Overlay Network and VXLAN (With Bug)

计算机网络 CS339

李子龙 518070910095

2021 年 12 月 7 日

目录

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.

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

```python
from mininet.cli import CLI
from mininet.link import TCLink
from mininet.node import CPULimitedHost
from mininet.topo import Topo
from mininet.net import Mininet
from mininet.log import lg, info
from mininet.util import dumpNodeConnections, run

vm1ip = "192.168.4.131"         #
vm2ip = "192.168.4.132"         #

class VM1Topo(Topo):
    "Topology of VM1."

    def build(self):
        switch1 = self.addSwitch('s1', ip='10.0.0.101')
        host1 = self.addHost('h1', cpu=.25, ip='10.0.0.1')
        host2 = self.addHost('h2', cpu=.25, ip='10.0.0.2')
        self.addLink(host1, switch1, use_htb=True)
        self.addLink(host2, switch1, use_htb=True)

if __name__=="__main__":
    topo = VM1Topo()
    net = Mininet(topo=topo,host=CPULimitedHost, link=TCLink, autoStaticArp=
    True)
    net.start()
    dumpNodeConnections(net.hosts)

    run('ifconfig s1 10.0.0.101/8 up')
    run('ovs-vsctl del-br br1')
    run('ovs-vsctl add-br br1')
    run('ifconfig ens33 0 up')
    run('ovs-vsctl add-port br1 ens33')
    run('ifconfig br1 ' + vm1ip +  '/24 up')

    # set up VxLAN
    run('ovs-vsctl add-port s1 vxlan0 -- set interface vxlan0 type=vxlan
    options:remote_ip=' + vm2ip + ' option:key=100 ofport_request=10')

    # config MTU
    run('ifconfig vxlan_sys_4789 mtu 1450')
    s1 = net.switches[0]
    h1, h2 = net.hosts
    h1.cmdPrint('ifconfig h1-eth0 mtu 1450')
    h2.cmdPrint('ifconfig h2-eth0 mtu 1450')
    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()
```

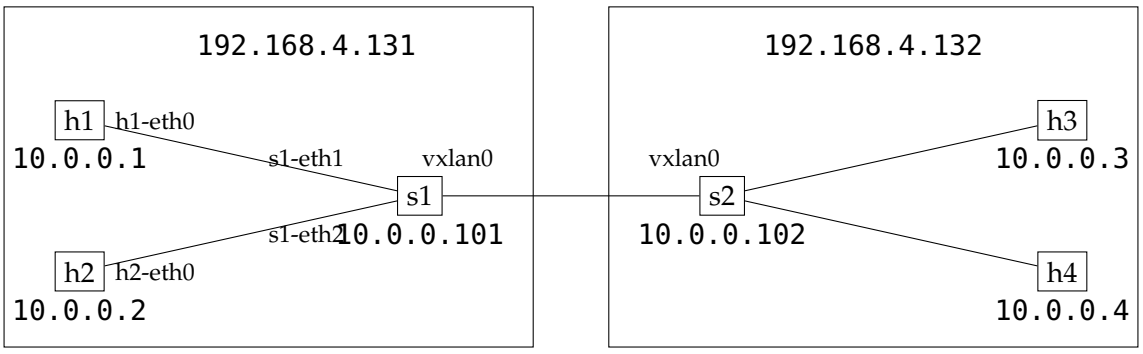

图 1: 拓扑结构

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。

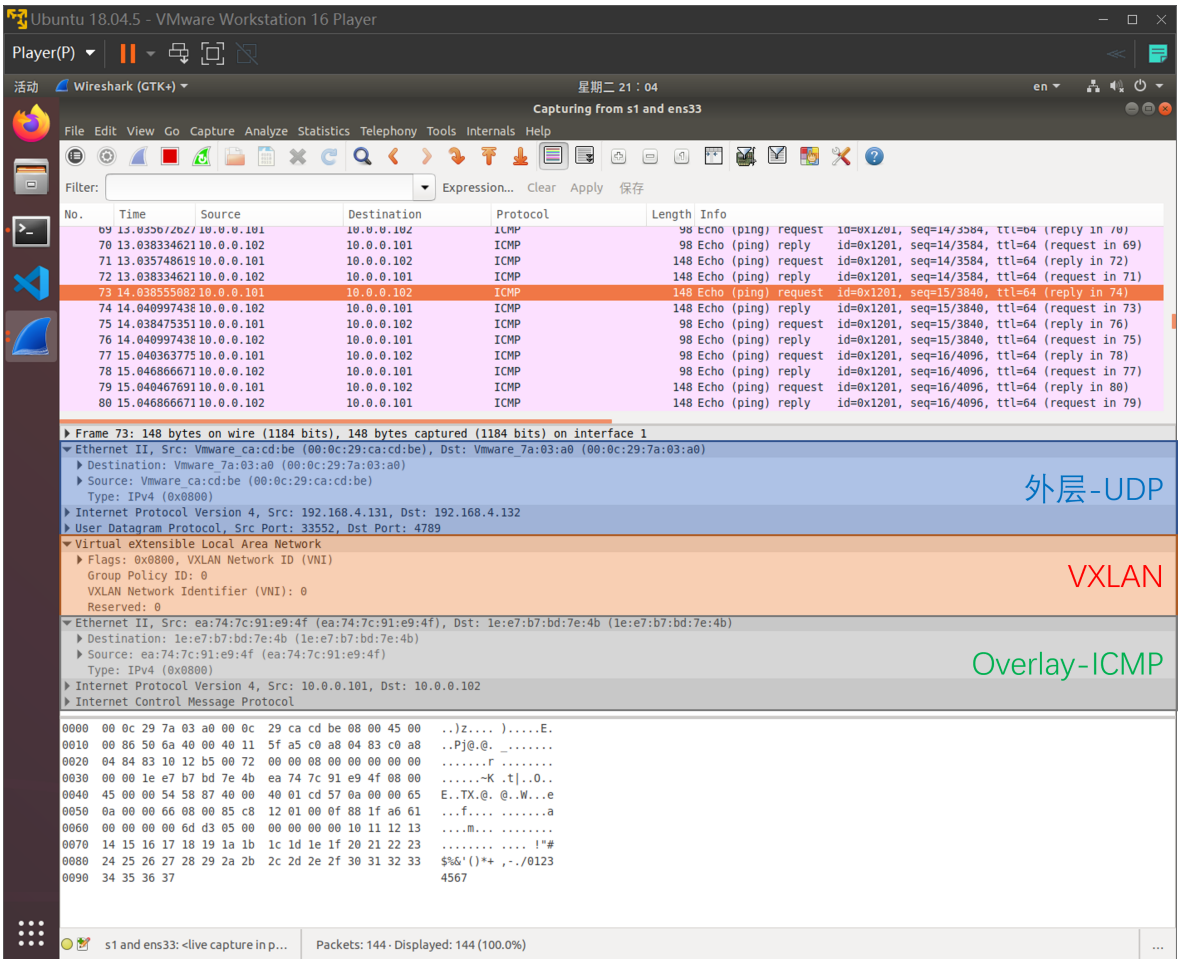

图 2: Wireshark 抓包

3 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.

4 PING 测试

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

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**。

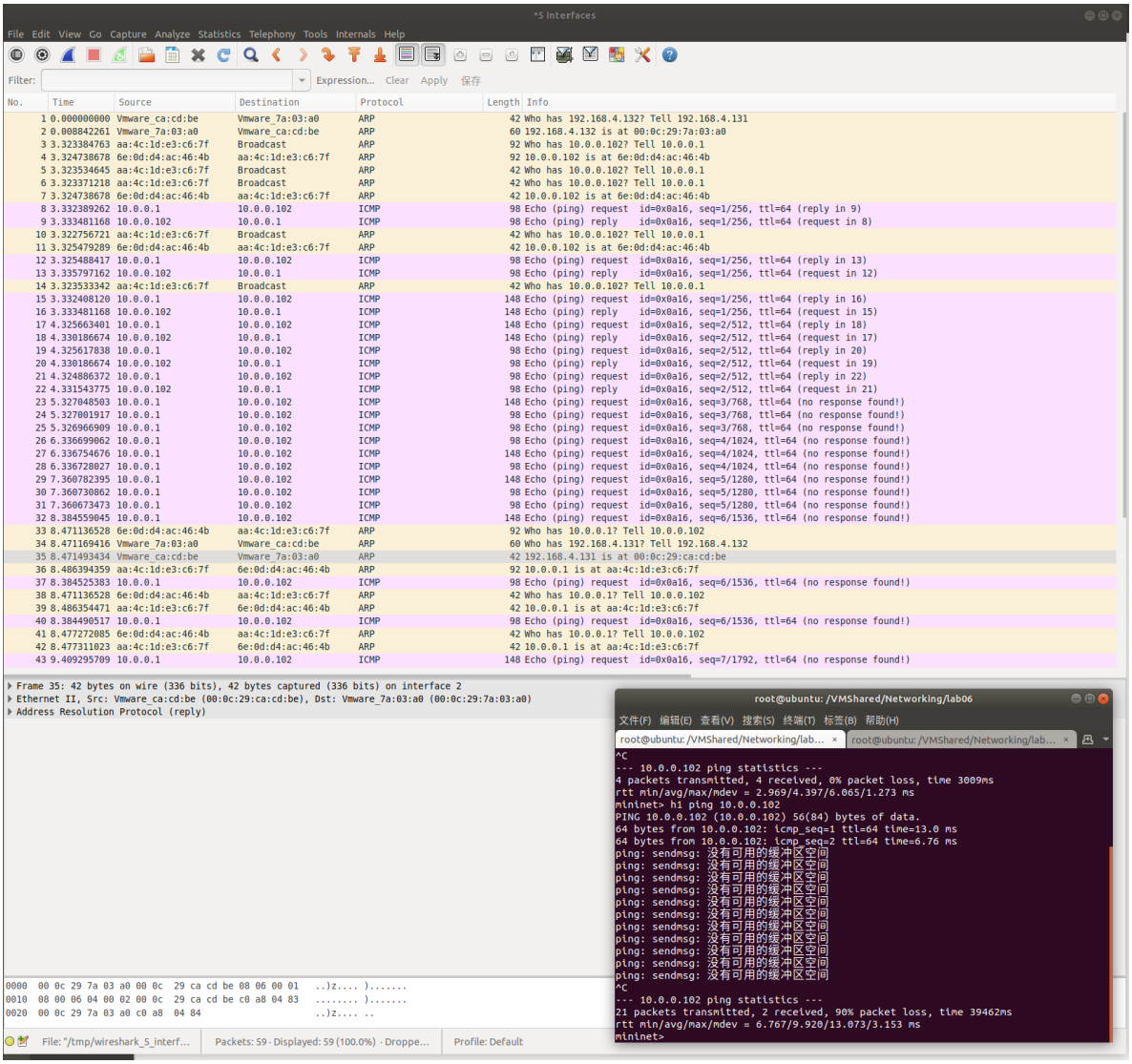

图 3: 无缓冲区空间抓包情况

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。**

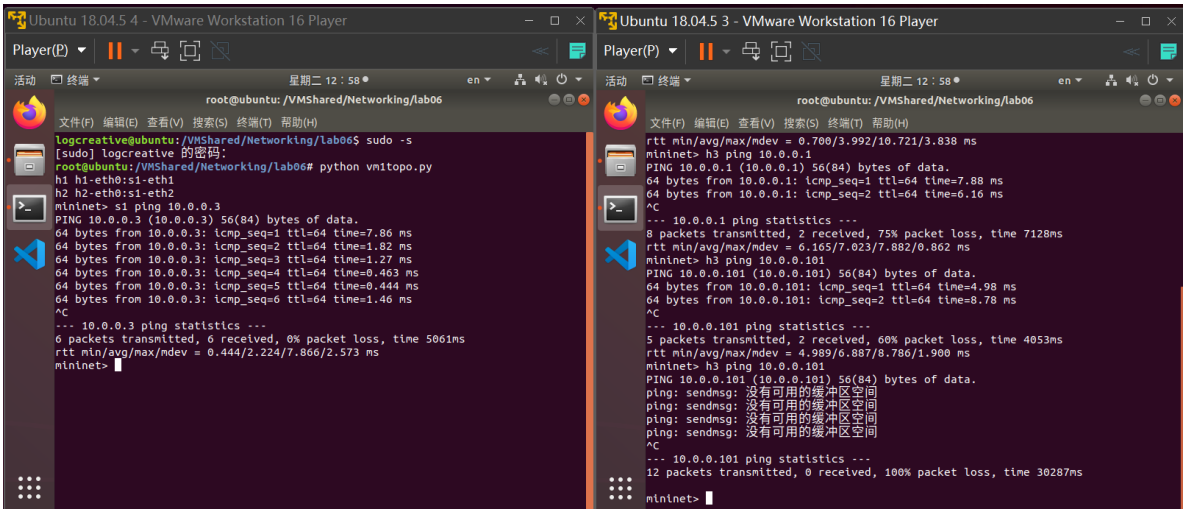

图 4: 同一路径不同发送主动方测试

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。以及可能会手写控制器[1]。

表 1: 运行环境

操作系统	Ubuntu 18.04.5
虚拟软件	VMWare Workstation
克隆设置	链接克隆与完整克隆皆已测试

参考文献

[1] XJTU_QYQ. 利用Mininet进行VxLAN验证实验[EB/OL]. 2018. https://blog.csdn.net/m0_37313888/article/details/82748230.