

Overlay Network and VXLAN

计算机网络 CS339

李子龙 518070910095

2021 年 12 月 7 日

目录

1 建立网络	2
2 Wireshark 抓包	3
3 ping 测试	4
3.1 bug 声明	4
3.2 物理IP级测试	6
3.3 交换机级测试	6
3.4 交换机–主机级测试	7
3.5 连通性测试小结	8
4 iperf 测试	9
4.1 物理IP级测试	9
4.2 交换机级测试	10
4.3 交换机–主机级测试	10
4.4 带宽测试小结	11

An overlay network can be thought of as a computer network on top of another network.
VXLAN is often described as an overlay technology because it allows to stretch Layer 2 connections over an intervening Layer 3 network by encapsulating (tunneling) Ethernet frames in a VXLAN packet that includes IP addresses.

本文先进行 ping 测试，后进行 iperf 测试。

1 建立网络

先克隆虚拟机¹，然后分别在 VM1 和 VM2 上分别运行 `vm1topo.py` 和 `vm2topo.py`（与 `vm1topo.py` 类似，略过），得到如图 1 所示的拓扑结构，注意拓扑已经与教程中

Listing 1: `vm1topo.py`

```
1  from mininet.cli import CLI
2  from mininet.link import TCLink
3  from mininet.node import CPULimitedHost
4  from mininet.topo import Topo
5  from mininet.net import Mininet
6  from mininet.log import lg, info
7  from mininet.util import dumpNodeConnections, run
8
9  vm1ip = "192.168.4.131"          #
10 vm2ip = "192.168.4.132"          #
11
12 class VM1Topo(Topo):
13     "Topology of VM1."
14
15     def build(self):
16         switch1 = self.addSwitch('s1', ip='10.0.0.101')
17         host1 = self.addHost('h1', cpu=.25, ip='10.0.0.1')
18         host2 = self.addHost('h2', cpu=.25, ip='10.0.0.2')
19         self.addLink(host1, switch1, use_htb=True)
20         self.addLink(host2, switch1, use_htb=True)
21
22     if __name__=="__main__":
23         topo = VM1Topo()
24         net = Mininet(topo=topo, host=CPULimitedHost, link=TCLink, autoStaticArp=True)
25         net.start()
26         dumpNodeConnections(net.hosts)
27
28         run('ifconfig s1 10.0.0.101/8 up')
29         run('ovs-vsctl del-br br1')
30         run('ovs-vsctl add-br br1')
31         run('ifconfig ens33 0 up')
32         run('ovs-vsctl add-port br1 ens33')
33         run('ifconfig br1 ' + vm1ip + '/24 up')
34
35     # set up VxLAN
36     run('ovs-vsctl add-port s1 vxlan0 -- set interface vxlan0 type=vxlan
options:remote_ip=' + vm2ip + ' option:key=100 ofport_request=10')
37
38     # config MTU
39     run('ifconfig vxlan_sys_4789 mtu 1500')
40     s1 = net.switches[0]
41     h1, h2 = net.hosts
42     h1.cmdPrint('ifconfig h1-eth0 mtu 1450')
43     h2.cmdPrint('ifconfig h2-eth0 mtu 1450')
44     s1.cmdPrint('ifconfig s1-eth1 mtu 1450')
```

¹感谢 VMWare Workstation 的快照技术，如果重新启动虚拟机，网络配置将会让其无法上网，必须返回原点重新配置。

```
45     s1.cmdPrint('ifconfig s1-eth2 mtu 1450')
46
47     # TBD: set up flow-table manually
48
49     CLI(net)
50     net.stop()
```

VM1 运行

```
python vm1topo.py
```

VM2 运行

```
python vm2topo.py
```

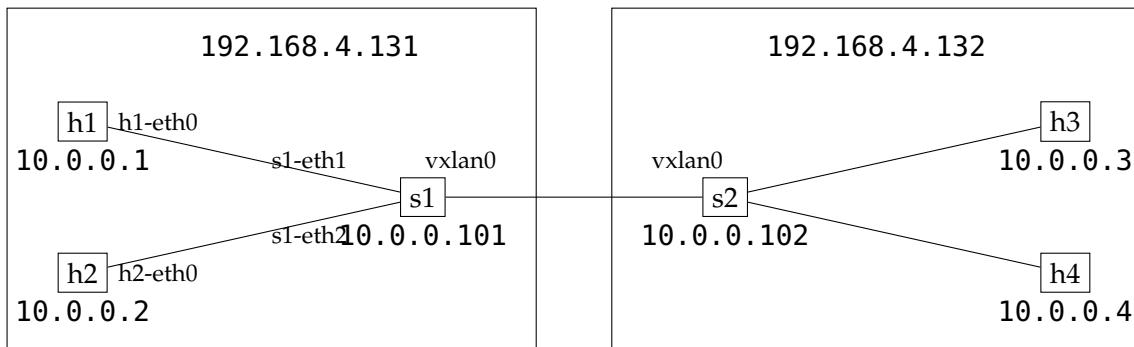


图 1: 拓扑结构

2 WIRESHARK 抓包

Use Wireshark to monitor the interfaces s1 and eth0, and describe your findings.

使用 Wireshark 抓取传输时数据，得到如图 2 所示的包。外层为 UDP 报文，显示的实际的 IP 地址；中间为 VXLAN 层；再内侧为 Overlay 的 ICMP 报文，显示的是 Overlay IP 地址。

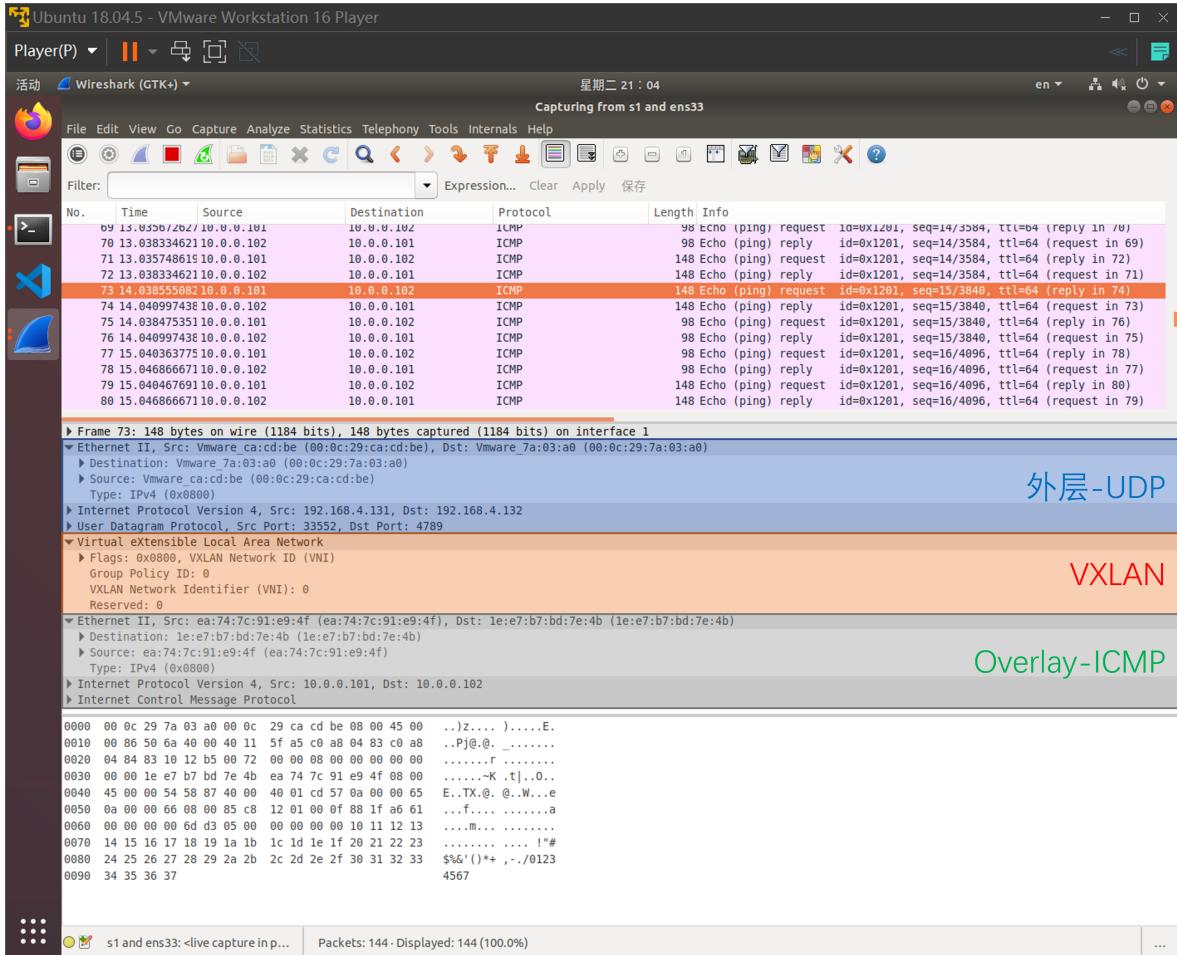


图 2: Wireshark 抓包

3 PING 测试

Similar to Q2, use ping to test the network latency and analyze your results.

3.1 BUG 声明

如果不调整 MTU 会导致从 h1 ping 出去的时候只有两个包被接收了，之后会提示“没有可用的缓冲区空间”，如图 3 所示。这是因为 VXLAN 会添加 50 — 54 Bytes 的额外头部，导致超出 MTU 限制。现在的 bug 是，已经调整 MTU 后，仍然无法 ping 通。

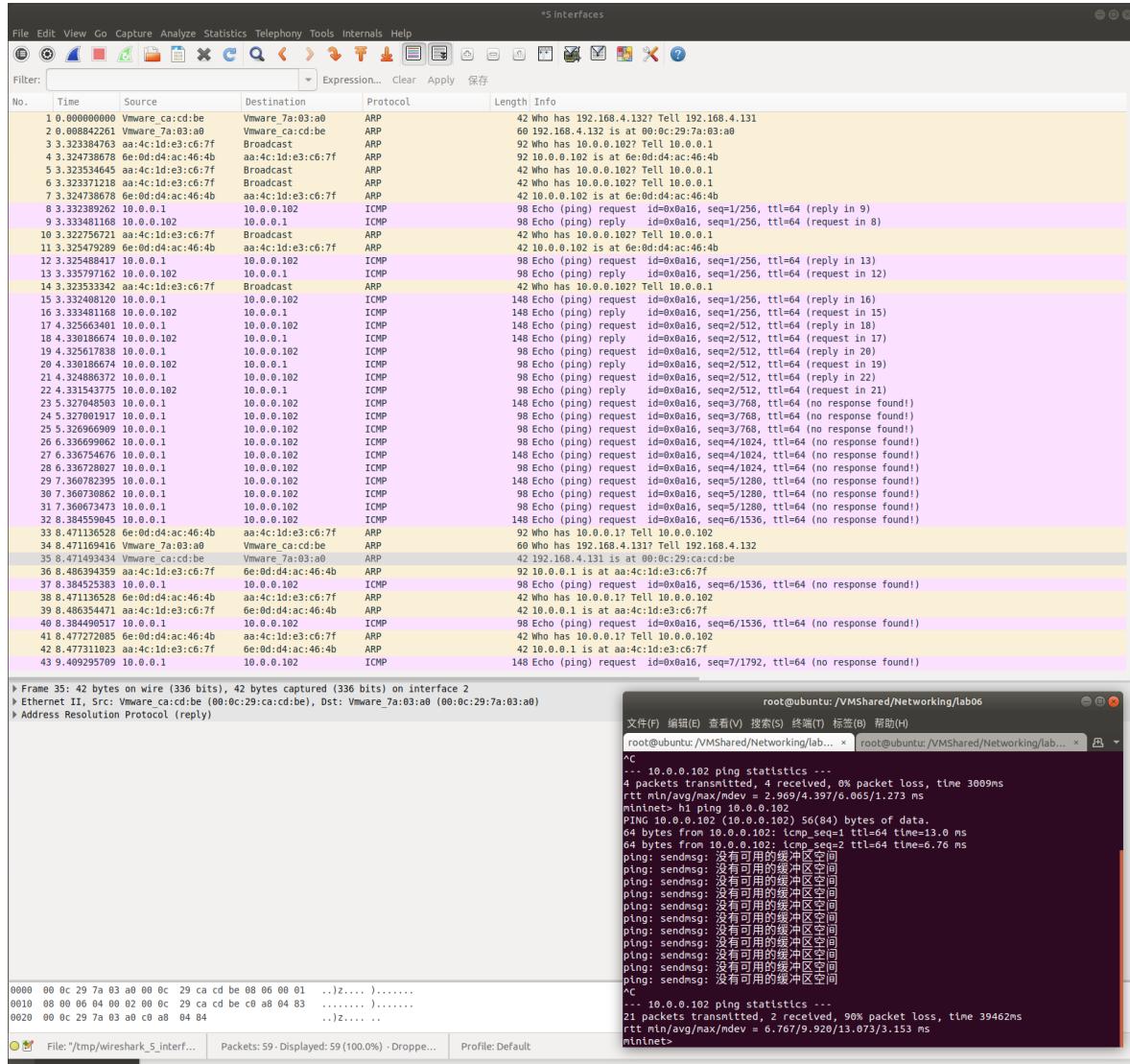


图 3: 无缓冲区空间抓包情况

更进一步的测试表明，同一条路径上 **s1-h3**，如果是 **s1** 为主动 ping 方，可以 ping 通，而如果是 **h3** 为主动方，则 ping 不通，如图 4 所示。疑似子网内的主机无法被 **VxLAN** 识别以正确发包，或者有机制缺陷，导致只能 ping 通两次而没有释放缓冲区。

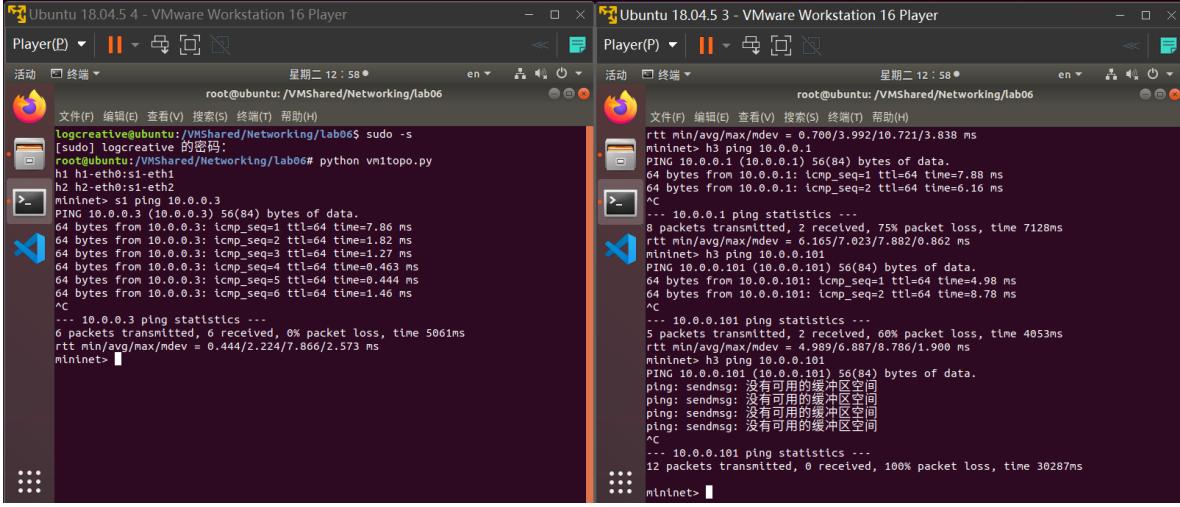


图 4: 同一路径不同发送主动方测试

下面暂时解决方法是直接将服务请求方全部放在交换机上。

3.2 物理IP级测试

两个虚拟机分别向对方的真实 IP 进行连通性测试。

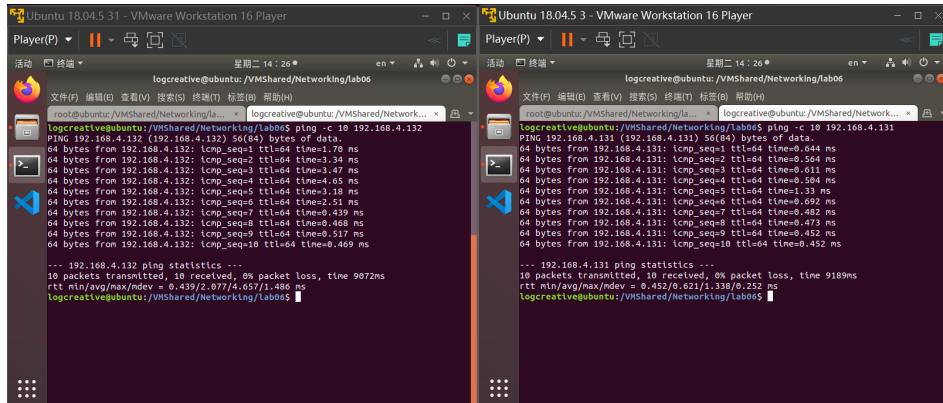


图 5: 物理IP级测试

3.3 交换机级测试

两个虚拟机的mininet交换机分别向对方的交换机进行连通性测试。

The screenshot shows two terminal windows side-by-side. Both are running on Ubuntu 18.04.5. The left window is titled 'Ubuntu 18.04.5 31 - VMware Workstation 16 Player' and the right window is titled 'Ubuntu 18.04.5 3 - VMware Workstation 16 Player'. Both terminals are connected to a virtual machine named 'lab06'. In the left terminal, the user runs 'logcreative' to capture logs, then uses 'vnztoppo.py' to analyze the captured traffic. The output shows a ping from 10.0.0.102 to 10.0.0.101. In the right terminal, the user runs 'logcreative' and then 'vnztoppo.py' to analyze the same traffic. The output shows a ping from 10.0.0.101 to 10.0.0.102. Both terminals show detailed network statistics and packet timing.

图 6: 交换机级测试

3.4 交换机-主机级测试

从交换机向对方虚拟机的主机请求连通性测试。

The screenshot shows two terminal windows side-by-side, both running on Ubuntu 18.04.5. The left window is titled 'Ubuntu 18.04.5 31 - VMware Workstation 16 Player' and the right window is titled 'Ubuntu 18.04.5 3 - VMware Workstation 16 Player'. Both terminals are connected to a virtual machine named 'lab06'. In the left terminal, the user runs 'logcreative' to capture logs, then uses 'vnztoppo.py' to analyze the captured traffic. The output shows a ping from 10.0.0.102 to 10.0.0.101. In the right terminal, the user runs 'logcreative' and then 'vnztoppo.py' to analyze the same traffic. The output shows a ping from 10.0.0.101 to 10.0.0.102. Both terminals show detailed network statistics and packet timing.

图 7: 交换机-主机级测试

3.5 连通性测试小结

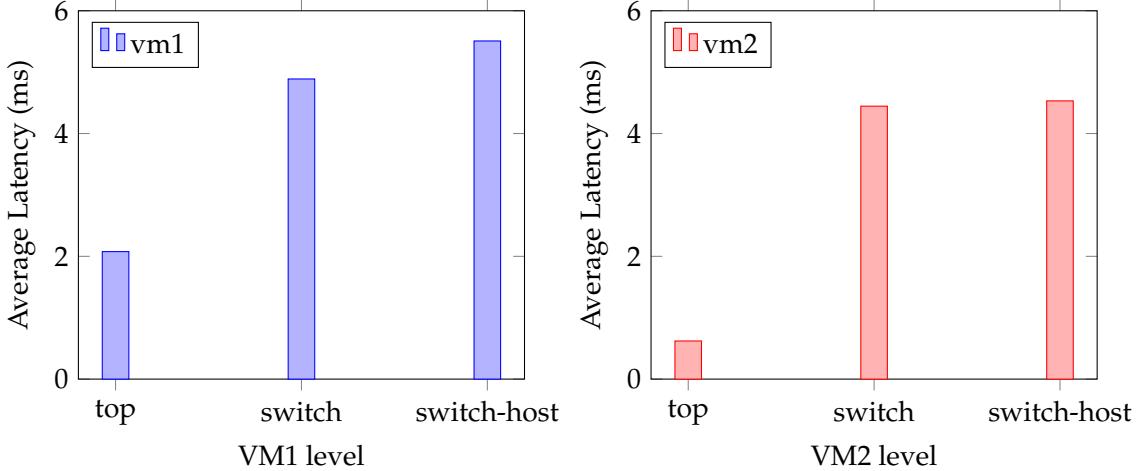


图 8: VM1发出连通性平均延迟

图 9: VM2发出连通性测试

VM1 的连通性测试小结如图 8 所示，VM2 如图 9 所示。由于 VM1 是链接克隆 VM2 所以会有一定的性能损失。¹¹但是整体而言，直接物理 IP 的延迟最小，因为 ping一秒一个且传输较小的包来测试连通性，这个时候交换机级 VxLAN 封装带来的延迟会占上风，ARP需要多次转换，如图 11；而物理IP支持各种特性会使这种时延下降，不需要地址转换，让ARP包更加简洁，如图 10。

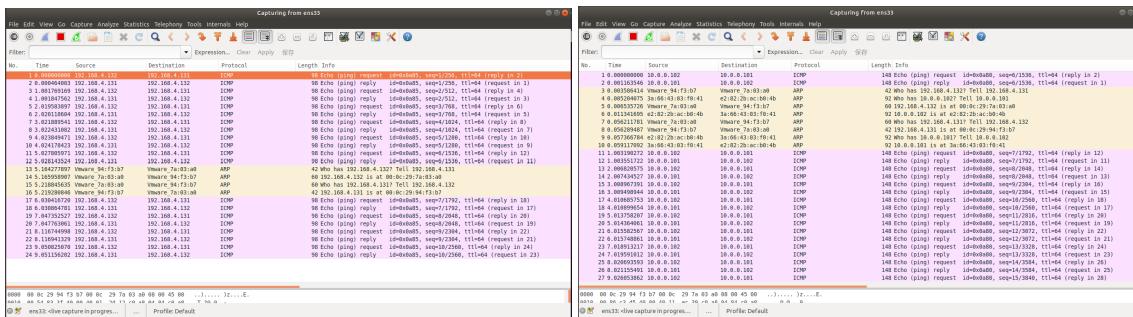


图 10: 物理 IP 抓包情况

图 11: 交换机级抓包情况

4 IPERF 测试

Use iperf to test the network bandwidth between the two virtual machines

- Test the bandwidth between 192.168.56.127 and 192.168.56.128
- Test the bandwidth between 10.0.0.1/10.0.0.2/10.0.0.101 and 10.0.0.102 (hint: you may need to specify a reasonable MTU size in order for your iperf to work in this case. Please also think about why.)

Compare the above results and explain the reason.

服务器

```
iperf -s
```

客户机

```
iperf -c [IP]
```

4.1 物理IP级测试

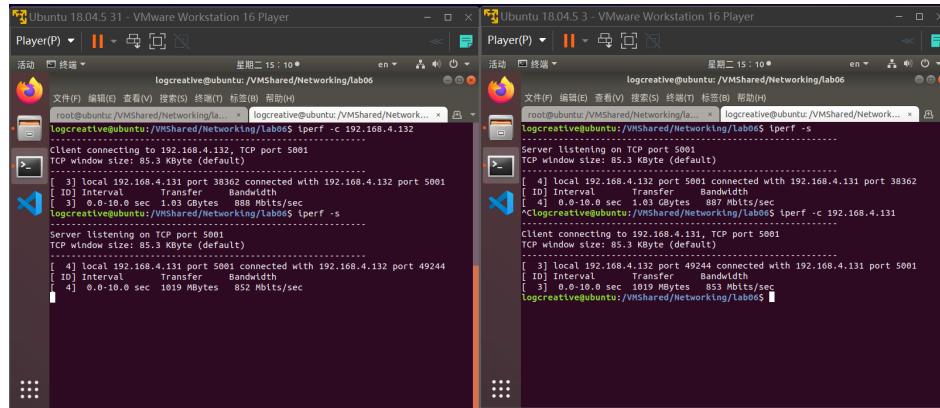
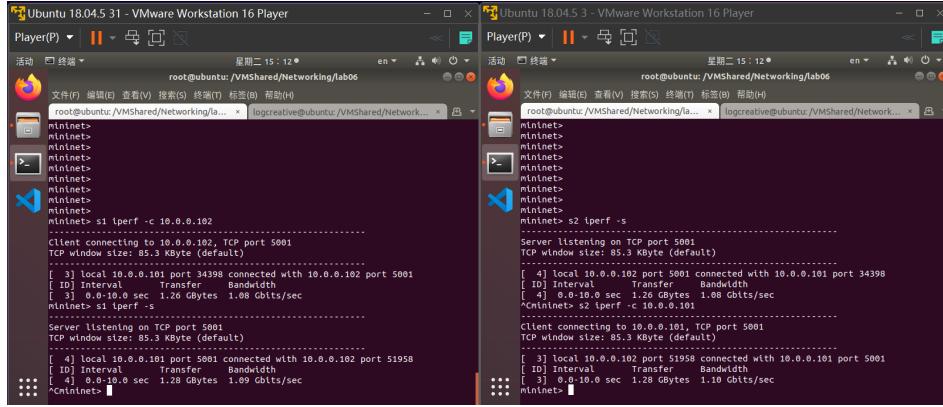


图 12: 物理IP级测试

4.2 交换机级测试



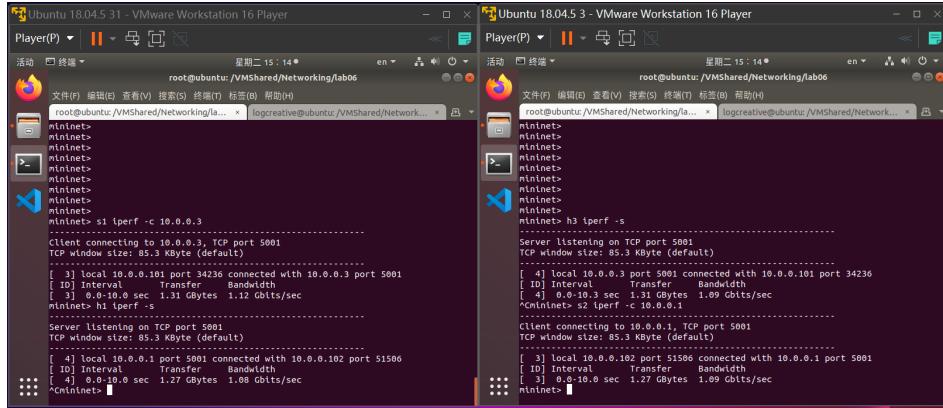
The screenshot shows two terminal windows from the VMware Workstation 16 Player interface. Both windows are running on Ubuntu 18.04.5. The left window (IP 10.0.0.3) has a command prompt for 'root@ubuntu:~/VMShared/Networking/lab06'. It runs 'iperf -s' to start a server on port 5001. The right window (IP 10.0.0.102) has a command prompt for 'root@ubuntu:~/VMShared/Networking/lab06'. It runs 'iperf -c 10.0.0.3' to connect to the server. The output shows a bandwidth of 1.08 Gbytes/sec.

```
root@ubuntu:~/VMShared/Networking/lab06
iperf -s
[ 4] local 10.0.0.3 port 5001
[ 4] Interval Transfer Bandwidth
[ 4] 0.0-10.0 sec 1.26 Gbytes 1.08 Gbytes/sec
mininet> s1 iperf -s
[ 4] local 10.0.0.102 port 5001 connected with 10.0.0.3 port 5001
[ 4] Interval Transfer Bandwidth
[ 4] 0.0-10.0 sec 1.26 Gbytes 1.08 Gbytes/sec
mininet>
```

```
root@ubuntu:~/VMShared/Networking/lab06
iperf -c 10.0.0.3
[ 3] local 10.0.0.102 port 51958 connected with 10.0.0.3 port 5001
[ 3] Interval Transfer Bandwidth
[ 3] 0.0-10.0 sec 1.26 Gbytes 1.08 Gbytes/sec
mininet>
```

图 13: 交换机级测试

4.3 交换机-主机级测试



The screenshot shows two terminal windows from the VMware Workstation 16 Player interface. Both windows are running on Ubuntu 18.04.5. The left window (IP 10.0.0.3) has a command prompt for 'root@ubuntu:~/VMShared/Networking/lab06'. It runs 'iperf -s' to start a server on port 5001. The right window (IP 10.0.0.1) has a command prompt for 'root@ubuntu:~/VMShared/Networking/lab06'. It runs 'iperf -c 10.0.0.3' to connect to the server. The output shows a bandwidth of 1.09 Gbytes/sec.

```
root@ubuntu:~/VMShared/Networking/lab06
iperf -s
[ 4] local 10.0.0.3 port 5001
[ 4] Interval Transfer Bandwidth
[ 4] 0.0-10.0 sec 1.26 Gbytes 1.08 Gbytes/sec
mininet> h1 iperf -s
[ 4] local 10.0.0.1 port 5001 connected with 10.0.0.3 port 5001
[ 4] Interval Transfer Bandwidth
[ 4] 0.0-10.0 sec 1.26 Gbytes 1.08 Gbytes/sec
mininet>
```

```
root@ubuntu:~/VMShared/Networking/lab06
iperf -c 10.0.0.3
[ 3] local 10.0.0.1 port 51506 connected with 10.0.0.3 port 5001
[ 3] Interval Transfer Bandwidth
[ 3] 0.0-10.0 sec 1.27 Gbytes 1.09 Gbytes/sec
mininet>
```

图 14: 交换机-主机测试

4.4 带宽测试小结

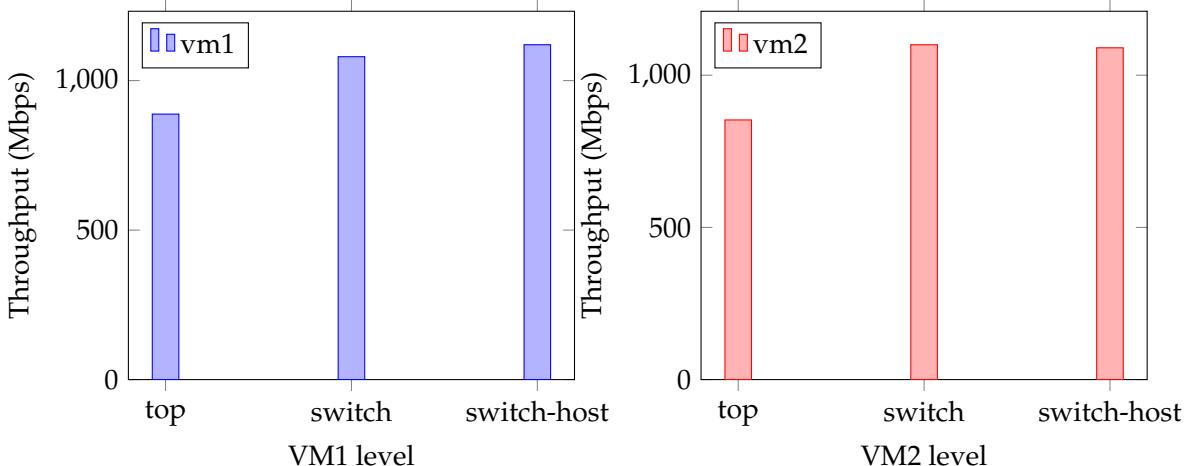


图 15: VM1发出带宽测试

图 16: VM2发出带宽测试

VM1 和 VM2 的带宽测试小结分别如图 15 和图 16 所示。与 ping 不同的是，交换机级与交换机-主机级的带宽较高²，原因是VxLAN是交换机的直接端口，连续发包的iperf会让这个直接优势占上风（物理IP发包需要经过交换机）。注意相较于ping的一秒一个包有一定的区别，并且包会稍大。

附录

表 1: 运行环境

操作系统	Ubuntu 18.04.5
虚拟软件	VMWare Workstation 16.0

参考文献

- [1] VMWare. 使用链接克隆[M/OL]. 2019. <https://docs.vmware.com/cn/VMware-Workstation-Pro/16.0/com.vmware.ws.using.doc/GUID-BA264A65-C50F-4345-A787-DCC5C5324DD1.html>.

²事实上，物理机的带宽没有那么稳定，有时候会达到1.09Gbps。