111550139

郭芷安

SDN Lab4 Report

Link up

Result (using rate = 1M)

```
mininet> h1 iperf -c 10.6.1.2 -u -b 1M -r
------------------------------------------------------------
Server listening on UDP port 5001
UDP buffer size:  208 KByte (default)
------------------------------------------------------------
------------------------------------------------------------
Client connecting to 10.6.1.2, UDP port 5001
Sending 1470 byte datagrams, IPG target: 11215.21 us (kalman adjust)
UDP buffer size:  208 KByte (default)
------------------------------------------------------------
[  1] local 10.6.1.1 port 42850 connected with 10.6.1.2 port 5001
[ ID] Interval       Transfer     Bandwidth
[  1] 0.0000-10.0156 sec  1.25 MBytes  1.05 Mbits/sec
[  1] Sent 896 datagrams
[  1] Server Report:
[ ID] Interval           Transfer     Bandwidth        Jitter   Lost/Total  Latency avg/min/max/stdev PPS NetPwr
[  1] 0.0000-0.0000 sec  4.00 GBytes  -nan bits/sec   0.000 ms 4294966296/0 (0%) -/-/-/- ms 0 pps
[  2] local 10.6.1.1 port 5001 connected with 10.6.1.2 port 49414
[ ID] Interval       Transfer     Bandwidth        Jitter   Lost/Total Datagrams
[  2] 0.0000-10.0115 sec  1.25 MBytes  1.05 Mbits/sec   0.065 ms 0/894 (0%)
```

S1 interface

```
mininet> sh ovs-ofctl dump-ports -O OpenFlow14 s1
OFPST_PORT reply (OF1.4) (xid=0x2): 4 ports
  port LOCAL: rx pkts=0, bytes=0, drop=0, errs=0, frame=0, over=0, crc=0
           tx pkts=0, bytes=0, drop=0, errs=0, coll=0
           duration=1540.307s
  port  "s1-eth1": rx pkts=1823, bytes=1394700, drop=0, errs=0, frame=0, over=0, crc=0
           tx pkts=2724, bytes=1511351, drop=0, errs=0, coll=0
           duration=1540.341s
  port  "s1-eth2": rx pkts=933, bytes=128885, drop=0, errs=0, frame=0, over=0, crc=0
           tx pkts=2736, bytes=1522001, drop=0, errs=0, coll=0
           duration=1540.360s
  port  "s1-eth3": rx pkts=2721, bytes=1511071, drop=0, errs=0, frame=0, over=0, crc=0
           tx pkts=933, bytes=128885, drop=0, errs=0, coll=0
           duration=1540.331s
```

S4 interface

```
mininet> sh ovs-ofctl dump-ports -O OpenFlow14 s4
OFPST_PORT reply (OF1.4) (xid=0x2): 3 ports
  port LOCAL: rx pkts=0, bytes=0, drop=0, errs=0, frame=0, over=0, crc=0
           tx pkts=0, bytes=0, drop=0, errs=0, coll=0
           duration=1572.114s
  port  "s4-eth1": rx pkts=953, bytes=131665, drop=0, errs=0, frame=0, over=0, crc=0
           tx pkts=2741, bytes=1513851, drop=0, errs=0, coll=0
           duration=1572.303s
  port  "s4-eth2": rx pkts=2741, bytes=1513851, drop=0, errs=0, frame=0, over=0, crc=0
           tx pkts=951, bytes=131387, drop=0, errs=0, coll=0
           duration=1572.300s
```

1. Check if traffic from h1 to h2 through s2 and s3:
   a. Observe s1 interface, we can find that traffic from h1 to h2 will go through s2 and s3, because in s1 interface, we can see that the tx pkts in port 2 is much more than the rx pkt, which means that there is a user-generated

traffic go through s1-port2, in this case, the user-generated traffic is the udp traffic. Thus, this proves that the traffic from h1 to h2 will transport from s1-port2 to s2, s3, s5 and arrive at h2.

   b. Observe s4 interface, we can see that port2 is the port which receives user-generated traffic, and port1 is the port to send user-generated traffic. Thus, the traffic from h1 to h2 will not go through s4. If the traffic from h1 to h2 also go through s4, then the number of rx and tx packets of port1 and port2 will be approximately the same. (because in this case, the traffic from h2 to h1 go through s4).

2. Check the path from h2 to h1 through which switch
   a. Observe s1 interface, we can see that the rx packet is much more than the tx packet in port3, which means port3 is the port which receive udp traffic from h2.
   b. Observe s4 interface, we can see that port2 is the port which receives udp traffic, and port1 is the port to send udp traffic. Thus, the traffic from h2 to h1 will go through s4, and arrive at s1-port3, then received by h1.

Link down

Result (using rate = 10M)

```
mininet> h1 iperf -c 10.6.1.2 -u -b 10M -t 30 -r
------------------------------------------------------------
Server listening on UDP port 5001
UDP buffer size:  208 KByte (default)
------------------------------------------------------------
------------------------------------------------------------
Client connecting to 10.6.1.2, UDP port 5001
Sending 1470 byte datagrams, IPG target: 1121.52 us (kalman adjust)
UDP buffer size:  208 KByte (default)
------------------------------------------------------------
[  1] local 10.6.1.1 port 32964 connected with 10.6.1.2 port 5001
[ ID] Interval       Transfer     Bandwidth
[  1] 0.0000-30.0013 sec  37.5 MBytes  10.5 Mbits/sec
[  1] Sent 26753 datagrams
[  1] Server Report:
[ ID] Interval       Transfer     Bandwidth        Jitter   Lost/Total Datagrams
[  1] 0.0000-30.4153 sec   141 KBytes  37.9 Kbits/sec  2697.847 ms 26687/26791 (1e+02%)
[  1] 0.0000-30.4153 sec  6 datagrams received out-of-order
[  2] local 10.6.1.1 port 5001 connected with 10.6.1.2 port 43527
[ ID] Interval       Transfer     Bandwidth        Jitter   Lost/Total Datagrams
[  2] 0.0000-30.0013 sec  37.5 MBytes  10.5 Mbits/sec   0.047 ms 0/26752 (0%)
```

S1 interface

```
mininet> sh ovs-ofctl dump-ports -O OpenFlow14 s1
OFPST_PORT reply (OF1.4) (xid=0x2): 4 ports
  port LOCAL: rx pkts=0, bytes=0, drop=0, errs=0, frame=0, over=0, crc=0
           tx pkts=0, bytes=0, drop=0, errs=0, coll=0
           duration=659.796s
  port  "s1-eth1": rx pkts=55401, bytes=42807746, drop=0, errs=0, frame=0, over=0, crc=0
           tx pkts=55779, bytes=42818276, drop=0, errs=0, coll=0
           duration=660.163s
  port  "s1-eth2": rx pkts=23, bytes=3102, drop=0, errs=0, frame=0, over=0, crc=0
           tx pkts=21, bytes=2824, drop=0, errs=0, coll=0
           duration=660.164s
  port  "s1-eth3": rx pkts=55776, bytes=42818208, drop=0, errs=0, frame=0, over=0, crc=0
           tx pkts=55833, bytes=42868532, drop=0, errs=0, coll=0
           duration=660.160s
```

S4 interface

```
mininet> sh ovs-ofctl dump-ports -O OpenFlow14 s4
OFPST_PORT reply (OF1.4) (xid=0x2): 3 ports
  port LOCAL: rx pkts=0, bytes=0, drop=0, errs=0, frame=0, over=0, crc=0
           tx pkts=0, bytes=0, drop=0, errs=0, coll=0
           duration=700.720s
  port  "s4-eth1": rx pkts=55859, bytes=42872146, drop=0, errs=0, frame=0, over=0, crc=0
           tx pkts=55802, bytes=42821822, drop=0, errs=0, coll=0
           duration=700.847s
  port  "s4-eth2": rx pkts=55801, bytes=42821715, drop=0, errs=0, frame=0, over=0, crc=0
           tx pkts=28747, bytes=2911052, drop=0, errs=0, coll=0
           duration=700.846s
```

1. Check if both traffic go through s4
   a. Observe s1 interface

      We can see that both rx and tx packet in s1-port2 is much less than s1-port3. Thus, we can know that port3 is the port to receive and send udp packet, so both traffic go through s1-port3 and s4.

b. Observe s4 interface

   Observe s4 interface, the number of rx and tx packets of port1 and port2 is approximately the same. And we can also compare it with the number of rx packet s1-port2 and s1-port3 to know that most of packets are udp packet instead of default or background traffic.

2. Check if the iperf traffic rate is limited

   Observe s4 interface, we can see that the rx packet of port1 is 55859, and the tx packet of port2 is 28747, which indicates that there are a great number of packets from s1->s4->s5 will be dropped when they arrive s4. Because the band type of meter table is DROP, which will drop packets if the traffic rate is out of limit, so we can know that the traffic rate is limited.