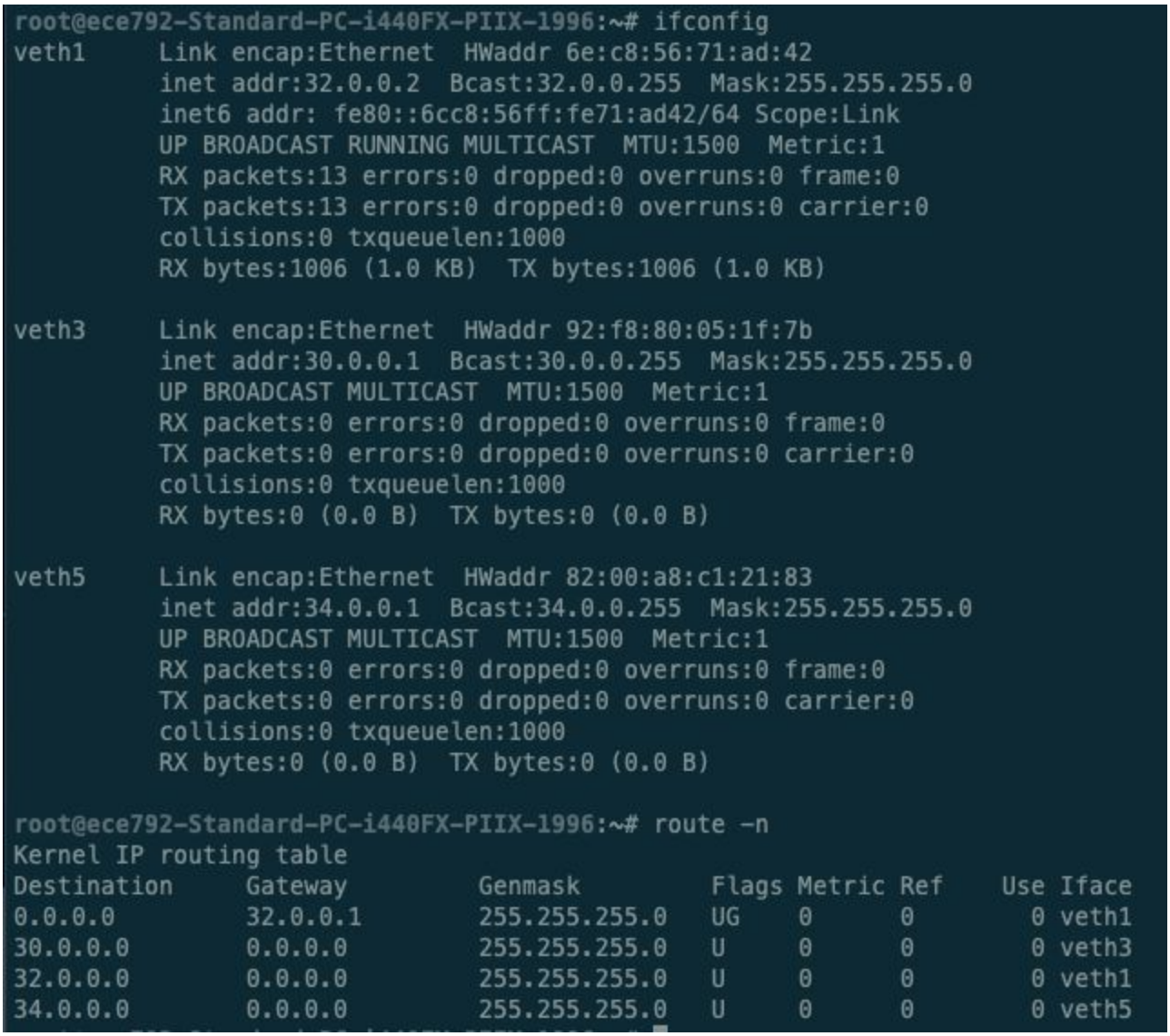
```
eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 34.0.0.3  netmask 255.255.255.0  broadcast 34.0.0.255
        ether 52:54:00:a0:70:0a  txqueuelen 1000  (Ethernet)
        RX packets 35  bytes 2770 (2.7 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 1469  bytes 290778 (283.9 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

[root@localhost ~]# ping google.com
PING google.com (172.217.15.78) 56(84) bytes of data.
64 bytes from iad23s63-in-f14.1e100.net (172.217.15.78): icmp_seq=
1 ttl=52 time=10.0 ms
64 bytes from iad23s63-in-f14.1e100.net (172.217.15.78): icmp_seq=
2 ttl=52 time=9.58 ms
64 bytes from iad23s63-in-f14.1e100.net (172.217.15.78): icmp_seq=
3 ttl=52 time=9.77 ms
64 bytes from iad23s63-in-f14.1e100.net (172.217.15.78): icmp_seq=
4 ttl=52 time=9.39 ms
64 bytes from iad23s63-in-f14.1e100.net (172.217.15.78): icmp_seq=
5 ttl=52 time=9.31 ms
64 bytes from iad23s63-in-f14.1e100.net (172.217.15.78): icmp_seq=
6 ttl=52 time=11.9 ms
64 bytes from iad23s63-in-f14.1e100.net (172.217.15.78): icmp_seq=
7 ttl=52 time=11.7 ms
64 bytes from iad23s63-in-f14.1e100.net (172.217.15.78): icmp_seq=
8 ttl=52 time=14.1 ms
64 bytes from iad23s63-in-f14.1e100.net (172.217.15.78): icmp_seq=
9 ttl=52 time=9.76 ms
64 bytes from iad23s63-in-f14.1e100.net (172.217.15.78): icmp_seq=
10 ttl=52 time=9.39 ms
64 bytes from iad23s63-in-f14.1e100.net (172.217.15.78): icmp_seq=
11 ttl=52 time=9.95 ms
64 bytes from iad23s63-in-f14.1e100.net (172.217.15.78): icmp_seq=
```

3. Forwarding table and IP tables in Host hypervisor. In this experiment, make sure not to use the hypervisor host's default forwarding table/IP tables. Configure a new network namespaces (call it provider ns) in the hypervisor. Implement and verify the following policies in the provider network namespace.

We assume the internet to be 54.0.0.2 which is present in a separate namespace called the provider_ns
The topology we use for the experiment is shown below:

Routes:
In blue_ns

```
sudo ip route add 30.0.0.0/24 dev veth3
sudo ip route add 34.0.0.0/24 dev veth5
sudo ip route add 0.0.0.0/0 via 32.0.0.1 dev veth1
```



In red_ns

```
sudo ip route add 40.0.0.0/24 dev veth7
sudo ip route add 34.0.0.0/24 dev veth10
sudo ip route add 0.0.0.0/0 via 42.0.0.1 dev veth9
```

```
root@ece792-Standard-PC-i440FX-PIIX-1996:~# ifconfig
veth7     Link encap:Ethernet  HWaddr de:5d:fa:56:95:a4
          inet addr:40.0.0.1  Bcast:40.0.0.255  Mask:255.255.255.0
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

veth9     Link encap:Ethernet  HWaddr 76:b1:88:1f:2f:72
          inet addr:42.0.0.2  Bcast:42.0.0.255  Mask:255.255.255.0
          inet6 addr: fe80::74b1:88ff:fe1f:2f72/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:13 errors:0 dropped:0 overruns:0 frame:0
          TX packets:13 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:1006 (1.0 KB)  TX bytes:1006 (1.0 KB)

veth10    Link encap:Ethernet  HWaddr 1a:f7:e8:88:3d:34
          inet addr:34.0.0.1  Bcast:34.0.0.255  Mask:255.255.255.0
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

root@ece792-Standard-PC-i440FX-PIIX-1996:~# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         42.0.0.1        255.255.255.0   UG    0      0        0 veth9
34.0.0.0        0.0.0.0         255.255.255.0   U     0      0        0 veth10
40.0.0.0        0.0.0.0         255.255.255.0   U     0      0        0 veth7
42.0.0.0        0.0.0.0         255.255.255.0   U     0      0        0 veth9
```

In provider_ns

```
sudo ip route add 0.0.0.0/0 via 54.0.0.1 dev veth12
```

```
root@ece792-Standard-PC-i440FX-PIIX-1996:~# ifconfig
veth0     Link encap:Ethernet  HWaddr 66:71:a4:b0:74:d2
          inet addr:32.0.0.1  Bcast:32.0.0.255  Mask:255.255.255.0
          inet6 addr: fe80::6471:a4ff:feb0:74d2/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:13 errors:0 dropped:0 overruns:0 frame:0
          TX packets:13 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:1006 (1.0 KB)  TX bytes:1006 (1.0 KB)

veth8     Link encap:Ethernet  HWaddr 22:69:3f:0a:17:89
          inet addr:42.0.0.1  Bcast:42.0.0.255  Mask:255.255.255.0
          inet6 addr: fe80::2069:3fff:fe0a:1789/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:13 errors:0 dropped:0 overruns:0 frame:0
          TX packets:13 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:1006 (1.0 KB)  TX bytes:1006 (1.0 KB)

veth12    Link encap:Ethernet  HWaddr 12:39:5d:fe:0b:0b
          inet addr:54.0.0.2  Bcast:54.0.0.255  Mask:255.255.255.0
          inet6 addr: fe80::1039:5dff:fefe:b0b/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:40 errors:0 dropped:0 overruns:0 frame:0
          TX packets:13 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:5285 (5.2 KB)  TX bytes:1006 (1.0 KB)

root@ece792-Standard-PC-i440FX-PIIX-1996:~# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         54.0.0.1        255.255.255.0   UG    0      0        0 veth12
32.0.0.0        0.0.0.0         255.255.255.0   U     0      0        0 veth0
42.0.0.0        0.0.0.0         255.255.255.0   U     0      0        0 veth8
54.0.0.0        0.0.0.0         255.255.255.0   U     0      0        0 veth12
```

(a) Internet policy. Allow ICMP traffic for both tenants. Allow SSH traffic for only the blue tenant.

blue_ns

```
sudo iptables -t nat -A POSTROUTING -s 30.0.0.0/24 ! -d 30.0.0.0/24 -j MASQUERADE
sudo iptables -t nat -A POSTROUTING -s 34.0.0.0/24 ! -d 34.0.0.0/24 -j MASQUERADE
```

red_ns

```
sudo iptables -t nat -A POSTROUTING -s 40.0.0.0/24 ! -d 40.0.0.0/24 -j MASQUERADE
sudo iptables -t nat -A POSTROUTING -s 34.0.0.0/24 ! -d 34.0.0.0/24 -j MASQUERADE
sudo iptables -I FORWARD 1 -p TCP -d 54.0.0.0/24 --dport 22 -j DROP
```

inside provider_ns

```
sudo iptables -t nat -A POSTROUTING -p ICMP -s 32.0.0.0/24 ! -d 32.0.0.0/24 -j
```

```
MASQUERADE
sudo iptables -t nat -A POSTROUTING -p ICMP -s 42.0.0.0/24 ! -d 42.0.0.0/24 -j
MASQUERADE
```

- Screenshot of iptables inside the **provider_ns**

```
# Generated by iptables-save v1.6.0 on Tue Oct 30 22:07:09 2018
*filter
:INPUT ACCEPT [2:120]
:FORWARD ACCEPT [861:81426]
:OUTPUT ACCEPT [145:9779]
COMMIT
# Completed on Tue Oct 30 22:07:09 2018
# Generated by iptables-save v1.6.0 on Tue Oct 30 22:07:09 2018
*nat
:PREROUTING ACCEPT [6:392]
:INPUT ACCEPT [1:60]
:OUTPUT ACCEPT [6:402]
:POSTROUTING ACCEPT [11:734]
-A POSTROUTING -s 32.0.0.0/24 ! -d 32.0.0.0/24 -p icmp -j MASQUERADE
-A POSTROUTING -s 42.0.0.0/24 ! -d 42.0.0.0/24 -p icmp -j MASQUERADE
-A POSTROUTING -s 32.0.0.0/24 ! -d 32.0.0.0/24 -p icmp -j MASQUERADE
-A POSTROUTING -s 42.0.0.0/24 ! -d 42.0.0.0/24 -p icmp -j MASQUERADE
-A POSTROUTING -s 32.0.0.0/24 ! -d 32.0.0.0/24 -p icmp -j MASQUERADE
-A POSTROUTING -s 42.0.0.0/24 ! -d 42.0.0.0/24 -p icmp -j MASQUERADE
COMMIT
# Completed on Tue Oct 30 22:07:09 2018
```

- Screenshot of iptables inside **blue_ns**

```
# Generated by iptables-save v1.6.0 on Tue Oct 30 22:13:19 2018
*filter
:INPUT ACCEPT [4994:1635104]
:FORWARD ACCEPT [1072:96866]
:OUTPUT ACCEPT [78:6710]
COMMIT
# Completed on Tue Oct 30 22:13:19 2018
# Generated by iptables-save v1.6.0 on Tue Oct 30 22:13:19 2018
*nat
:PREROUTING ACCEPT [741:151160]
:INPUT ACCEPT [397:125336]
:OUTPUT ACCEPT [39:2948]
:POSTROUTING ACCEPT [41:3044]
-A POSTROUTING -s 30.0.0.0/24 ! -d 30.0.0.0/24 -j MASQUERADE
-A POSTROUTING -s 34.0.0.0/24 ! -d 34.0.0.0/24 -j MASQUERADE
COMMIT
# Completed on Tue Oct 30 22:13:19 2018
```

- Screenshot of iptables inside **red_ns**

```
# Generated by iptables-save v1.6.0 on Tue Oct 30 22:24:26 2018
*filter
:INPUT ACCEPT [19:6232]
:FORWARD ACCEPT [10:600]
:OUTPUT ACCEPT [8:536]
-A FORWARD -d 54.0.0.0/24 -p tcp -m tcp --dport 22 -j DROP
COMMIT
# Completed on Tue Oct 30 22:24:26 2018
# Generated by iptables-save v1.6.0 on Tue Oct 30 22:24:26 2018
*nat
:PREROUTING ACCEPT [925:167140]
:INPUT ACCEPT [396:128424]
:OUTPUT ACCEPT [25:1696]
:POSTROUTING ACCEPT [26:1756]
-A POSTROUTING -s 40.0.0.0/24 ! -d 40.0.0.0/24 -j MASQUERADE
-A POSTROUTING -s 34.0.0.0/24 ! -d 34.0.0.0/24 -j MASQUERADE
COMMIT
# Completed on Tue Oct 30 22:24:26 2018
```

- ICMP Traffic for Blue Tenant

```
[root@localhost ~]# ifconfig eth1
eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 30.0.0.2  netmask 255.255.255.0  broadcast 30.0.0.255
        ether 52:54:00:c5:5c:53  txqueuelen 1000  (Ethernet)
        RX packets 698  bytes 69605 (67.9 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 1243  bytes 226093 (220.7 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

[root@localhost ~]# ping 54.0.0.1
PING 54.0.0.1 (54.0.0.1) 56(84) bytes of data.
64 bytes from 54.0.0.1: icmp_seq=1 ttl=62 time=0.590 ms
64 bytes from 54.0.0.1: icmp_seq=2 ttl=62 time=0.467 ms
64 bytes from 54.0.0.1: icmp_seq=3 ttl=62 time=0.448 ms
^C
--- 54.0.0.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2000ms
rtt min/avg/max/mdev = 0.448/0.501/0.590/0.068 ms
```

- ICMP Traffic for Red Tenant

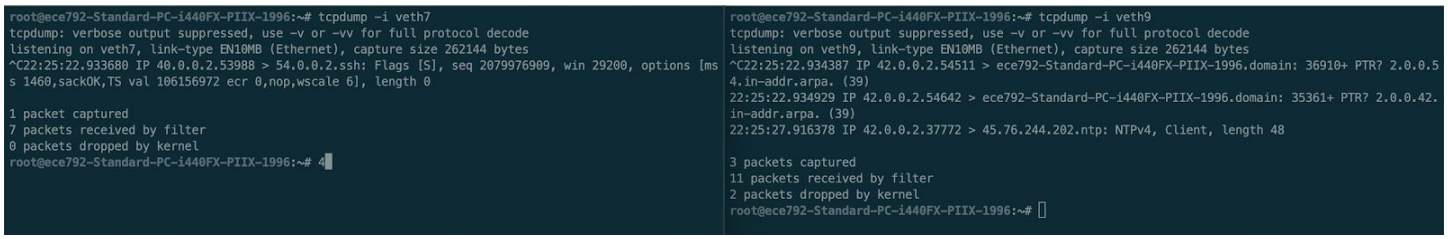
```
[root@localhost ~]# ifconfig eth1
eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 40.0.0.2  netmask 255.255.255.0  broadcast 40.0.0.255
        ether 52:54:00:7b:50:b5  txqueuelen 1000  (Ethernet)
        RX packets 862  bytes 77967 (76.1 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 3421  bytes 759531 (741.7 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

[root@localhost ~]# ping 54.0.0.1
PING 54.0.0.1 (54.0.0.1) 56(84) bytes of data.
64 bytes from 54.0.0.1: icmp_seq=1 ttl=62 time=0.531 ms
64 bytes from 54.0.0.1: icmp_seq=2 ttl=62 time=0.299 ms
^C
--- 54.0.0.1 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1000ms
rtt min/avg/max/mdev = 0.299/0.415/0.531/0.116 ms
```

- Screenshot of Tenant1 (blue subnet) who is able to ssh to 54.0.0.1

```
[root@localhost ~]# ifconfig eth1
eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 34.0.0.2  netmask 255.255.255.0  broadcast 34.0.0.255
        ether 52:54:00:b9:0d:c7  txqueuelen 1000  (Ethernet)
        RX packets 439  bytes 34499 (33.6 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 979  bytes 179432 (175.2 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

[root@localhost ~]# ssh ece792@54.0.0.1
ece792@54.0.0.1's password:
Welcome to Ubuntu 16.04.3 LTS (GNU/Linux 4.10.0-28-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

250 packages can be updated.
8 updates are security updates.

New release '18.04.1 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Last login: Sun Oct 28 18:21:42 2018 from 192.168.122.109
ece792@ece792-Standard-PC-i440FX-PIIX-1996:~$
```

- Following Screenshot of a VM in red_ns, which can ping an IP in the internet, but can't ssh to 54.0.0.1 host or any other VM of a different tenant.

For the above, the following screenshot shows that, the SSH packet is being received at veth7 but it is dropped at red_ns and not able to capture at veth9.

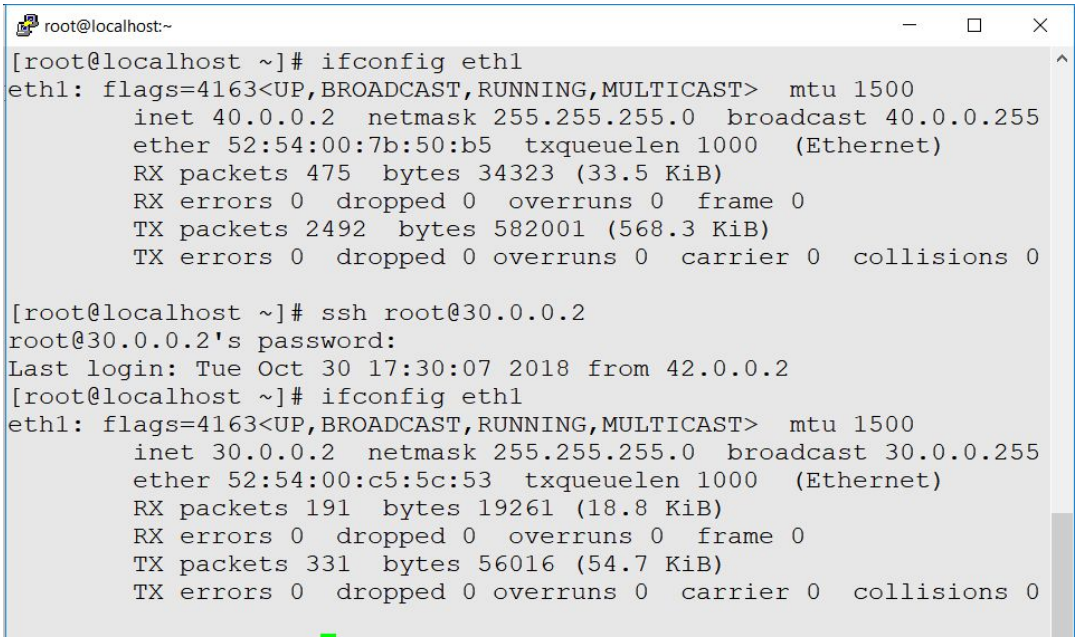


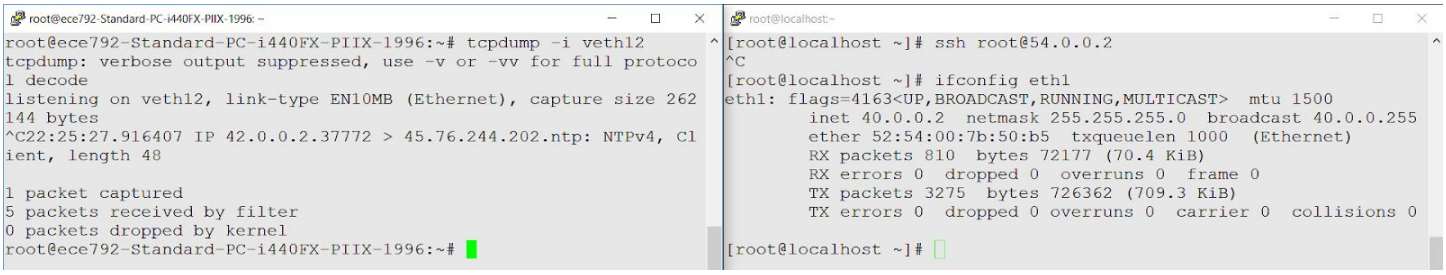
(b) Local L3 policy. Allow red tenant and blue tenant to ssh each other's VM, provided the subnets are different.

We add routes to the subnets which are not common among both the tenants at provider_ns to ensure that they can SSH to each other's VMs.

```
sudo ip route add 30.0.0.0/24 via 32.0.0.2 dev veth0
sudo ip route add 40.0.0.0/24 via 42.0.0.2 dev veth8
```

The below is a screenshot of tenant2's VM which is not in the common subnet, being able to ssh into the tenant1's VM.





The above screenshot displays that the ssh traffic from 40.0.0.2 doesn't reach the internet(54.0.0.2). Thus we have prevented SSH traffic from going out of the red_ns with the drop rule at red_ns.

Reference:

- https://askubuntu.com/questions/466445/what-is-masquerade-in-the-context-of-iptables
- https://www.crybit.com/how-to-save-current-iptables-rules/
- https://serverfault.com/questions/201186/iptables-forwarding-ssh-ports
- https://blog.scottlowe.org/2013/09/04/introducing-linux-network-namespaces/