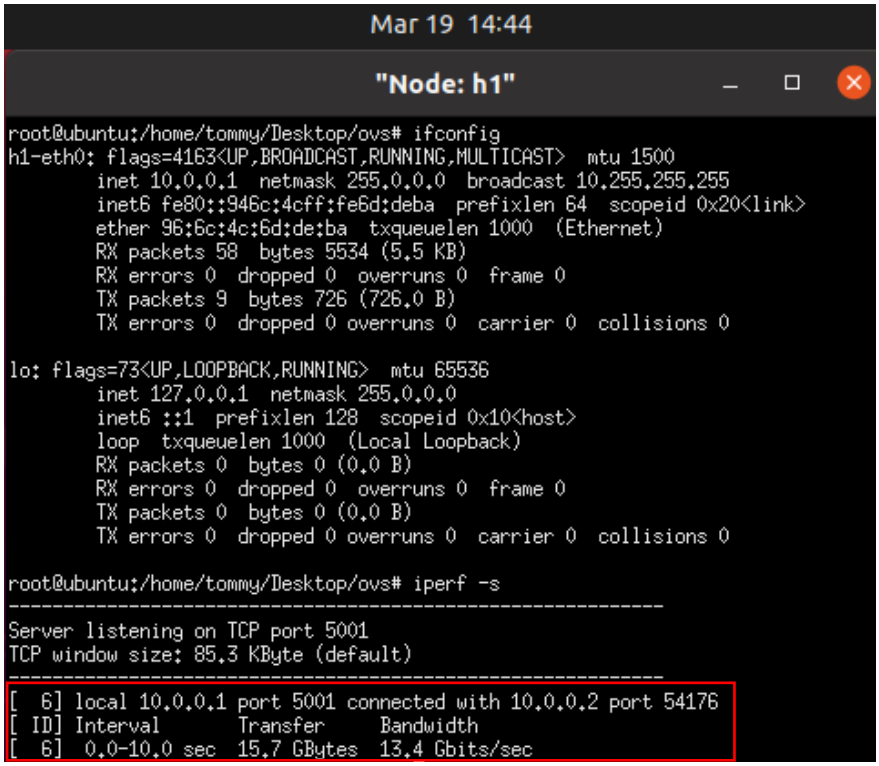
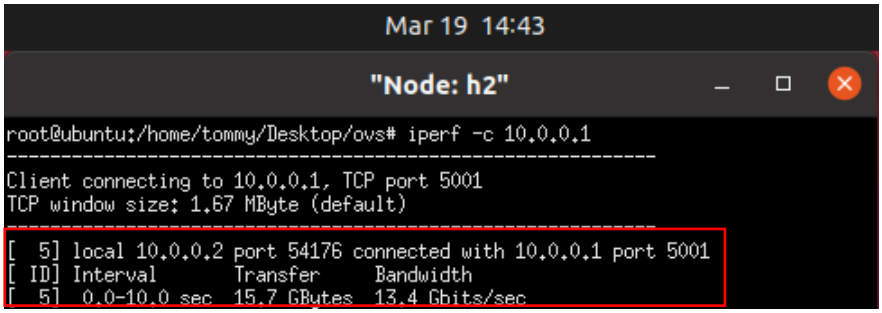
Report for Lab3: QoS Implementation with OvS

519021910913 黄喆敏

Task1 请在你自己的环境中完成上面的连通性测试，并以截图的形式分别记录 Node:h1 和 Node:h2 中 iperf 的输出结果。

答：截图如下所示。可以看到 Node:h1 与 Node:h2 已正常联通，带宽等指标正常。

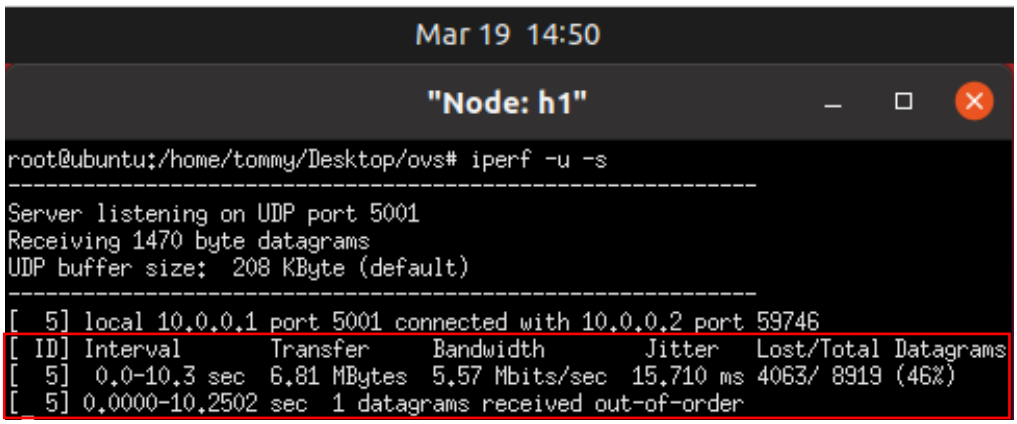


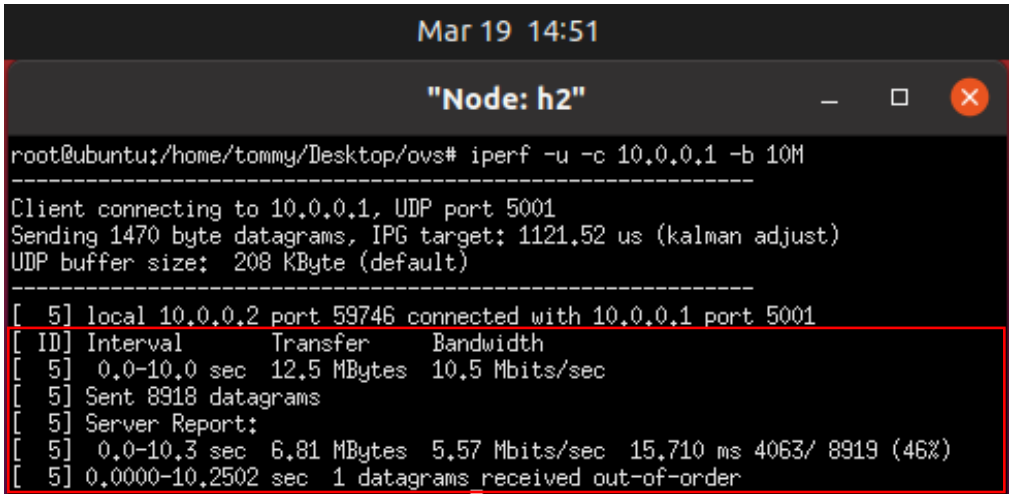


Task2.1 请截图记录输出结果，截图要求同 Task1，并着重关注其中的带宽、抖动、丢包率等数据。

答：截图如下所示。可以看到，发送端 h2 的发送带宽约为 10Mbps，但由于我们对接收端 h1 的收包速率进行了限速，当收包速率超过 5Mbps 时，将多余的包直接丢掉。

因此接收带宽为 5.57Mbps，约为 5Mbps；抖动为 15.710ms；丢包率为 46%，即丢掉了约一半的包，符合预期。



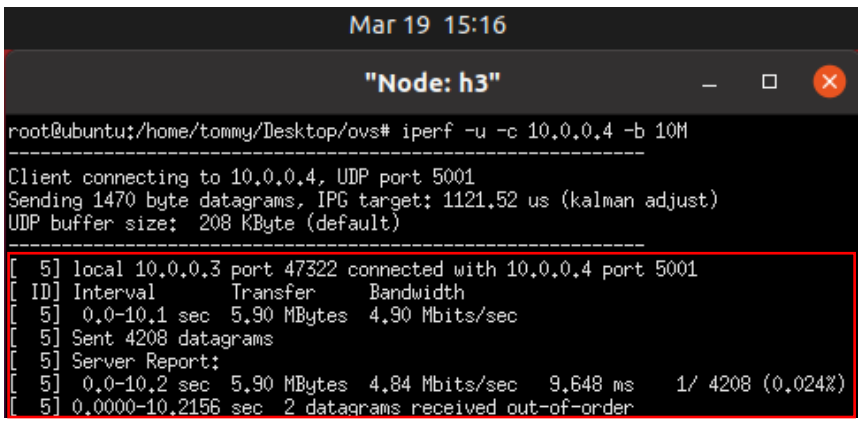


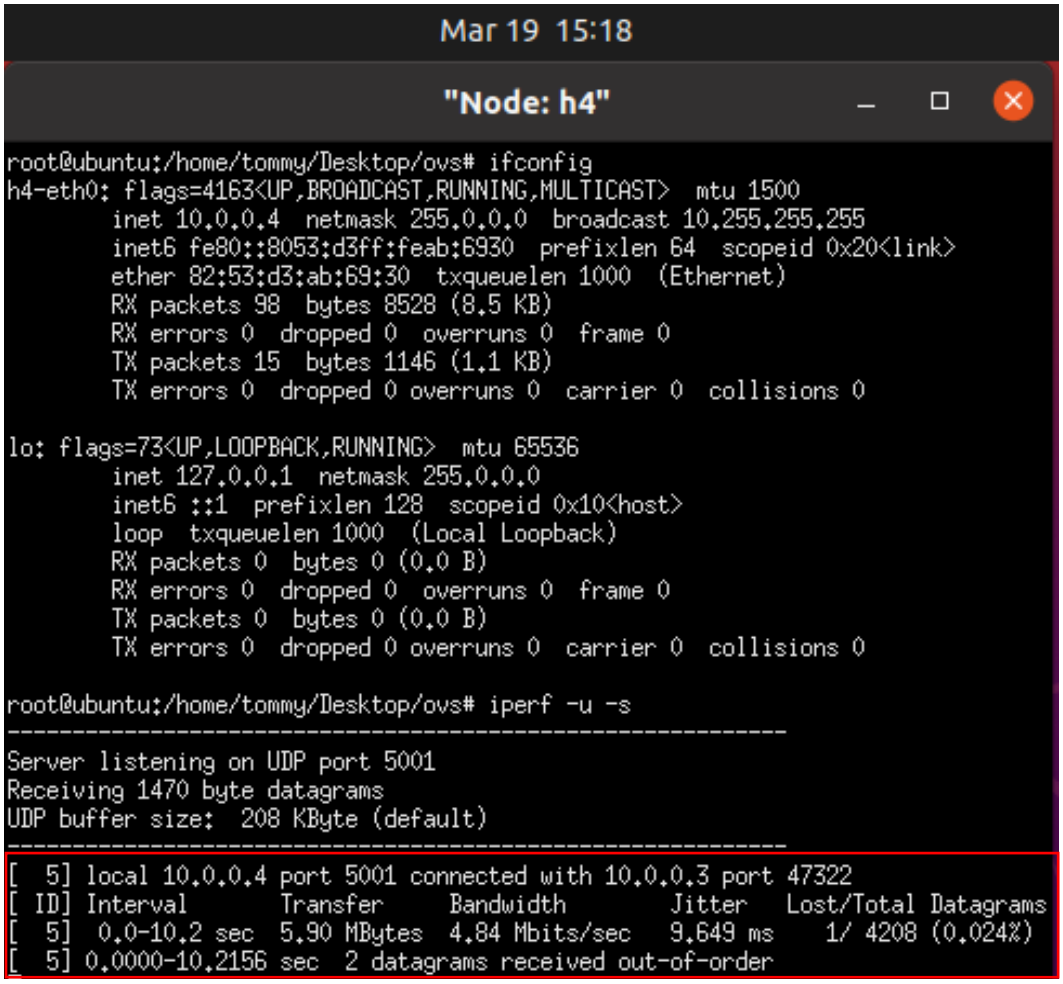
Task2.2 同上，此处也需要截图记录实验结果。（队列限速）

答： h3、h4 的截图分别如下所示。可以看到，由于指定了队列最大速率为 5Mbps，因此接收端带宽为 4.84Mbps，约为 5Mbps。

抖动为 9.649ms，小于网卡限速的抖动。

由于**队列限速会将数据包缓存起来，而不会像网卡限速那样简单的把数据包丢弃**，因此丢包率远小于网卡限速，实验中丢包率为 0.024%。





Question 1 尝试理解 Line19,20 两条指令，指出每条指令的具体工作是什么，并逐个分析其中各个参数的具体含义。

答：

Line19：为 s1 加入新的流内容，对于来自端口 5 的包，使用 Meter 表限速，丢弃超过 5M 的数据包，然后从端口 6 转发出去，协议为 openflow13。（修改流表以使用 Meter 表）

Line20：输出 s1 上的所有流内容，包括隐藏的流，协议为 openflow13。

参数含义：

s1：指交换机

in_port：数据包输入端口号

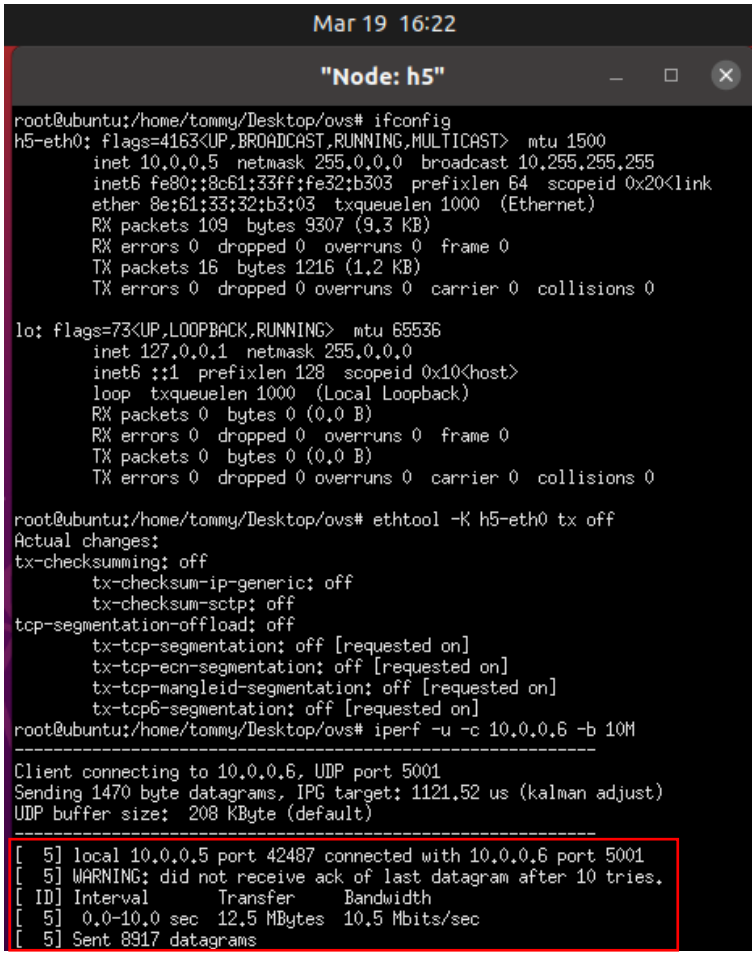
action：转发动作，meter:1 指 1 号 Meter 表

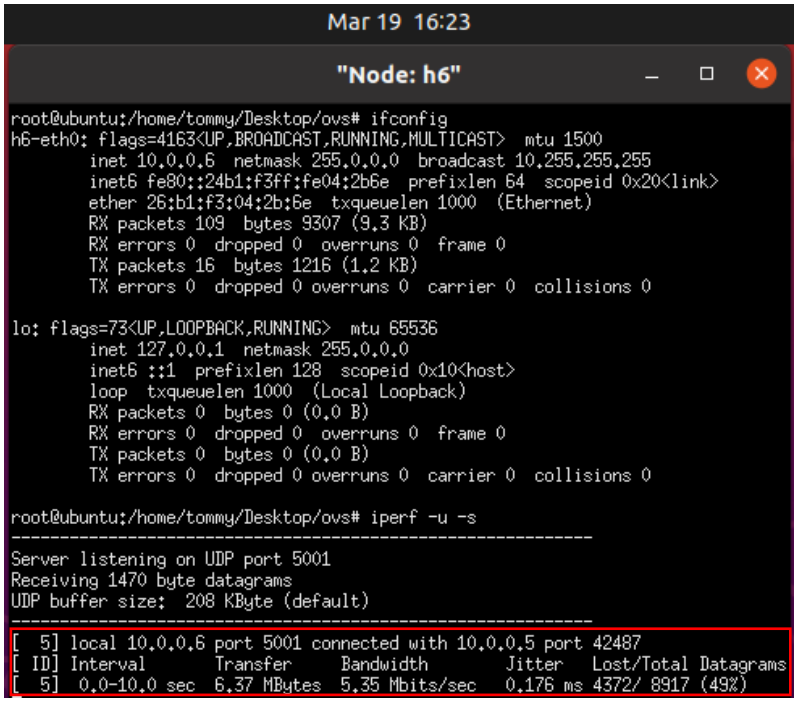
output：数据包输出端口号

-O：后面跟使用的协议

Task2.3 同上，请将此处的实验结果按要求截图。（Meter 表）

答：h5 与 h6 的截图如下所示。可以看到使用 Meter 表成功进行了限速，带宽为 5.35Mbps。抖动率较低，为 0.176ms。丢包率则为 49%左右。





Question 2 到这里，你已经完成了三种限速方式的实验，并获得了三组测试数据，请你就三组数据中的带宽、抖动和丢包率等参数，对三种限速方式进行横向比较，并适当地分析原因。

答：比较如下表所示，我们同时计算出带宽与限制速度（5M）的误差率：

	带宽	抖动	丢包率	带宽误差率
网卡限速	5.57Mbps	15.710ms	45.554%	11.4%
队列限速	4.84Mbps	9.649ms	0.024%	-3.2%
Meter 表限速	5.35Mbps	0.176ms	49.030%	7.0%

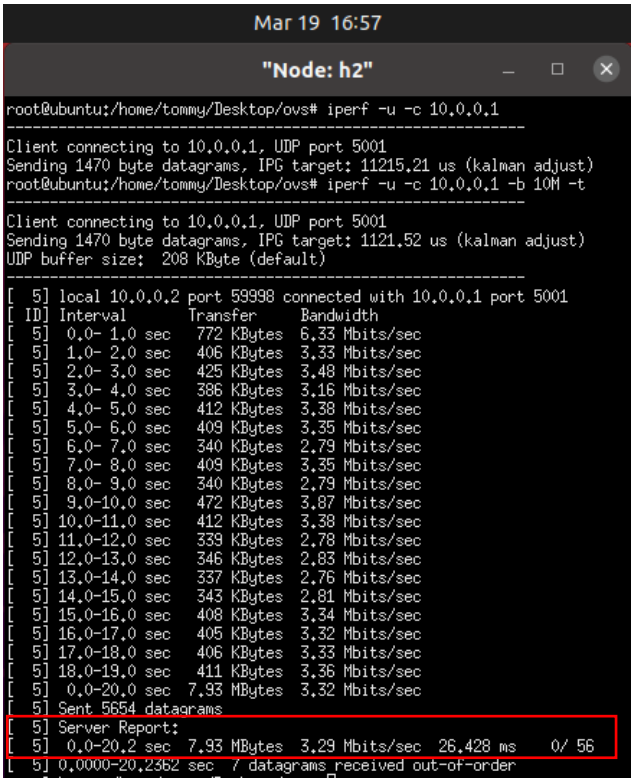
对于网卡限速，其表现较差，其实现最简单，控制粒度较粗。

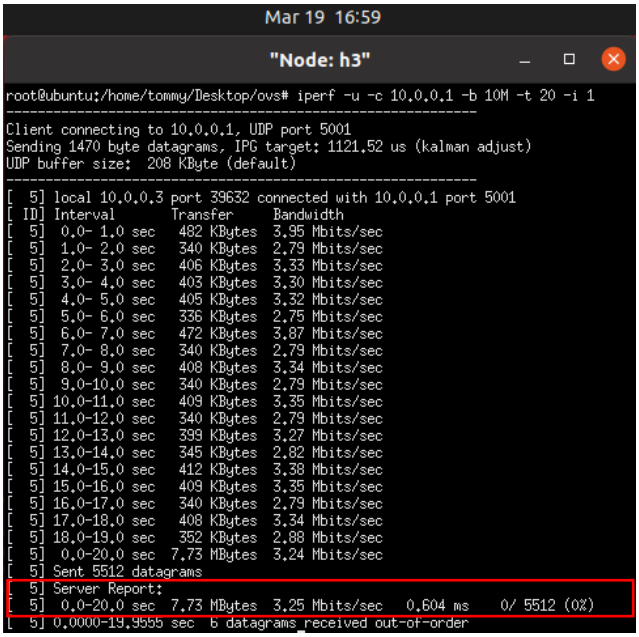
对于队列限速，其丢包率远小于另两种方法，因为当速率超过配置速率时，队列限速会将数据包缓存，而不会简单地丢弃；并且带宽误差率最小，表现最好。

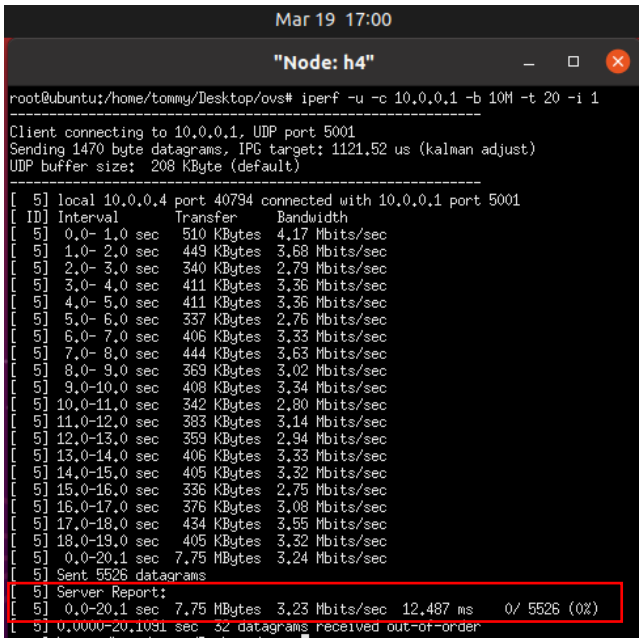
对于 Meter 表限速，其抖动远小于另两种方法，但是丢包率最高；而带宽误差率介于网卡限速与 Meter 表限速两者之间。

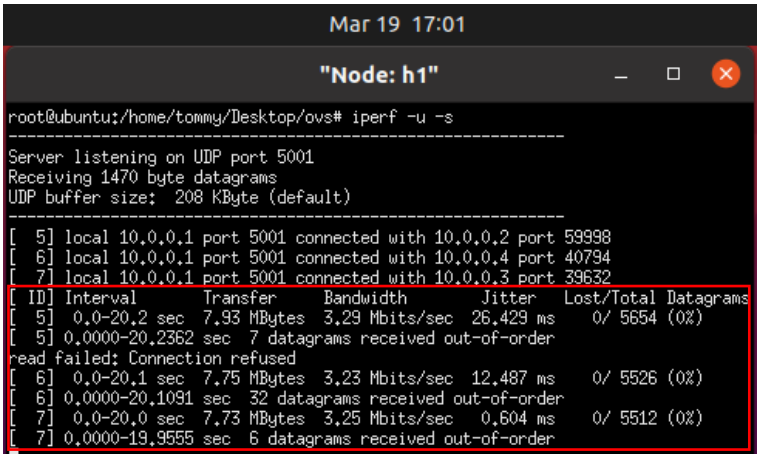
Task3 在限制 Server 端（h1）的带宽为 10Mb 的前提下，观察稳定后的三个 Client 的带宽，将结果截图并简单分析。

答：我们首先将之前实验的配置清除，再运行命令。我们采用**队列限速**的方法限制 h1 的带宽。结果截图如下：




Mar 19 16:59
"Node: h3"
root@ubuntu:/home/tommy/Desktop/ovs# iperf -u -c 10.0.0.1 -b 10M -t 20 -i 1
Client connecting to 10.0.0.1, UDP port 5001
Sending 1470 byte datagrams, IPG target: 1121.52 us (kalman adjust)
UDP buffer size: 208 KByte (default)
[5] local 10.0.0.3 port 39632 connected with 10.0.0.1 port 5001
[ID] Interval Transfer Bandwidth
[5] 0.0- 1.0 sec 482 KBytes 3.95 Mbits/sec
[5] 1.0- 2.0 sec 340 KBytes 2.79 Mbits/sec
[5] 2.0- 3.0 sec 406 KBytes 3.33 Mbits/sec
[5] 3.0- 4.0 sec 403 KBytes 3.30 Mbits/sec
[5] 4.0- 5.0 sec 405 KBytes 3.32 Mbits/sec
[5] 5.0- 6.0 sec 336 KBytes 2.75 Mbits/sec
[5] 6.0- 7.0 sec 472 KBytes 3.87 Mbits/sec
[5] 7.0- 8.0 sec 340 KBytes 2.79 Mbits/sec
[5] 8.0- 9.0 sec 408 KBytes 3.34 Mbits/sec
[5] 9.0-10.0 sec 340 KBytes 2.79 Mbits/sec
[5] 10.0-11.0 sec 409 KBytes 3.35 Mbits/sec
[5] 11.0-12.0 sec 340 KBytes 2.79 Mbits/sec
[5] 12.0-13.0 sec 399 KBytes 3.27 Mbits/sec
[5] 13.0-14.0 sec 345 KBytes 2.82 Mbits/sec
[5] 14.0-15.0 sec 412 KBytes 3.38 Mbits/sec
[5] 15.0-16.0 sec 409 KBytes 3.35 Mbits/sec
[5] 16.0-17.0 sec 340 KBytes 2.79 Mbits/sec
[5] 17.0-18.0 sec 408 KBytes 3.34 Mbits/sec
[5] 18.0-19.0 sec 352 KBytes 2.88 Mbits/sec
[5] 0.0-20.0 sec 7.73 MBytes 3.24 Mbits/sec
[5] Sent 5512 datagrams
[5] Server Report:
[5] 0.0-20.0 sec 7.73 MBytes 3.25 Mbits/sec 0.604 ms 0/ 5512 (0%)
[5] 0.0000-19.9555 sec 6 datagrams received out-of-order


Mar 19 17:00
"Node: h4"
root@ubuntu:/home/tommy/Desktop/ovs# iperf -u -c 10.0.0.1 -b 10M -t 20 -i 1
Client connecting to 10.0.0.1, UDP port 5001
Sending 1470 byte datagrams, IPG target: 1121.52 us (kalman adjust)
UDP buffer size: 208 KByte (default)
[5] local 10.0.0.4 port 40794 connected with 10.0.0.1 port 5001
[ID] Interval Transfer Bandwidth
[5] 0.0- 1.0 sec 510 KBytes 4.17 Mbits/sec
[5] 1.0- 2.0 sec 449 KBytes 3.68 Mbits/sec
[5] 2.0- 3.0 sec 340 KBytes 2.79 Mbits/sec
[5] 3.0- 4.0 sec 411 KBytes 3.36 Mbits/sec
[5] 4.0- 5.0 sec 411 KBytes 3.36 Mbits/sec
[5] 5.0- 6.0 sec 337 KBytes 2.76 Mbits/sec
[5] 6.0- 7.0 sec 406 KBytes 3.33 Mbits/sec
[5] 7.0- 8.0 sec 444 KBytes 3.63 Mbits/sec
[5] 8.0- 9.0 sec 369 KBytes 3.02 Mbits/sec
[5] 9.0-10.0 sec 408 KBytes 3.34 Mbits/sec
[5] 10.0-11.0 sec 342 KBytes 2.80 Mbits/sec
[5] 11.0-12.0 sec 383 KBytes 3.14 Mbits/sec
[5] 12.0-13.0 sec 359 KBytes 2.94 Mbits/sec
[5] 13.0-14.0 sec 406 KBytes 3.33 Mbits/sec
[5] 14.0-15.0 sec 405 KBytes 3.32 Mbits/sec
[5] 15.0-16.0 sec 336 KBytes 2.75 Mbits/sec
[5] 16.0-17.0 sec 376 KBytes 3.08 Mbits/sec
[5] 17.0-18.0 sec 434 KBytes 3.55 Mbits/sec
[5] 18.0-19.0 sec 405 KBytes 3.32 Mbits/sec
[5] 0.0-20.1 sec 7.75 MBytes 3.24 Mbits/sec
[5] Sent 5526 datagrams
[5] Server Report:
[5] 0.0-20.1 sec 7.75 MBytes 3.23 Mbits/sec 12.487 ms 0/ 5526 (0%)
[5] 0.0000-20.1091 sec 32 datagrams received out-of-order


Mar 19 17:01
"Node: h1"
root@ubuntu:/home/tommy/Desktop/ovs# iperf -u -s
Server listening on UDP port 5001
Receiving 1470 byte datagrams
UDP buffer size: 208 KByte (default)
[5] local 10.0.0.1 port 5001 connected with 10.0.0.2 port 59998
[6] local 10.0.0.1 port 5001 connected with 10.0.0.4 port 40794
[7] local 10.0.0.1 port 5001 connected with 10.0.0.3 port 39632
[ID] Interval Transfer Bandwidth Jitter Lost/Total Datagrams
[5] 0.0-20.2 sec 7.93 MBytes 3.29 Mbits/sec 26.429 ms 0/ 5654 (0%)
[5] 0.0000-20.2362 sec 7 datagrams received out-of-order
read failed: Connection refused
[6] 0.0-20.1 sec 7.75 MBytes 3.23 Mbits/sec 12.487 ms 0/ 5526 (0%)
[6] 0.0000-20.1091 sec 32 datagrams received out-of-order
[7] 0.0-20.0 sec 7.73 MBytes 3.25 Mbits/sec 0.604 ms 0/ 5512 (0%)
[7] 0.0000-19.9555 sec 6 datagrams received out-of-order

可以看到，稳定后三个 Client 的带宽分别为 3.29Mb、3.23Mb 和 3.25Mb 左右，三者相差不大。但三个 Client 的总带宽为 9.77Mb，没有达到限速带宽，有一定的损耗。

Task4 你可以通过上述三种限速的方法来达成目标，请记录你的设计过程（思路及运行指令），并将你稳定后的三个 Client 的带宽结果截图。

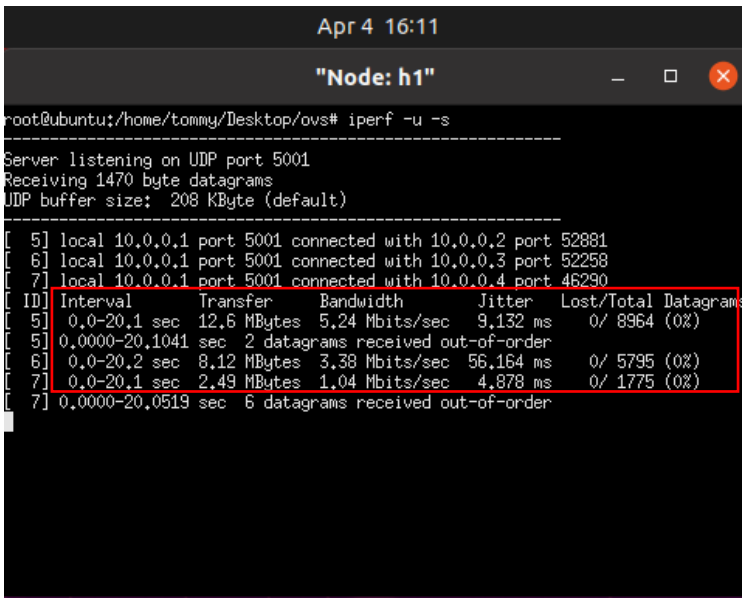
答：通过查询文档，我们发现**队列限速中可以设置最小的速率**。因此为了保证 h2 和 h3 的带宽，我们首先设置两条队列，最小速率分别为 5Mb 和 3Mb，接着下发流表，设置对应的队列。

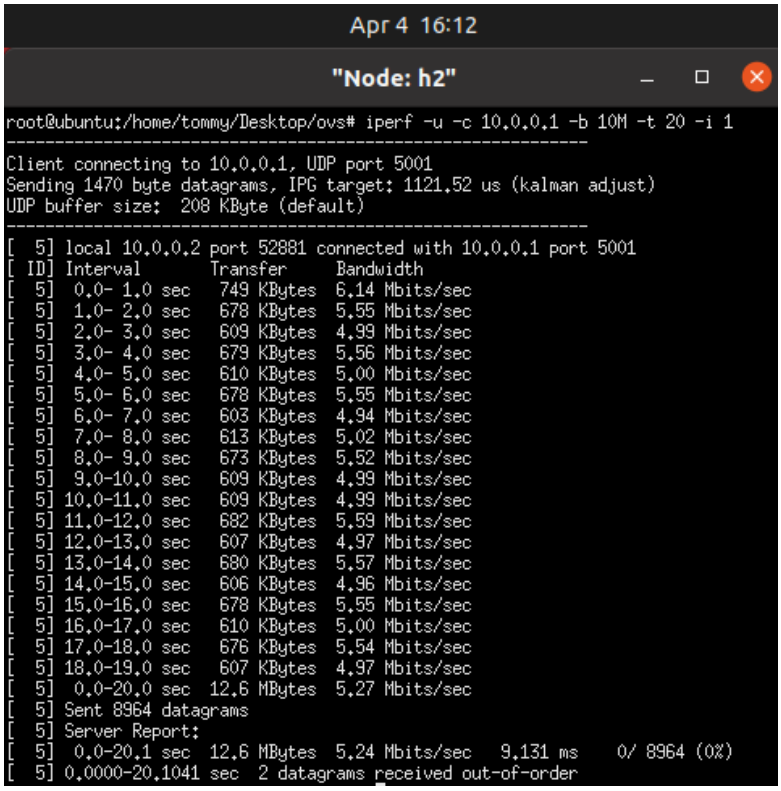
在进行第一步的实验，即只设置了 min-rate，没有设置 max-rate 后，我们发现 **h4 的带宽过小，经常在几十 k 到几百 k 之间浮动**。因此为了在保证 h2 和 h3 的前提下，使 h4 的带宽尽量多，我们设置了 **h2 的带宽上限为 5.5Mb，h3 的带宽上限为 3.5Mb**。最终 h4 的带宽可以达到 1Mb 左右。

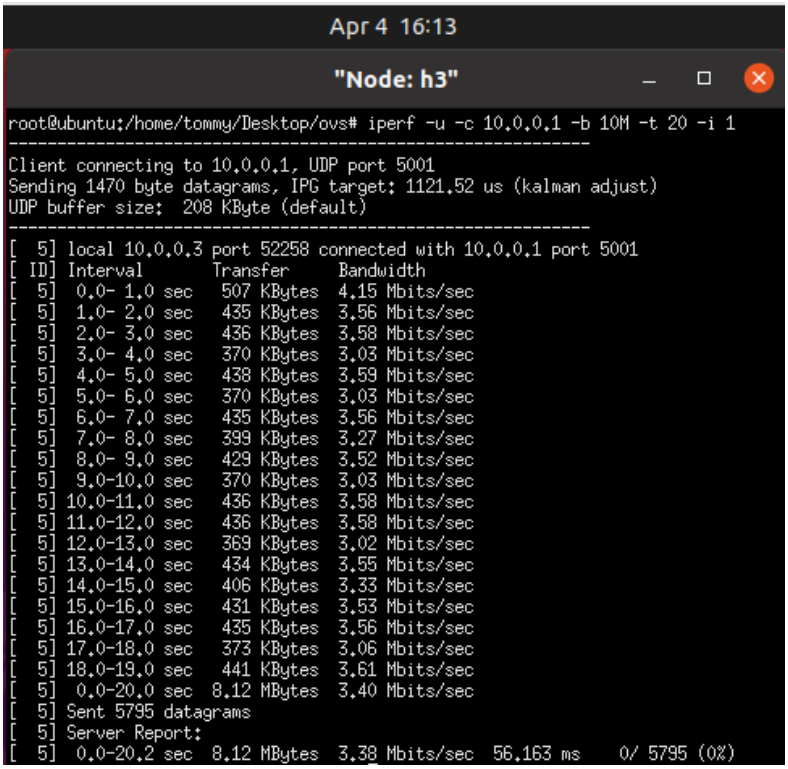
我们采取的命令如下：

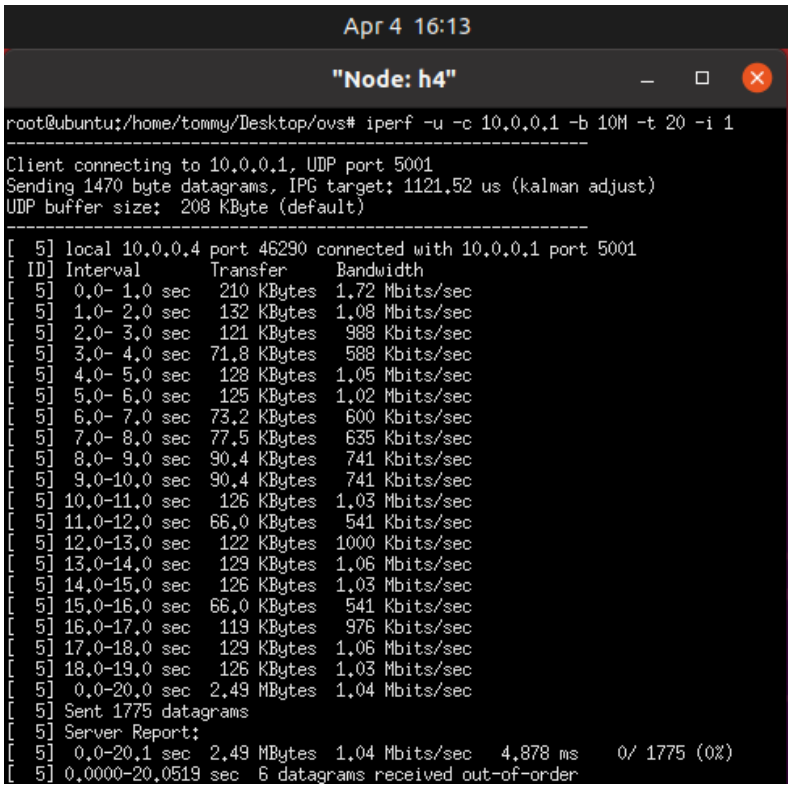
```
$ ovs-vsctl set port s1-eth1 qos=@newqos -- \
--id=@newqos create qos type=linux-htb
other-config:max-rate=10000000 queues=1=@q1,2=@q2 -- \
--id=@q1 create queue other-config:min-rate=5000000
other-config:max-rate=5500000 -- \
--id=@q2 create queue other-config:min-rate=3000000
other-config:max-rate=3500000

$ ovs-ofctl add-flow s1 in_port=2,actions=set_queue:2,output:1 -O openflow13
$ ovs-ofctl add-flow s1 in_port=3,actions=set_queue:3,output:1 -O openflow13
```

最终结果如下图所示，h2 带宽为 5.24Mb，h3 带宽为 3.38Mb，h4 带宽为 1.04Mb，符合要求。






Apr 4 16:13
"Node: h3"
root@ubuntu:/home/tommy/Desktop/ovs# iperf -u -c 10.0.0.1 -b 10M -t 20 -i 1
Client connecting to 10.0.0.1, UDP port 5001
Sending 1470 byte datagrams, IPG target: 1121.52 us (kalman adjust)
UDP buffer size: 208 KByte (default)
[5] local 10.0.0.3 port 52258 connected with 10.0.0.1 port 5001
[ID] Interval Transfer Bandwidth
[5] 0.0- 1.0 sec 507 KBytes 4.15 Mbits/sec
[5] 1.0- 2.0 sec 435 KBytes 3.56 Mbits/sec
[5] 2.0- 3.0 sec 436 KBytes 3.58 Mbits/sec
[5] 3.0- 4.0 sec 370 KBytes 3.03 Mbits/sec
[5] 4.0- 5.0 sec 438 KBytes 3.59 Mbits/sec
[5] 5.0- 6.0 sec 370 KBytes 3.03 Mbits/sec
[5] 6.0- 7.0 sec 435 KBytes 3.56 Mbits/sec
[5] 7.0- 8.0 sec 399 KBytes 3.27 Mbits/sec
[5] 8.0- 9.0 sec 429 KBytes 3.52 Mbits/sec
[5] 9.0-10.0 sec 370 KBytes 3.03 Mbits/sec
[5] 10.0-11.0 sec 436 KBytes 3.58 Mbits/sec
[5] 11.0-12.0 sec 436 KBytes 3.58 Mbits/sec
[5] 12.0-13.0 sec 369 KBytes 3.02 Mbits/sec
[5] 13.0-14.0 sec 434 KBytes 3.55 Mbits/sec
[5] 14.0-15.0 sec 406 KBytes 3.33 Mbits/sec
[5] 15.0-16.0 sec 431 KBytes 3.53 Mbits/sec
[5] 16.0-17.0 sec 435 KBytes 3.56 Mbits/sec
[5] 17.0-18.0 sec 373 KBytes 3.06 Mbits/sec
[5] 18.0-19.0 sec 441 KBytes 3.61 Mbits/sec
[5] 0.0-20.0 sec 8.12 MBytes 3.40 Mbits/sec
[5] Sent 5795 datagrams
[5] Server Report:
[5] 0.0-20.2 sec 8.12 MBytes 3.38 Mbits/sec 56.163 ms 0/ 5795 (0%)


Apr 4 16:13
"Node: h4"
root@ubuntu:/home/tommy/Desktop/ovs# iperf -u -c 10.0.0.1 -b 10M -t 20 -i 1
Client connecting to 10.0.0.1, UDP port 5001
Sending 1470 byte datagrams, IPG target: 1121.52 us (kalman adjust)
UDP buffer size: 208 KByte (default)
[5] local 10.0.0.4 port 46290 connected with 10.0.0.1 port 5001
[ID] Interval Transfer Bandwidth
[5] 0.0- 1.0 sec 210 KBytes 1.72 Mbits/sec
[5] 1.0- 2.0 sec 132 KBytes 1.08 Mbits/sec
[5] 2.0- 3.0 sec 121 KBytes 988 Kbits/sec
[5] 3.0- 4.0 sec 71.8 KBytes 588 Kbits/sec
[5] 4.0- 5.0 sec 128 KBytes 1.05 Mbits/sec
[5] 5.0- 6.0 sec 125 KBytes 1.02 Mbits/sec
[5] 6.0- 7.0 sec 73.2 KBytes 600 Kbits/sec
[5] 7.0- 8.0 sec 77.5 KBytes 635 Kbits/sec
[5] 8.0- 9.0 sec 90.4 KBytes 741 Kbits/sec
[5] 9.0-10.0 sec 90.4 KBytes 741 Kbits/sec
[5] 10.0-11.0 sec 126 KBytes 1.03 Mbits/sec
[5] 11.0-12.0 sec 66.0 KBytes 541 Kbits/sec
[5] 12.0-13.0 sec 122 KBytes 1000 Kbits/sec
[5] 13.0-14.0 sec 129 KBytes 1.06 Mbits/sec
[5] 14.0-15.0 sec 126 KBytes 1.03 Mbits/sec
[5] 15.0-16.0 sec 66.0 KBytes 541 Kbits/sec
[5] 16.0-17.0 sec 119 KBytes 976 Kbits/sec
[5] 17.0-18.0 sec 129 KBytes 1.06 Mbits/sec
[5] 18.0-19.0 sec 126 KBytes 1.03 Mbits/sec
[5] 0.0-20.0 sec 2.49 MBytes 1.04 Mbits/sec
[5] Sent 1775 datagrams
[5] Server Report:
[5] 0.0-20.1 sec 2.49 MBytes 1.04 Mbits/sec 4.878 ms 0/ 1775 (0%)
[5] 0.0000-20.0519 sec 6 datagrams received out-of-order

Reference

[1] https://docs.pica8.com/pages/viewpage.action?pageId=52207258

[2] https://docs.pica8.com/display/PicOS422sp/Configuring+Meter

[3] https://www.cnblogs.com/goldsunshine/p/11720310.html

[4] https://www.cnblogs.com/goldsunshine/p/13056429.html