实验名称：Lab 5：**网络层数据平面观察实验**

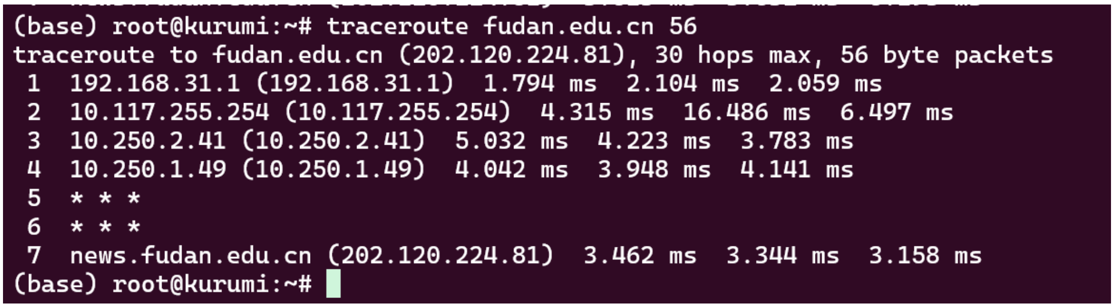
实验人：谢志康

学号：22307110187

时间：24.11.17—

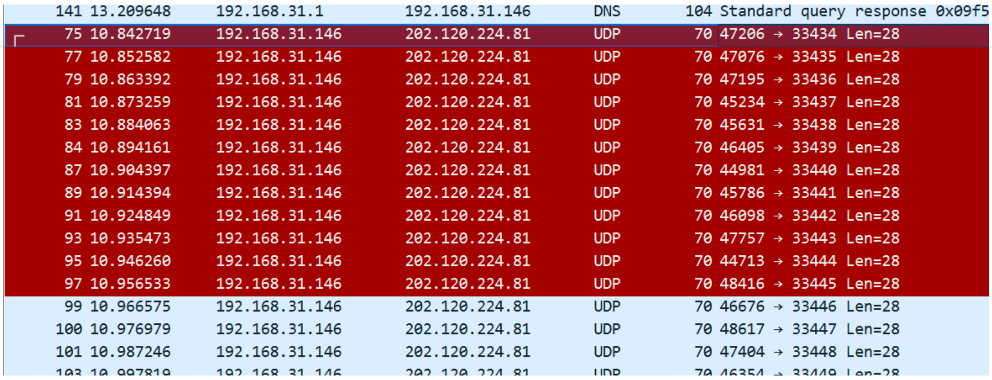
Part1：

用 traceroute 发送大小为 56 bytes 的 UDP 包，比如 traceroute fudan.edu.cn 56 ；并使用 wireshark 抓包。



```
(base) root@kurumi:~# traceroute fudan.edu.cn 56
traceroute to fudan.edu.cn (202.120.224.81), 30 hops max, 56 byte packets
 1  192.168.31.1 (192.168.31.1)  1.794 ms  2.104 ms  2.059 ms
 2  10.117.255.254 (10.117.255.254)  4.315 ms  16.486 ms  6.497 ms
 3  10.250.2.41 (10.250.2.41)  5.032 ms  4.223 ms  3.783 ms
 4  10.250.1.49 (10.250.1.49)  4.042 ms  3.948 ms  4.141 ms
 5  * * *
 6  * * *
 7  news.fudan.edu.cn (202.120.224.81)  3.462 ms  3.344 ms  3.158 ms
(base) root@kurumi:~# 
```

任务 1：选择第一个发送的 UDP 包，观察：

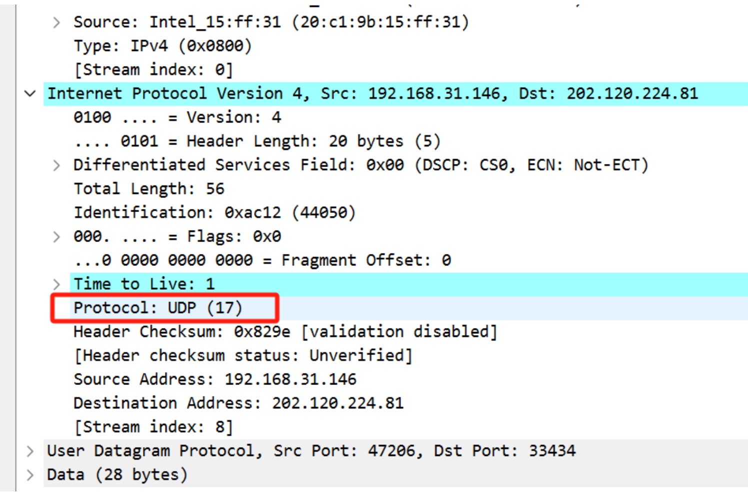


```
  141 13.209648   192.168.31.1     192.168.31.146   DNS   104 Standard query response 0x09f5
   75 10.842719   192.168.31.146   202.120.224.81   UDP    70 47206 → 33434 Len=28
   77 10.852582   192.168.31.146   202.120.224.81   UDP    70 47076 → 33435 Len=28
   79 10.863392   192.168.31.146   202.120.224.81   UDP    70 47195 → 33436 Len=28
   81 10.873259   192.168.31.146   202.120.224.81   UDP    70 45234 → 33437 Len=28
   83 10.884063   192.168.31.146   202.120.224.81   UDP    70 45631 → 33438 Len=28
   84 10.894161   192.168.31.146   202.120.224.81   UDP    70 46405 → 33439 Len=28
   87 10.904397   192.168.31.146   202.120.224.81   UDP    70 44981 → 33440 Len=28
   89 10.914394   192.168.31.146   202.120.224.81   UDP    70 45786 → 33441 Len=28
   91 10.924849   192.168.31.146   202.120.224.81   UDP    70 46098 → 33442 Len=28
   93 10.935473   192.168.31.146   202.120.224.81   UDP    70 47757 → 33443 Len=28
   95 10.946260   192.168.31.146   202.120.224.81   UDP    70 44713 → 33444 Len=28
   97 10.956533   192.168.31.146   202.120.224.81   UDP    70 48416 → 33445 Len=28
   99 10.966575   192.168.31.146   202.120.224.81   UDP    70 46676 → 33446 Len=28
  100 10.976979   192.168.31.146   202.120.224.81   UDP    70 48617 → 33447 Len=28
  101 10.987246   192.168.31.146   202.120.224.81   UDP    70 47404 → 33448 Len=28
  103 10.997819   192.168.31.146   202.120.224.81   UDP    70 46354 → 33449 Len=28
```

1. 发送端的 IP 地址?

192.168.31.146

2. 在 IP header 中，上层协议的数值是多少?



```
    > Source: Intel_15:ff:31 (20:c1:9b:15:ff:31)
      Type: IPv4 (0x0800)
      [Stream index: 0]
  v Internet Protocol Version 4, Src: 192.168.31.146, Dst: 202.120.224.81
      0100 .... = Version: 4
      .... 0101 = Header Length: 20 bytes (5)
    > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
      Total Length: 56
      Identification: 0xac12 (44050)
    > 000. .... = Flags: 0x0
      ...0 0000 0000 0000 = Fragment Offset: 0
    > Time to Live: 1
      Protocol: UDP (17)
      Header Checksum: 0x829e [validation disabled]
      [Header checksum status: Unverified]
      Source Address: 192.168.31.146
      Destination Address: 202.120.224.81
      [Stream index: 8]
  > User Datagram Protocol, Src Port: 47206, Dst Port: 33434
  > Data (28 bytes)
```

UDP (17)

3. IP header 有多少 bytes? IP 数据报数据载荷有多少 bytes?

Header length 是 20bytes。数据载荷也就是 56-20 = 36bytes。（最底下显示 data 的长

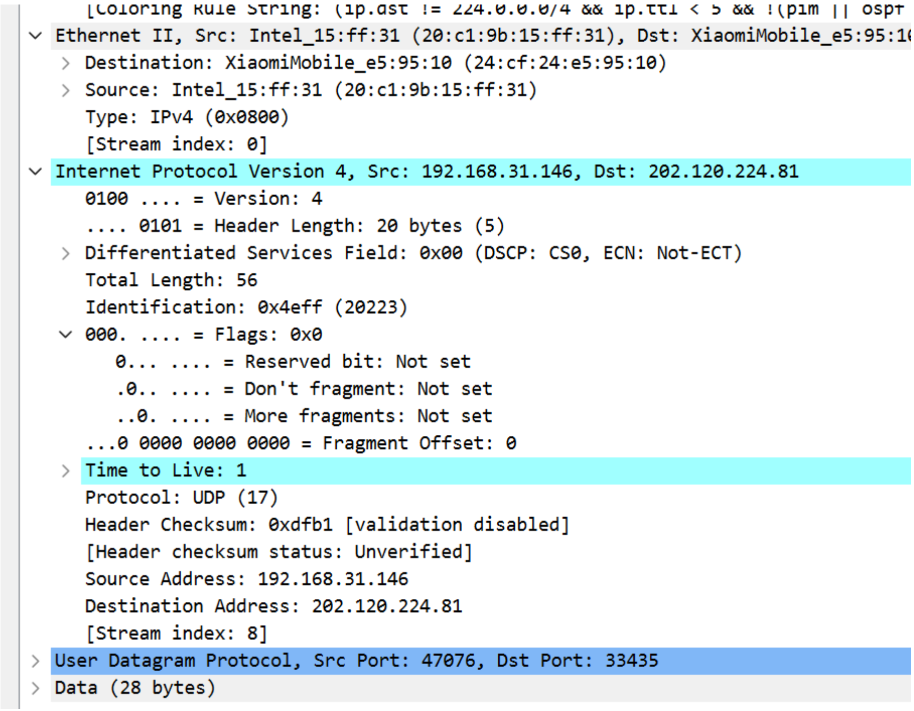
度为 28bytes）

4. 该 IP 数据报是否分片？

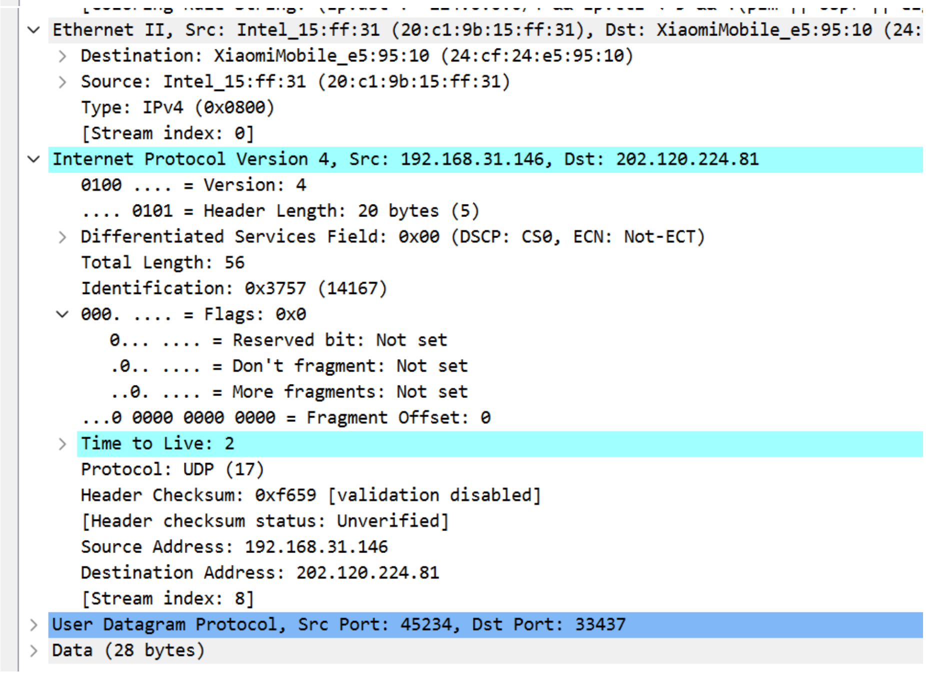
Flag 为 0，所以没有分片。（fragment offset 各种都是 0）

任务 2：观察连续的 UDP 包（穿插其他包），观察：

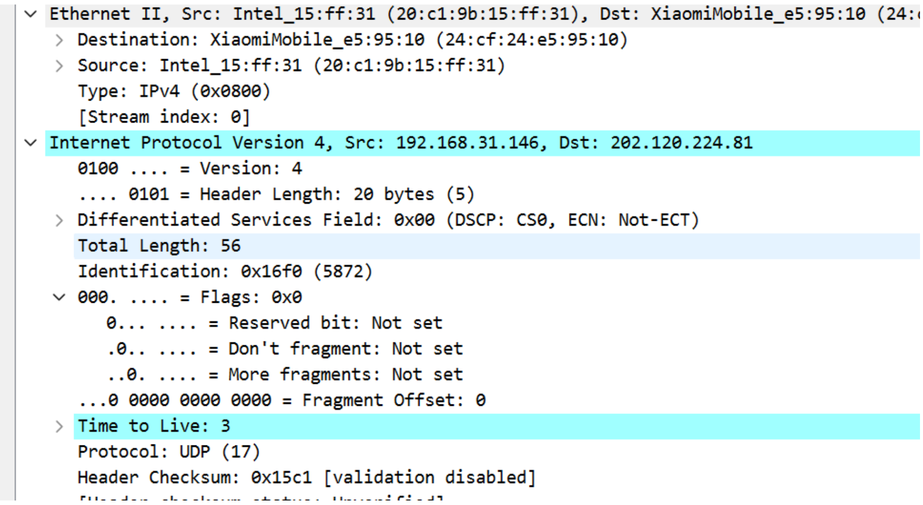
1. IP 数据报中哪些字段不断变化，哪些保持不变 ？



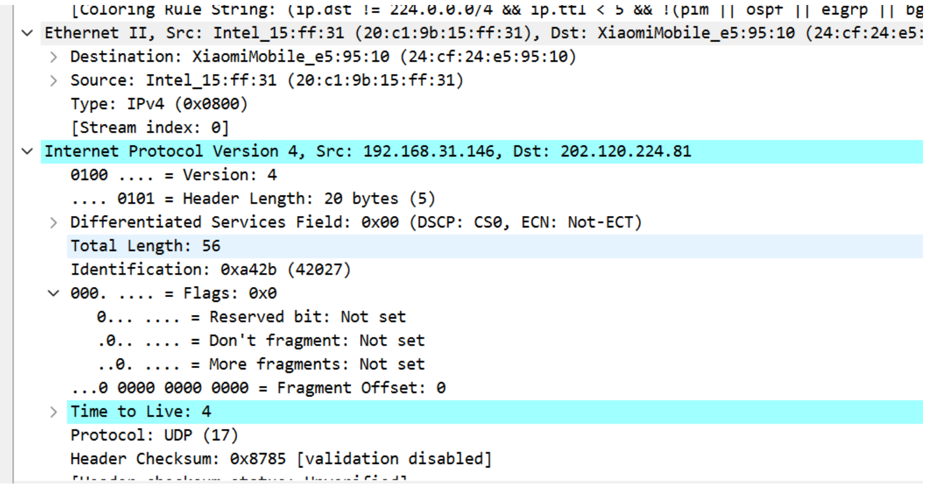
```
    [Coloring Rule String: (ip.dst != 224.0.0.0/4 && ip.ttl < 5 && !(pim || ospf
v Ethernet II, Src: Intel_15:ff:31 (20:c1:9b:15:ff:31), Dst: XiaomiMobile_e5:95:1(
  > Destination: XiaomiMobile_e5:95:10 (24:cf:24:e5:95:10)
  > Source: Intel_15:ff:31 (20:c1:9b:15:ff:31)
    Type: IPv4 (0x0800)
    [Stream index: 0]
v Internet Protocol Version 4, Src: 192.168.31.146, Dst: 202.120.224.81
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 56
    Identification: 0x4eff (20223)
  v 000. .... = Flags: 0x0
      0... .... = Reserved bit: Not set
      .0.. .... = Don't fragment: Not set
      ..0. .... = More fragments: Not set
    ...0 0000 0000 0000 = Fragment Offset: 0
  > Time to Live: 1
    Protocol: UDP (17)
    Header Checksum: 0xdfb1 [validation disabled]
    [Header checksum status: Unverified]
    Source Address: 192.168.31.146
    Destination Address: 202.120.224.81
    [Stream index: 8]
> User Datagram Protocol, Src Port: 47076, Dst Port: 33435
> Data (28 bytes)
```



```
v Ethernet II, Src: Intel_15:ff:31 (20:c1:9b:15:ff:31), Dst: XiaomiMobile_e5:95:10 (24:
  > Destination: XiaomiMobile_e5:95:10 (24:cf:24:e5:95:10)
  > Source: Intel_15:ff:31 (20:c1:9b:15:ff:31)
    Type: IPv4 (0x0800)
    [Stream index: 0]
v Internet Protocol Version 4, Src: 192.168.31.146, Dst: 202.120.224.81
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 56
    Identification: 0x3757 (14167)
  v 000. .... = Flags: 0x0
      0... .... = Reserved bit: Not set
      .0.. .... = Don't fragment: Not set
      ..0. .... = More fragments: Not set
    ...0 0000 0000 0000 = Fragment Offset: 0
  > Time to Live: 2
    Protocol: UDP (17)
    Header Checksum: 0xf659 [validation disabled]
    [Header checksum status: Unverified]
    Source Address: 192.168.31.146
    Destination Address: 202.120.224.81
    [Stream index: 8]
> User Datagram Protocol, Src Port: 45234, Dst Port: 33437
> Data (28 bytes)
```



```
∨ Ethernet II, Src: Intel_15:ff:31 (20:c1:9b:15:ff:31), Dst: XiaomiMobile_e5:95:10 (24:
    > Destination: XiaomiMobile_e5:95:10 (24:cf:24:e5:95:10)
    > Source: Intel_15:ff:31 (20:c1:9b:15:ff:31)
      Type: IPv4 (0x0800)
      [Stream index: 0]
∨ Internet Protocol Version 4, Src: 192.168.31.146, Dst: 202.120.224.81
      0100 .... = Version: 4
      .... 0101 = Header Length: 20 bytes (5)
    > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
      Total Length: 56
      Identification: 0x16f0 (5872)
    ∨ 000. .... = Flags: 0x0
        0... .... = Reserved bit: Not set
        .0.. .... = Don't fragment: Not set
        ..0. .... = More fragments: Not set
      ...0 0000 0000 0000 = Fragment Offset: 0
    > Time to Live: 3
      Protocol: UDP (17)
      Header Checksum: 0x15c1 [validation disabled]
      [Header checksum status: Unverified]
```



```
      [Coloring Rule String: (ip.dst != 224.0.0.0/4 && ip.ttl < 5 && !(pim || ospf || eigrp || bg
∨ Ethernet II, Src: Intel_15:ff:31 (20:c1:9b:15:ff:31), Dst: XiaomiMobile_e5:95:10 (24:cf:24:e5:
    > Destination: XiaomiMobile_e5:95:10 (24:cf:24:e5:95:10)
    > Source: Intel_15:ff:31 (20:c1:9b:15:ff:31)
      Type: IPv4 (0x0800)
      [Stream index: 0]
∨ Internet Protocol Version 4, Src: 192.168.31.146, Dst: 202.120.224.81
      0100 .... = Version: 4
      .... 0101 = Header Length: 20 bytes (5)
    > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
      Total Length: 56
      Identification: 0xa42b (42027)
    ∨ 000. .... = Flags: 0x0
        0... .... = Reserved bit: Not set
        .0.. .... = Don't fragment: Not set
        ..0. .... = More fragments: Not set
      ...0 0000 0000 0000 = Fragment Offset: 0
    > Time to Live: 4
      Protocol: UDP (17)
      Header Checksum: 0x8785 [validation disabled]
      [Header checksum status: Unverified]
```

Identification 在变化，Time to Live 在变化，Header Checksum 在变化

其它部分没有变化。

2. 为什么有些字段不断变化，为什么有些不变 ？

Identification 字段用于对同一个数据包分片进行标识。当 IP 分组被分片（Fragmentation）时，所有分片的 Identification 字段值相同，用于在接收端将分片重新组装成完整的包。而当没有分片时，一般不会有相同 Identification，因为 IP 协议通常仍然会为每个新生成的数据包赋予唯一的 Identification。

Time to Live(TTL)变化，TTL 字段用于限制数据包在网络中的存活时间。每当数据包经过一个路由器时，TTL 会递减 1。如果 TTL 减为 0，数据包会被丢弃，防止在网络中无限循环。Traceroute 通过发送不同 TTL 的报文，来收到每一条设备的超时信息。

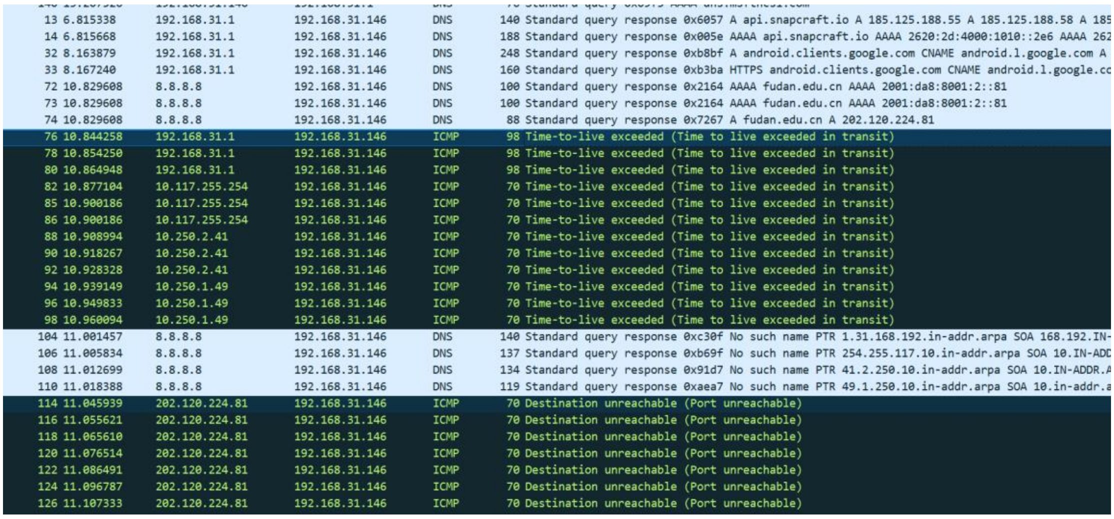
Header Checksum 用来验证 IP 包头完整性，每次 IP 包头中的任何字段发生变化（例如 Identification 和 TTL），都会导致校验和重新计算，因此每个 UDP 包的校验和可能不同，属于很正常的现象。

其它部分不变：例如源地址、目标地址、协议字段肯定是不变的，他们在整个 UDP 通信过程中通常是固定的。因为这些字段描述了通信双方的网络位置和协议类型，在一次通信会话中不需要更改。Flags 一直为 0，因为发送 56bytes 的数据，也不需要分片。

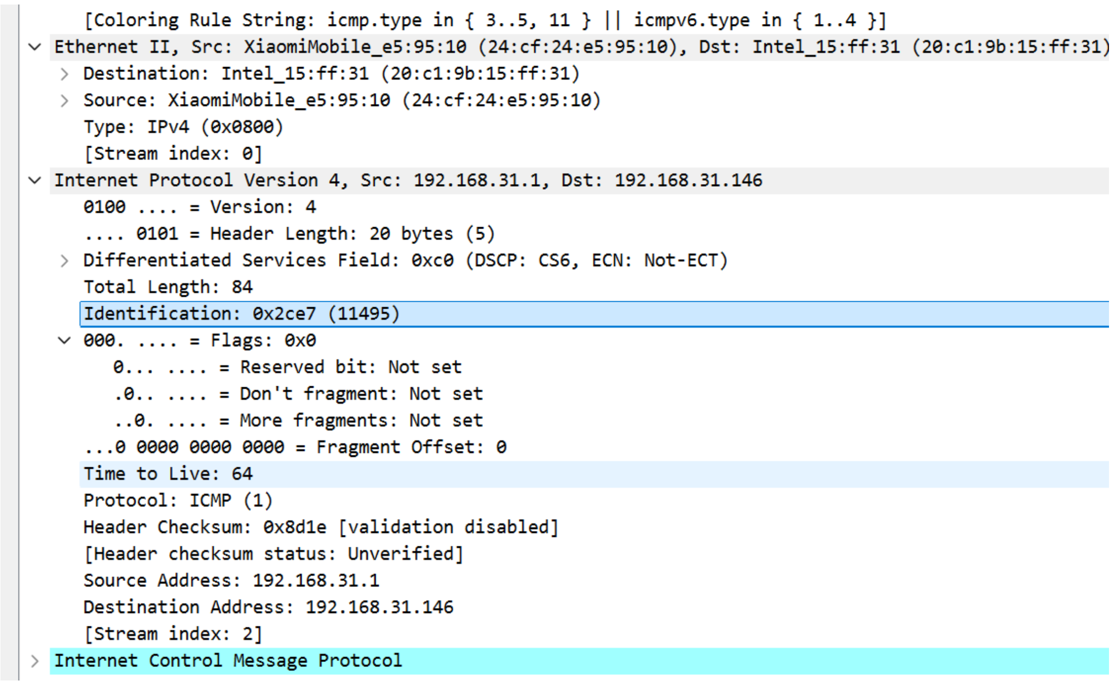
3. 列出连续 IP 数据报中的标识序列。

0x829e　0x4eff　0x3757　0x16f0　0xa42b

任务 3：观察收到的第一个 TTL-exceeded replies，观察：



```
 13 6.815338   192.168.31.1     192.168.31.146   DNS    140 Standard query response 0x6057 A api.snapcraft.io A 185.125.188.55 A 185.125.188.58 A 185
 14 6.815668   192.168.31.1     192.168.31.146   DNS    188 Standard query response 0x005e AAAA api.snapcraft.io AAAA 2620:2d:4000:1010::2e6 AAAA 262
 32 8.163879   192.168.31.1     192.168.31.146   DNS    248 Standard query response 0xb8bf A android.clients.google.com CNAME android.l.google.com A
 33 8.167240   192.168.31.1     192.168.31.146   DNS    160 Standard query response 0xb3ba HTTPS android.clients.google.com CNAME android.l.google.co
 72 10.829608  8.8.8.8          192.168.31.146   DNS    100 Standard query response 0x2164 AAAA fudan.edu.cn AAAA 2001:da8:8001:2::81
 73 10.829608  8.8.8.8          192.168.31.146   DNS    100 Standard query response 0x2164 AAAA fudan.edu.cn AAAA 2001:da8:8001:2::81
 74 10.829608  8.8.8.8          192.168.31.146   DNS     88 Standard query response 0x7267 A fudan.edu.cn A 202.120.224.81
 76 10.844258  192.168.31.1     192.168.31.146   ICMP    98 Time-to-live exceeded (Time to live exceeded in transit)
 78 10.854250  192.168.31.1     192.168.31.146   ICMP    98 Time-to-live exceeded (Time to live exceeded in transit)
 80 10.864948  192.168.31.1     192.168.31.146   ICMP    98 Time-to-live exceeded (Time to live exceeded in transit)
 82 10.877104  10.117.255.254   192.168.31.146   ICMP    70 Time-to-live exceeded (Time to live exceeded in transit)
 85 10.900186  10.117.255.254   192.168.31.146   ICMP    70 Time-to-live exceeded (Time to live exceeded in transit)
 86 10.900186  10.117.255.254   192.168.31.146   ICMP    70 Time-to-live exceeded (Time to live exceeded in transit)
 88 10.908994  10.250.2.41      192.168.31.146   ICMP    70 Time-to-live exceeded (Time to live exceeded in transit)
 90 10.918267  10.250.2.41      192.168.31.146   ICMP    70 Time-to-live exceeded (Time to live exceeded in transit)
 92 10.928328  10.250.2.41      192.168.31.146   ICMP    70 Time-to-live exceeded (Time to live exceeded in transit)
 94 10.939149  10.250.1.49      192.168.31.146   ICMP    70 Time-to-live exceeded (Time to live exceeded in transit)
 96 10.949833  10.250.1.49      192.168.31.146   ICMP    70 Time-to-live exceeded (Time to live exceeded in transit)
 98 10.960094  10.250.1.49      192.168.31.146   ICMP    70 Time-to-live exceeded (Time to live exceeded in transit)
104 11.001457  8.8.8.8          192.168.31.146   DNS    140 Standard query response 0xc30f No such name PTR 1.31.168.192.in-addr.arpa SOA 168.192.IN-
106 11.005834  8.8.8.8          192.168.31.146   DNS    137 Standard query response 0xb69f No such name PTR 254.255.117.10.in-addr.arpa SOA 10.IN-ADD
108 11.012699  8.8.8.8          192.168.31.146   DNS    134 Standard query response 0x91d7 No such name PTR 41.2.250.10.in-addr.arpa SOA 10.IN-ADDR.A
110 11.018388  8.8.8.8          192.168.31.146   DNS    119 Standard query response 0xaea7 No such name PTR 49.1.250.10.in-addr.arpa SOA 10.in-addr.a
114 11.045939  202.120.224.81   192.168.31.146   ICMP    70 Destination unreachable (Port unreachable)
116 11.055621  202.120.224.81   192.168.31.146   ICMP    70 Destination unreachable (Port unreachable)
118 11.065610  202.120.224.81   192.168.31.146   ICMP    70 Destination unreachable (Port unreachable)
120 11.076514  202.120.224.81   192.168.31.146   ICMP    70 Destination unreachable (Port unreachable)
122 11.086491  202.120.224.81   192.168.31.146   ICMP    70 Destination unreachable (Port unreachable)
124 11.096787  202.120.224.81   192.168.31.146   ICMP    70 Destination unreachable (Port unreachable)
126 11.107333  202.120.224.81   192.168.31.146   ICMP    70 Destination unreachable (Port unreachable)
```

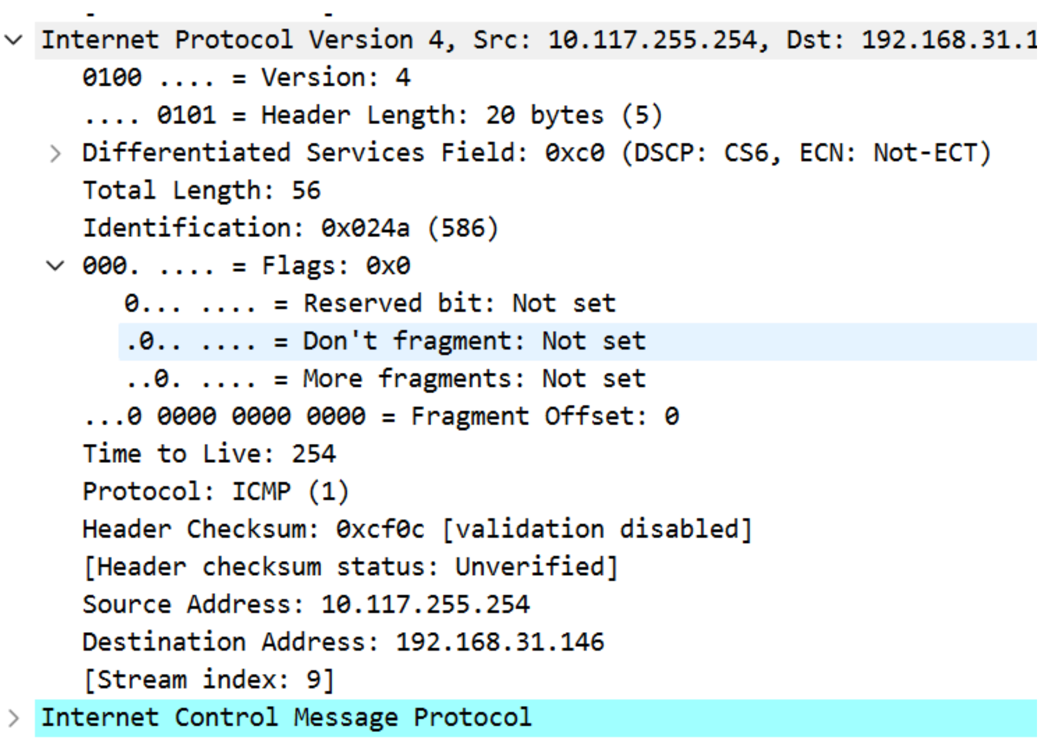


```
    [Coloring Rule String: icmp.type in { 3..5, 11 } || icmpv6.type in { 1..4 }]
  Ethernet II, Src: XiaomiMobile_e5:95:10 (24:cf:24:e5:95:10), Dst: Intel_15:ff:31 (20:c1:9b:15:ff:31)
    Destination: Intel_15:ff:31 (20:c1:9b:15:ff:31)
    Source: XiaomiMobile_e5:95:10 (24:cf:24:e5:95:10)
    Type: IPv4 (0x0800)
    [Stream index: 0]
  Internet Protocol Version 4, Src: 192.168.31.1, Dst: 192.168.31.146
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
    Differentiated Services Field: 0xc0 (DSCP: CS6, ECN: Not-ECT)
    Total Length: 84
    Identification: 0x2ce7 (11495)
    000. .... = Flags: 0x0
      0... .... = Reserved bit: Not set
      .0.. .... = Don't fragment: Not set
      ..0. .... = More fragments: Not set
    ...0 0000 0000 0000 = Fragment Offset: 0
    Time to Live: 64
    Protocol: ICMP (1)
    Header Checksum: 0x8d1e [validation disabled]
    [Header checksum status: Unverified]
    Source Address: 192.168.31.1
    Destination Address: 192.168.31.146
    [Stream index: 2]
  Internet Control Message Protocol
```

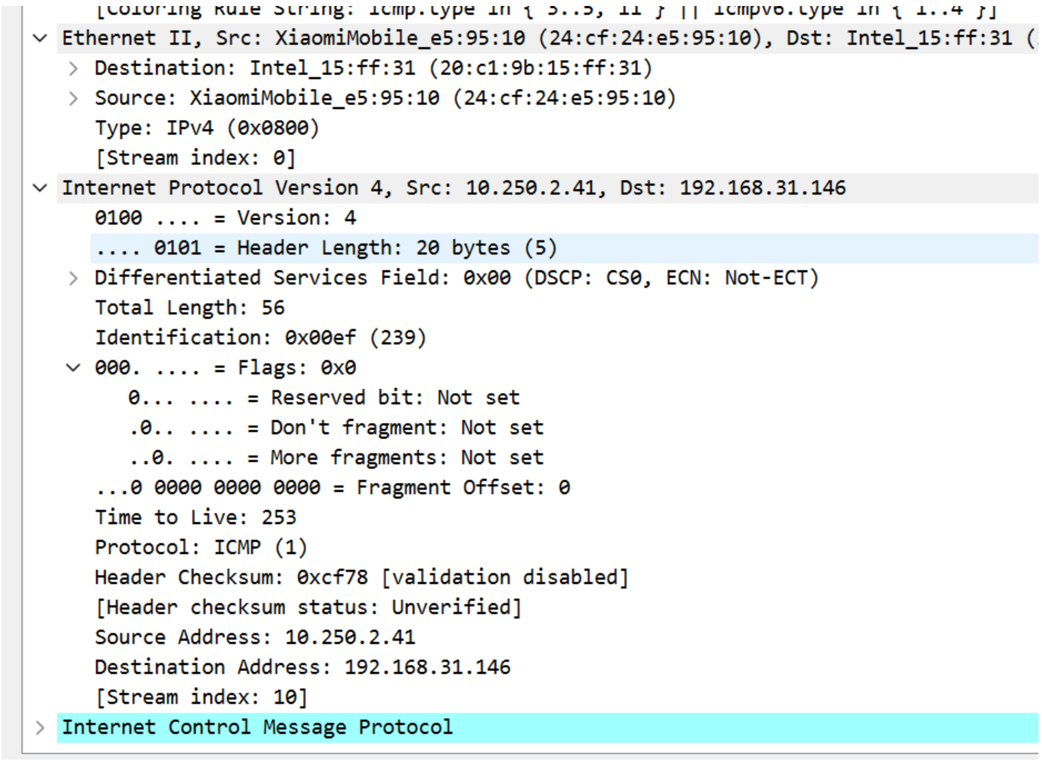
1. 标识字段与 TTL 字段分别是多少？

标识字段（identification）是 0x2ce7　TTL 字段是 64

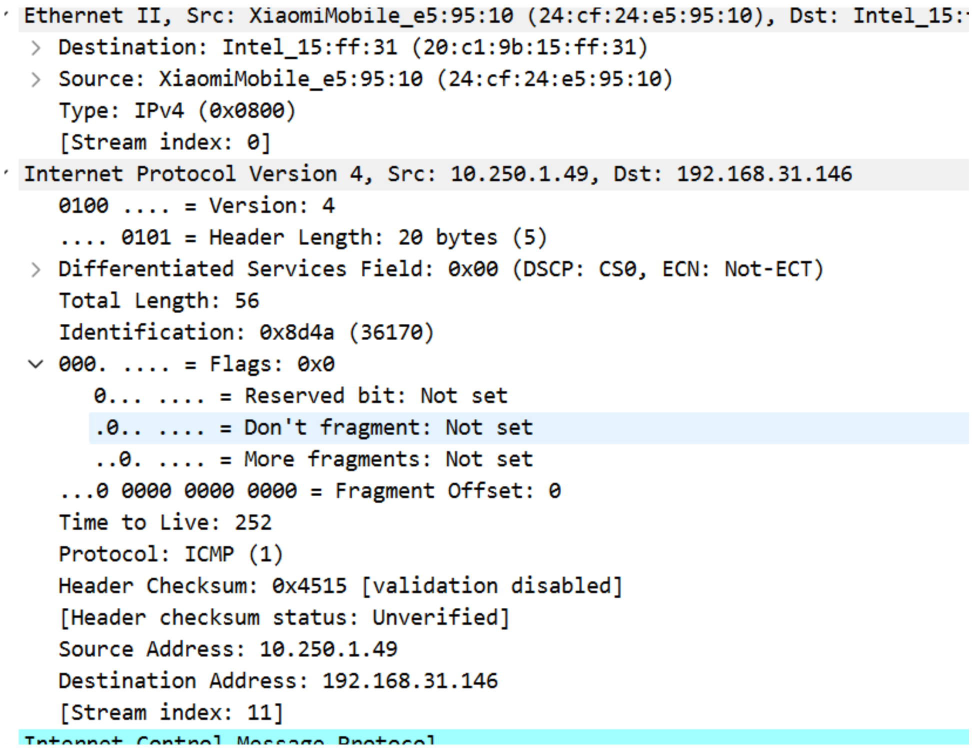
2. 收到的所有 TTL-exceeded replies 中，这两个字段是否不变？为什么？



```
v Internet Protocol Version 4, Src: 10.117.255.254, Dst: 192.168.31.1
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0xc0 (DSCP: CS6, ECN: Not-ECT)
    Total Length: 56
    Identification: 0x024a (586)
  v 000. .... = Flags: 0x0
      0... .... = Reserved bit: Not set
      .0.. .... = Don't fragment: Not set
      ..0. .... = More fragments: Not set
    ...0 0000 0000 0000 = Fragment Offset: 0
    Time to Live: 254
    Protocol: ICMP (1)
    Header Checksum: 0xcf0c [validation disabled]
    [Header checksum status: Unverified]
    Source Address: 10.117.255.254
    Destination Address: 192.168.31.146
    [Stream index: 9]
> Internet Control Message Protocol
```



```
    [Coloring Rule String: icmp.type in { 3..5, 11 } || icmpv6.type in { 1..4 }]
v Ethernet II, Src: XiaomiMobile_e5:95:10 (24:cf:24:e5:95:10), Dst: Intel_15:ff:31 (
  > Destination: Intel_15:ff:31 (20:c1:9b:15:ff:31)
  > Source: XiaomiMobile_e5:95:10 (24:cf:24:e5:95:10)
    Type: IPv4 (0x0800)
    [Stream index: 0]
v Internet Protocol Version 4, Src: 10.250.2.41, Dst: 192.168.31.146
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 56
    Identification: 0x00ef (239)
  v 000. .... = Flags: 0x0
      0... .... = Reserved bit: Not set
      .0.. .... = Don't fragment: Not set
      ..0. .... = More fragments: Not set
    ...0 0000 0000 0000 = Fragment Offset: 0
    Time to Live: 253
    Protocol: ICMP (1)
    Header Checksum: 0xcf78 [validation disabled]
    [Header checksum status: Unverified]
    Source Address: 10.250.2.41
    Destination Address: 192.168.31.146
    [Stream index: 10]
> Internet Control Message Protocol
```



```
Ethernet II, Src: XiaomiMobile_e5:95:10 (24:cf:24:e5:95:10), Dst: Intel_15:
 › Destination: Intel_15:ff:31 (20:c1:9b:15:ff:31)
 › Source: XiaomiMobile_e5:95:10 (24:cf:24:e5:95:10)
   Type: IPv4 (0x0800)
   [Stream index: 0]
Internet Protocol Version 4, Src: 10.250.1.49, Dst: 192.168.31.146
   0100 .... = Version: 4
   .... 0101 = Header Length: 20 bytes (5)
 › Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
   Total Length: 56
   Identification: 0x8d4a (36170)
 ∨ 000. .... = Flags: 0x0
      0... .... = Reserved bit: Not set
      .0.. .... = Don't fragment: Not set
      ..0. .... = More fragments: Not set
   ...0 0000 0000 0000 = Fragment Offset: 0
   Time to Live: 252
   Protocol: ICMP (1)
   Header Checksum: 0x4515 [validation disabled]
   [Header checksum status: Unverified]
   Source Address: 10.250.1.49
   Destination Address: 192.168.31.146
   [Stream index: 11]
Internet Control Message Protocol
```

随便挑几个看发现均有改变。

标识字段（identification）会改变是正常的，因为他们并不是一个分片，所以按照随机分配 identification 的策略来看不同是正常的。

TTL 不同，Traceroute 发送的就是不同 TTL 的报文，这里不同也是正常的。

Part2：Mininet

任务 1

解决error后才开始creating…的问题：wsl2功能补全，用linux虚拟机即可。

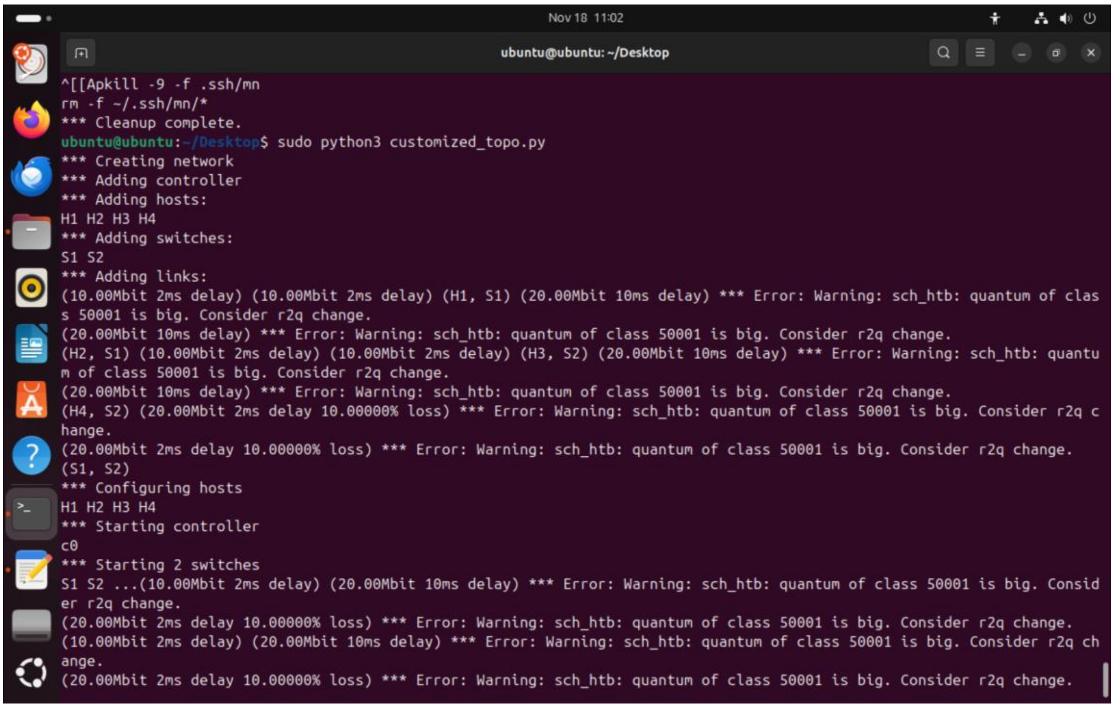
解决*** Starting controllerc0 Cannot find required executable ovs-controller报错问题：未链接好导致的——[Cannot find required executable controller_cannot find required executable ovs-vsctl-CSDN博客]

解决Error: Warning: sch htb: quantum of class 50001 is big. Consider r2g change.：其实不是这个的问题，iperf一直卡住。

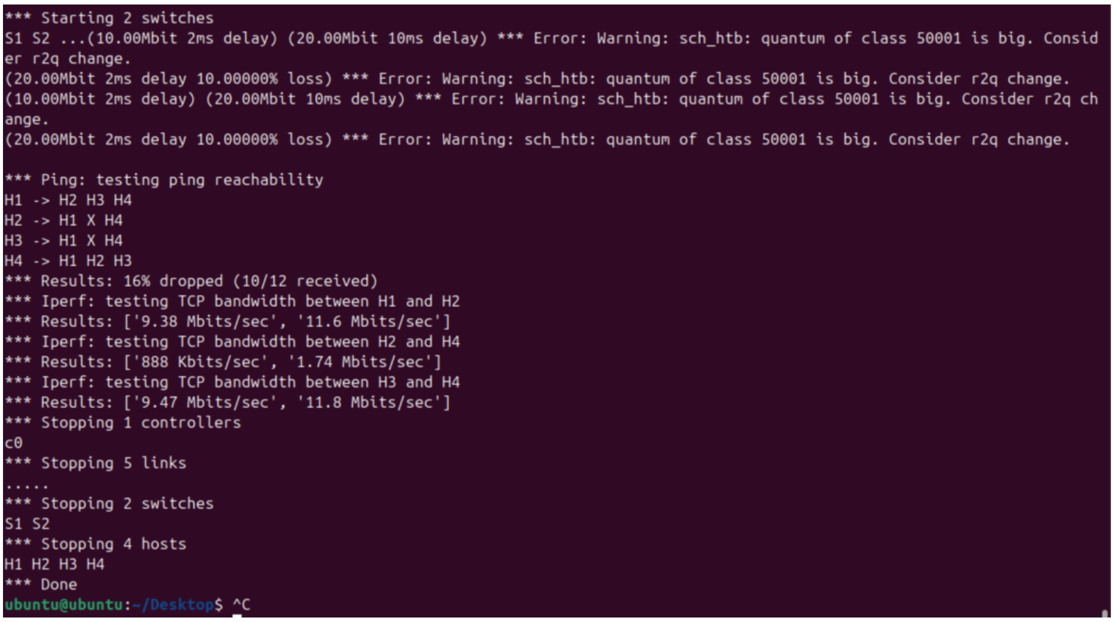
一边terminal开着iperf -s -p 5001，另一端跑代码，会发现这个终端直接跑的killed了。

开着top跑，可以看到cpu占用低于10%，不是oom错误。

最终发现是版本问题，iperf太新了导致ubuntu跟不上出错。开虚拟环境覆盖下载低版本的iperf即可。最终 mininet 2.3.0 iperf 2.0.9



```
^[[Apkill -9 -f .ssh/mn
rm -f ~/.ssh/mn/*
*** Cleanup complete.
ubuntu@ubuntu:~/Desktop$ sudo python3 customized_topo.py
*** Creating network
*** Adding controller
*** Adding hosts:
H1 H2 H3 H4
*** Adding switches:
S1 S2
*** Adding links:
(10.00Mbit 2ms delay) (10.00Mbit 2ms delay) (H1, S1) (20.00Mbit 10ms delay) *** Error: Warning: sch_htb: quantum of clas
s 50001 is big. Consider r2q change.
(20.00Mbit 10ms delay) *** Error: Warning: sch_htb: quantum of class 50001 is big. Consider r2q change.
(H2, S1) (10.00Mbit 2ms delay) (10.00Mbit 2ms delay) (H3, S2) (20.00Mbit 10ms delay) *** Error: Warning: sch_htb: quantu
m of class 50001 is big. Consider r2q change.
(20.00Mbit 10ms delay) *** Error: Warning: sch_htb: quantum of class 50001 is big. Consider r2q change.
(H4, S2) (20.00Mbit 2ms delay 10.00000% loss) *** Error: Warning: sch_htb: quantum of class 50001 is big. Consider r2q c
hange.
(20.00Mbit 2ms delay 10.00000% loss) *** Error: Warning: sch_htb: quantum of class 50001 is big. Consider r2q change.
(S1, S2)
*** Configuring hosts
H1 H2 H3 H4
*** Starting controller
c0
*** Starting 2 switches
S1 S2 ...(10.00Mbit 2ms delay) (20.00Mbit 10ms delay) *** Error: Warning: sch_htb: quantum of class 50001 is big. Consid
er r2q change.
(20.00Mbit 2ms delay 10.00000% loss) *** Error: Warning: sch_htb: quantum of class 50001 is big. Consider r2q change.
(10.00Mbit 2ms delay) (20.00Mbit 10ms delay) *** Error: Warning: sch_htb: quantum of class 50001 is big. Consider r2q ch
ange.
(20.00Mbit 2ms delay 10.00000% loss) *** Error: Warning: sch_htb: quantum of class 50001 is big. Consider r2q change.
```



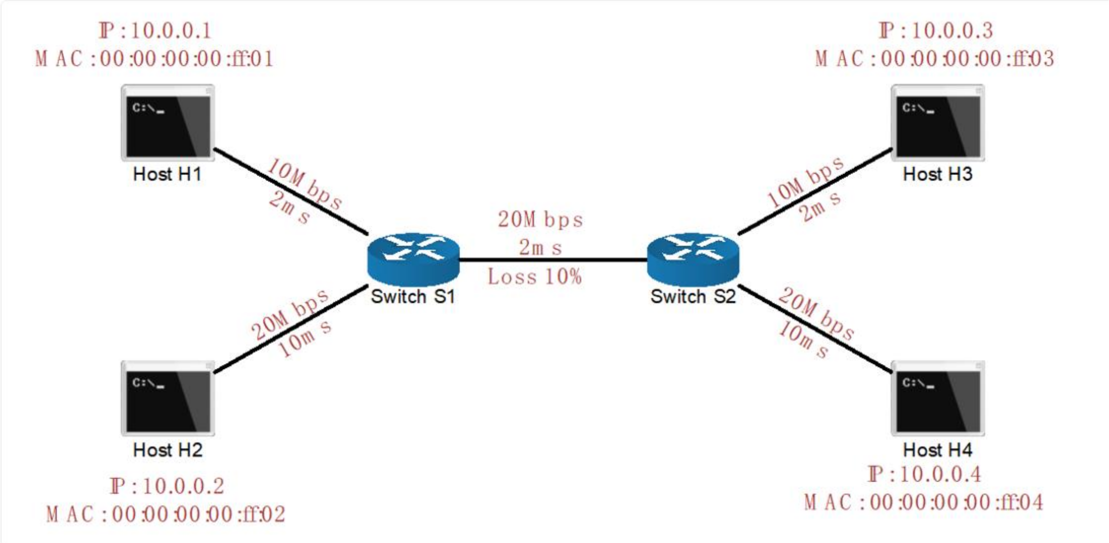
```
*** Starting 2 switches
S1 S2 ...(10.00Mbit 2ms delay) (20.00Mbit 10ms delay) *** Error: Warning: sch_htb: quantum of class 50001 is big. Consid
er r2q change.
(20.00Mbit 2ms delay 10.00000% loss) *** Error: Warning: sch_htb: quantum of class 50001 is big. Consider r2q change.
(10.00Mbit 2ms delay) (20.00Mbit 10ms delay) *** Error: Warning: sch_htb: quantum of class 50001 is big. Consider r2q ch
ange.
(20.00Mbit 2ms delay 10.00000% loss) *** Error: Warning: sch_htb: quantum of class 50001 is big. Consider r2q change.

*** Ping: testing ping reachability
H1 -> H2 H3 H4
H2 -> H1 X H4
H3 -> H1 X H4
H4 -> H1 H2 H3
*** Results: 16% dropped (10/12 received)
*** Iperf: testing TCP bandwidth between H1 and H2
*** Results: ['9.38 Mbits/sec', '11.6 Mbits/sec']
*** Iperf: testing TCP bandwidth between H2 and H4
*** Results: ['888 Kbits/sec', '1.74 Mbits/sec']
*** Iperf: testing TCP bandwidth between H3 and H4
*** Results: ['9.47 Mbits/sec', '11.8 Mbits/sec']
*** Stopping 1 controllers
c0
*** Stopping 5 links
.....
*** Stopping 2 switches
S1 S2
*** Stopping 4 hosts
H1 H2 H3 H4
*** Done
ubuntu@ubuntu:~/Desktop$ ^C
```

丢包率未 16%，10/12（其实多次实验发现丢 0 个包—丢 3 个包都有发生，大部分情况是 1 个或两个丢包），和我们设的丢包率相近。

H1 和 H2 间拓扑：［‘9.38Mbps’，　’11.6Mbps’］和实验设置（10Mbps）相近

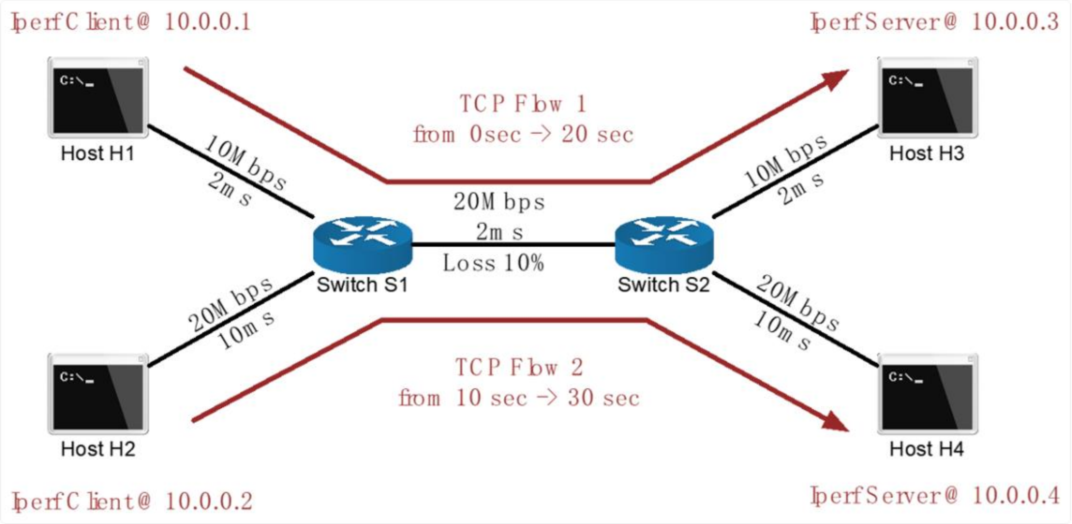
H2 和 H4 间拓扑：［‘888Kbps’，　‘1.74Mbps’］这一条倒是和实验设置相差较大，实验设置 H2 到 H4 之间三条串联的 link 都是 20Mbps，估计是丢包率的原因。

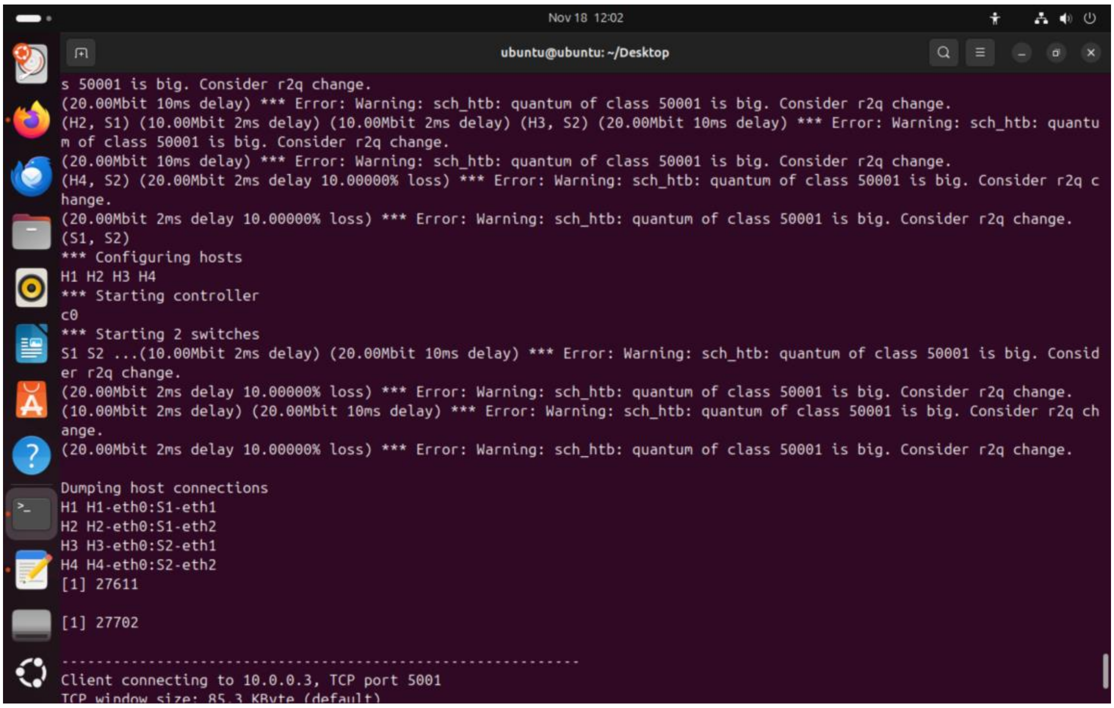
H3 和 H4 间拓扑：［‘9.47Mbps’，　’11.8Mbps’］和实验设置（10Mbps）相近



（直接运行 python 脚本即是以上内容。若需通过 sudo mn --custom ./customized_topo.py --topo mytopo --test pingall --link tc 指令检验，在 main 函数外多注册一个：Topos = {‘MyTopo’:MyTopo} 即可，同样结果。）

任务 2

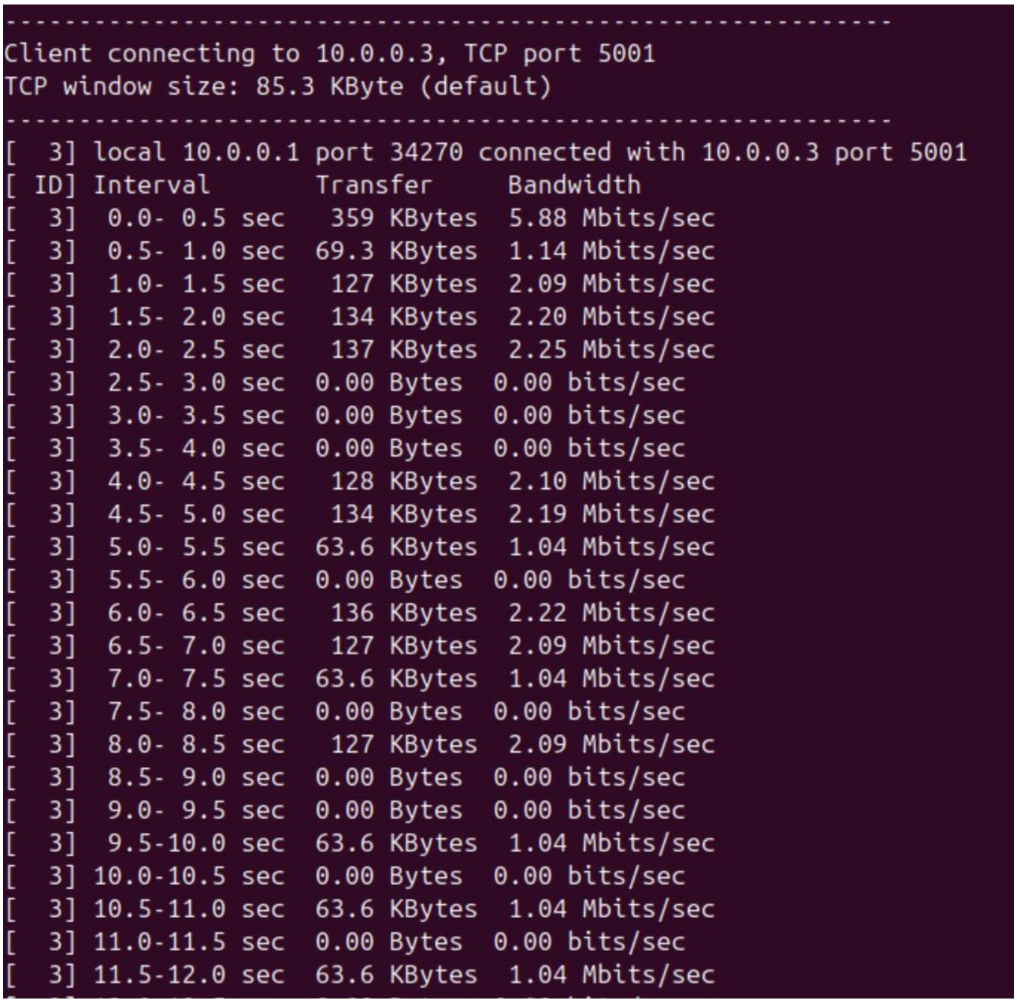




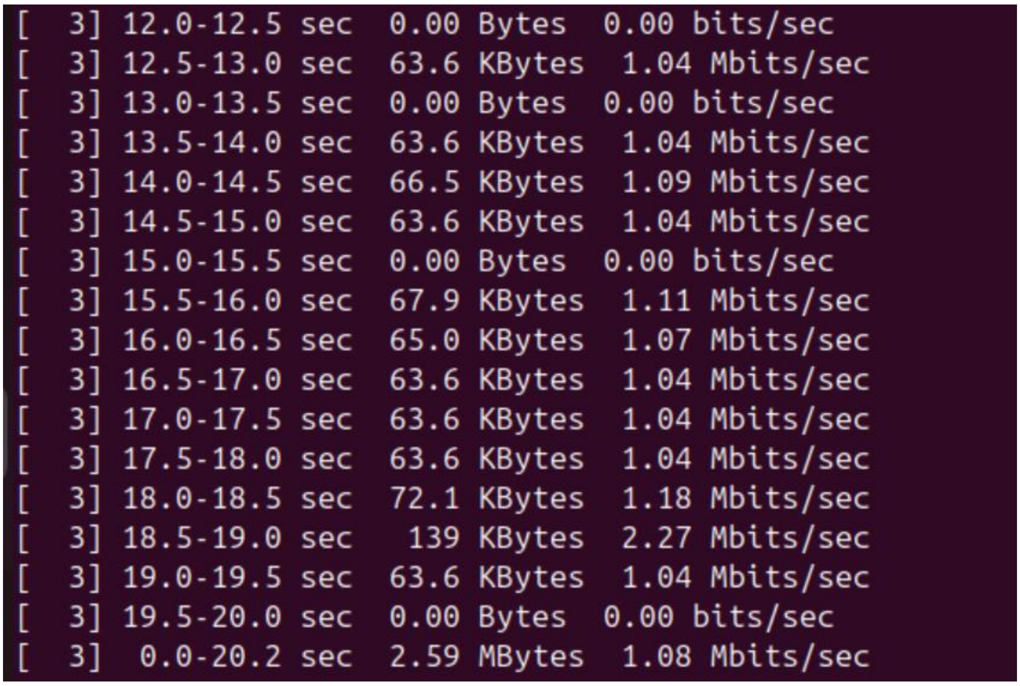
```
s 50001 is big. Consider r2q change.
(20.00Mbit 10ms delay) *** Error: Warning: sch_htb: quantum of class 50001 is big. Consider r2q change.
(H2, S1) (10.00Mbit 2ms delay) (10.00Mbit 2ms delay) (H3, S2) (20.00Mbit 10ms delay) *** Error: Warning: sch_htb: quantu
m of class 50001 is big. Consider r2q change.
(20.00Mbit 10ms delay) *** Error: Warning: sch_htb: quantum of class 50001 is big. Consider r2q change.
(H4, S2) (20.00Mbit 2ms delay 10.00000% loss) *** Error: Warning: sch_htb: quantum of class 50001 is big. Consider r2q c
hange.
(20.00Mbit 2ms delay 10.00000% loss) *** Error: Warning: sch_htb: quantum of class 50001 is big. Consider r2q change.
(S1, S2)
*** Configuring hosts
H1 H2 H3 H4
*** Starting controller
c0
*** Starting 2 switches
S1 S2 ...(10.00Mbit 2ms delay) (20.00Mbit 10ms delay) *** Error: Warning: sch_htb: quantum of class 50001 is big. Consid
er r2q change.
(20.00Mbit 2ms delay 10.00000% loss) *** Error: Warning: sch_htb: quantum of class 50001 is big. Consider r2q change.
(10.00Mbit 2ms delay) (20.00Mbit 10ms delay) *** Error: Warning: sch_htb: quantum of class 50001 is big. Consider r2q ch
ange.
(20.00Mbit 2ms delay 10.00000% loss) *** Error: Warning: sch_htb: quantum of class 50001 is big. Consider r2q change.

Dumping host connections
H1 H1-eth0:S1-eth1
H2 H2-eth0:S1-eth2
H3 H3-eth0:S2-eth1
H4 H4-eth0:S2-eth2
[1] 27611

[1] 27702

------------------------------------------------------------
Client connecting to 10.0.0.3, TCP port 5001
TCP window size: 85.3 KByte (default)
```

实验结果：

Flow1 输出结果（0.5s 测量一次）

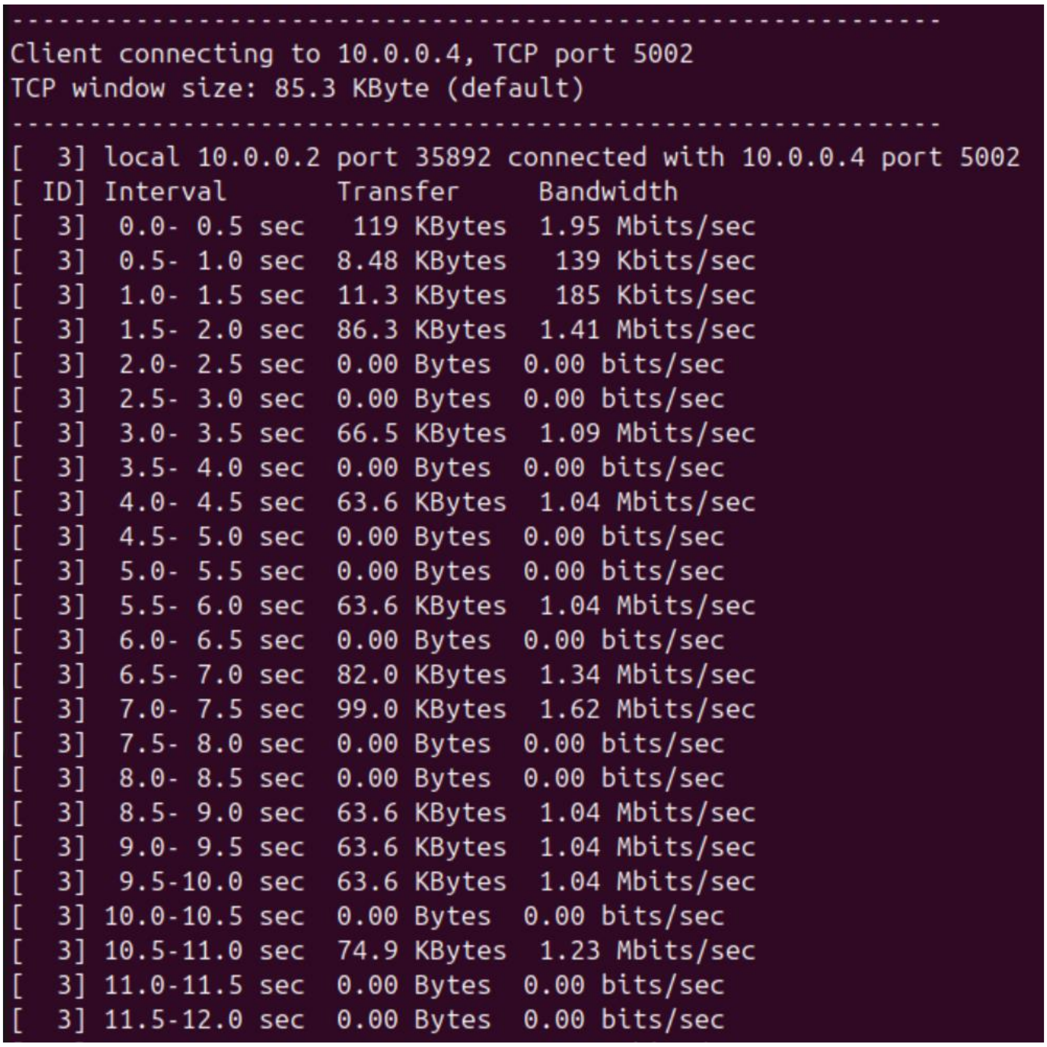


```
------------------------------------------------------------
Client connecting to 10.0.0.3, TCP port 5001
TCP window size: 85.3 KByte (default)
------------------------------------------------------------
[  3] local 10.0.0.1 port 34270 connected with 10.0.0.3 port 5001
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0- 0.5 sec   359 KBytes  5.88 Mbits/sec
[  3]  0.5- 1.0 sec  69.3 KBytes  1.14 Mbits/sec
[  3]  1.0- 1.5 sec   127 KBytes  2.09 Mbits/sec
[  3]  1.5- 2.0 sec   134 KBytes  2.20 Mbits/sec
[  3]  2.0- 2.5 sec   137 KBytes  2.25 Mbits/sec
[  3]  2.5- 3.0 sec  0.00 Bytes  0.00 bits/sec
[  3]  3.0- 3.5 sec  0.00 Bytes  0.00 bits/sec
[  3]  3.5- 4.0 sec  0.00 Bytes  0.00 bits/sec
[  3]  4.0- 4.5 sec   128 KBytes  2.10 Mbits/sec
[  3]  4.5- 5.0 sec   134 KBytes  2.19 Mbits/sec
[  3]  5.0- 5.5 sec  63.6 KBytes  1.04 Mbits/sec
[  3]  5.5- 6.0 sec  0.00 Bytes  0.00 bits/sec
[  3]  6.0- 6.5 sec   136 KBytes  2.22 Mbits/sec
[  3]  6.5- 7.0 sec   127 KBytes  2.09 Mbits/sec
[  3]  7.0- 7.5 sec  63.6 KBytes  1.04 Mbits/sec
[  3]  7.5- 8.0 sec  0.00 Bytes  0.00 bits/sec
[  3]  8.0- 8.5 sec   127 KBytes  2.09 Mbits/sec
[  3]  8.5- 9.0 sec  0.00 Bytes  0.00 bits/sec
[  3]  9.0- 9.5 sec  0.00 Bytes  0.00 bits/sec
[  3]  9.5-10.0 sec  63.6 KBytes  1.04 Mbits/sec
[  3] 10.0-10.5 sec  0.00 Bytes  0.00 bits/sec
[  3] 10.5-11.0 sec  63.6 KBytes  1.04 Mbits/sec
[  3] 11.0-11.5 sec  0.00 Bytes  0.00 bits/sec
[  3] 11.5-12.0 sec  63.6 KBytes  1.04 Mbits/sec
```

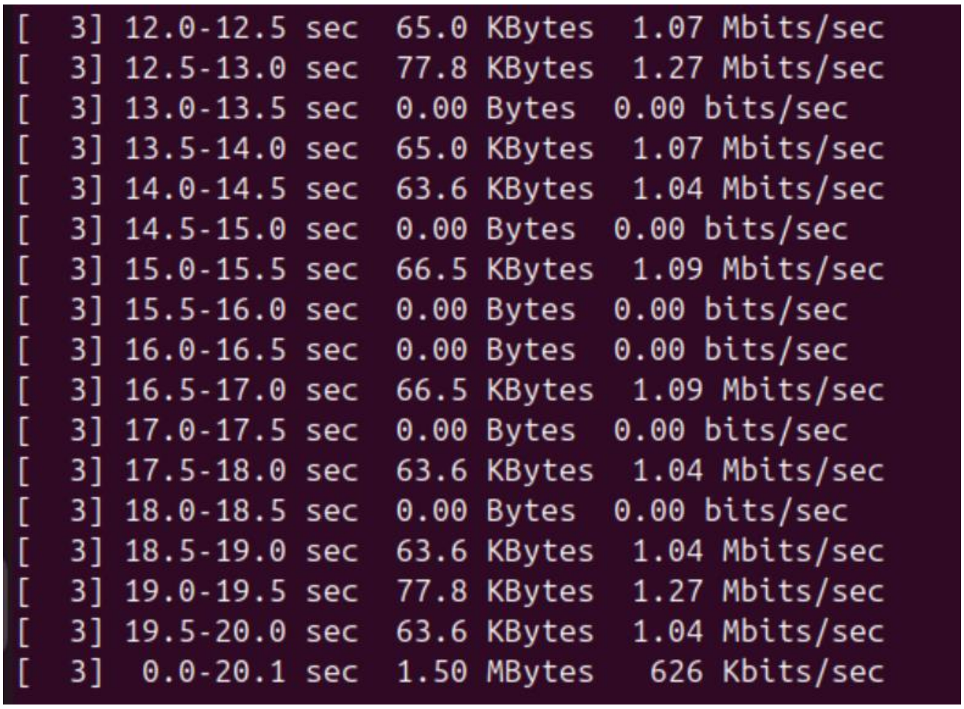


```
[  3] 12.0-12.5 sec  0.00 Bytes  0.00 bits/sec
[  3] 12.5-13.0 sec  63.6 KBytes  1.04 Mbits/sec
[  3] 13.0-13.5 sec  0.00 Bytes  0.00 bits/sec
[  3] 13.5-14.0 sec  63.6 KBytes  1.04 Mbits/sec
[  3] 14.0-14.5 sec  66.5 KBytes  1.09 Mbits/sec
[  3] 14.5-15.0 sec  63.6 KBytes  1.04 Mbits/sec
[  3] 15.0-15.5 sec  0.00 Bytes  0.00 bits/sec
[  3] 15.5-16.0 sec  67.9 KBytes  1.11 Mbits/sec
[  3] 16.0-16.5 sec  65.0 KBytes  1.07 Mbits/sec
[  3] 16.5-17.0 sec  63.6 KBytes  1.04 Mbits/sec
[  3] 17.0-17.5 sec  63.6 KBytes  1.04 Mbits/sec
[  3] 17.5-18.0 sec  63.6 KBytes  1.04 Mbits/sec
[  3] 18.0-18.5 sec  72.1 KBytes  1.18 Mbits/sec
[  3] 18.5-19.0 sec   139 KBytes  2.27 Mbits/sec
[  3] 19.0-19.5 sec  63.6 KBytes  1.04 Mbits/sec
[  3] 19.5-20.0 sec  0.00 Bytes  0.00 bits/sec
[  3]  0.0-20.2 sec  2.59 MBytes  1.08 Mbits/sec
```

Flow2 输出结果（0.5s 测量一次）



```
------------------------------------------------------------
Client connecting to 10.0.0.4, TCP port 5002
TCP window size: 85.3 KByte (default)
------------------------------------------------------------
[  3] local 10.0.0.2 port 35892 connected with 10.0.0.4 port 5002
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0- 0.5 sec   119 KBytes  1.95 Mbits/sec
[  3]  0.5- 1.0 sec  8.48 KBytes   139 Kbits/sec
[  3]  1.0- 1.5 sec  11.3 KBytes   185 Kbits/sec
[  3]  1.5- 2.0 sec  86.3 KBytes  1.41 Mbits/sec
[  3]  2.0- 2.5 sec  0.00 Bytes  0.00 bits/sec
[  3]  2.5- 3.0 sec  0.00 Bytes  0.00 bits/sec
[  3]  3.0- 3.5 sec  66.5 KBytes  1.09 Mbits/sec
[  3]  3.5- 4.0 sec  0.00 Bytes  0.00 bits/sec
[  3]  4.0- 4.5 sec  63.6 KBytes  1.04 Mbits/sec
[  3]  4.5- 5.0 sec  0.00 Bytes  0.00 bits/sec
[  3]  5.0- 5.5 sec  0.00 Bytes  0.00 bits/sec
[  3]  5.5- 6.0 sec  63.6 KBytes  1.04 Mbits/sec
[  3]  6.0- 6.5 sec  0.00 Bytes  0.00 bits/sec
[  3]  6.5- 7.0 sec  82.0 KBytes  1.34 Mbits/sec
[  3]  7.0- 7.5 sec  99.0 KBytes  1.62 Mbits/sec
[  3]  7.5- 8.0 sec  0.00 Bytes  0.00 bits/sec
[  3]  8.0- 8.5 sec  0.00 Bytes  0.00 bits/sec
[  3]  8.5- 9.0 sec  63.6 KBytes  1.04 Mbits/sec
[  3]  9.0- 9.5 sec  63.6 KBytes  1.04 Mbits/sec
[  3]  9.5-10.0 sec  63.6 KBytes  1.04 Mbits/sec
[  3] 10.0-10.5 sec  0.00 Bytes  0.00 bits/sec
[  3] 10.5-11.0 sec  74.9 KBytes  1.23 Mbits/sec
[  3] 11.0-11.5 sec  0.00 Bytes  0.00 bits/sec
[  3] 11.5-12.0 sec  0.00 Bytes  0.00 bits/sec
```



```
[  3] 12.0-12.5 sec  65.0 KBytes  1.07 Mbits/sec
[  3] 12.5-13.0 sec  77.8 KBytes  1.27 Mbits/sec
[  3] 13.0-13.5 sec  0.00 Bytes  0.00 bits/sec
[  3] 13.5-14.0 sec  65.0 KBytes  1.07 Mbits/sec
[  3] 14.0-14.5 sec  63.6 KBytes  1.04 Mbits/sec
[  3] 14.5-15.0 sec  0.00 Bytes  0.00 bits/sec
[  3] 15.0-15.5 sec  66.5 KBytes  1.09 Mbits/sec
[  3] 15.5-16.0 sec  0.00 Bytes  0.00 bits/sec
[  3] 16.0-16.5 sec  0.00 Bytes  0.00 bits/sec
[  3] 16.5-17.0 sec  66.5 KBytes  1.09 Mbits/sec
[  3] 17.0-17.5 sec  0.00 Bytes  0.00 bits/sec
[  3] 17.5-18.0 sec  63.6 KBytes  1.04 Mbits/sec
[  3] 18.0-18.5 sec  0.00 Bytes  0.00 bits/sec
[  3] 18.5-19.0 sec  63.6 KBytes  1.04 Mbits/sec
[  3] 19.0-19.5 sec  77.8 KBytes  1.27 Mbits/sec
[  3] 19.5-20.0 sec  63.6 KBytes  1.04 Mbits/sec
[  3]  0.0-20.1 sec  1.50 MBytes   626 Kbits/sec
```

```
h3.cmd(’iperf -s -p 5001 & ’)
h4.cmd(’iperf -s -p 5002 & ’)
h1.cmd(’iperf -c 10.0.0.3 -p 5001 -t 20 -i 0.5’)
h2.cmd(’iperf -c 10.0.0.4 -p 5002 -t 20 -i 0.5’)
```

将 host3 host4 作为服务器（server），host1 host2 作为客户端（client），同时使用 iperf 构成 tcp 流，将 host3 与 host1 绑定，（h1 对应 h3 的 ip 地址，同时选用相同的 port），host4 绑定 host2 同理。并且时间 20s，interval 为 0.5，每 0.5s 测量一次。

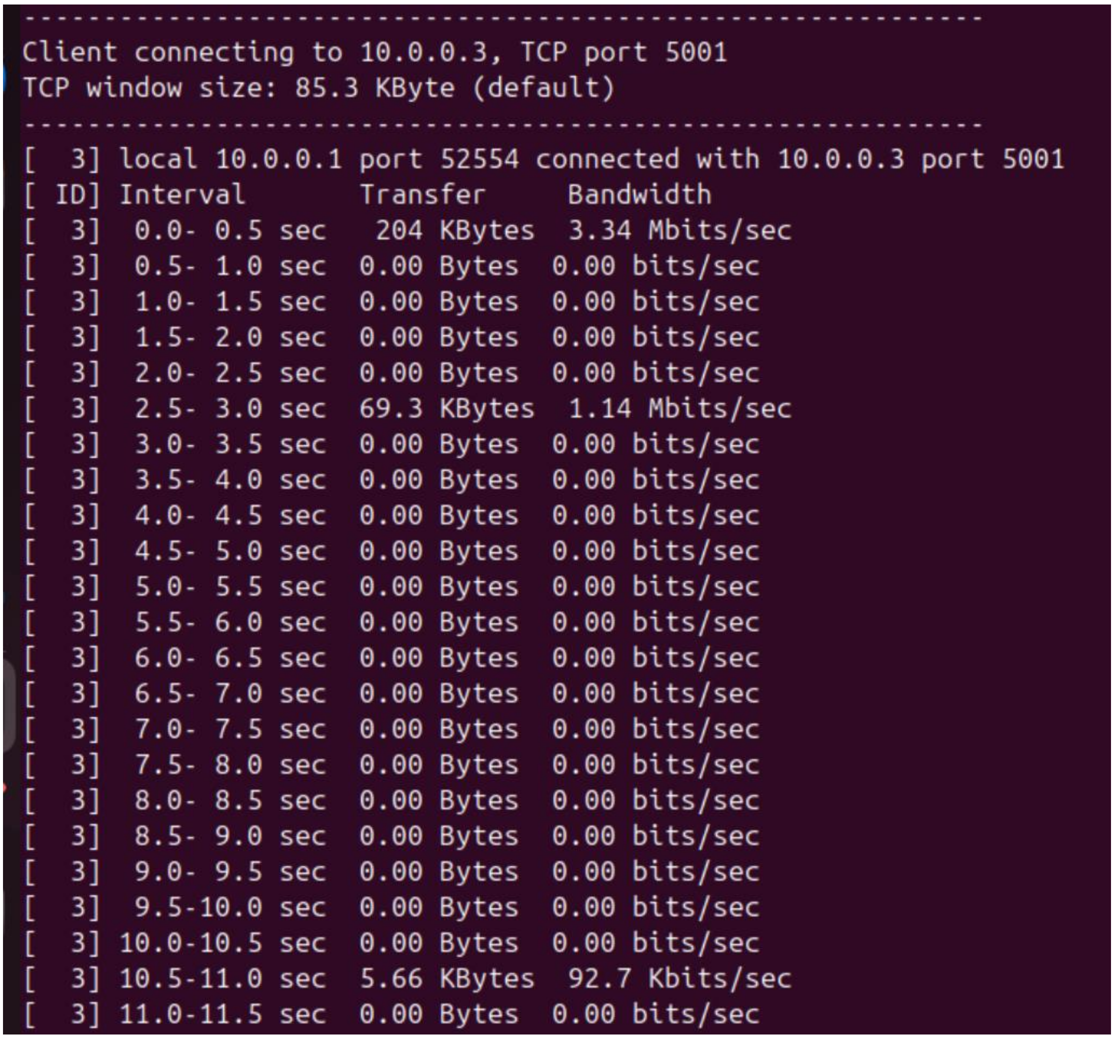
同时，使用 python 提供的线程库做并发，因为两段 tcp 流有时间交集，中间存在并发问题。24 之间的 tcp 流晚 10s，使用 time 函数停 10s 即可。

输出结果如上图所示（loss 选用图中 default 情况 10%）

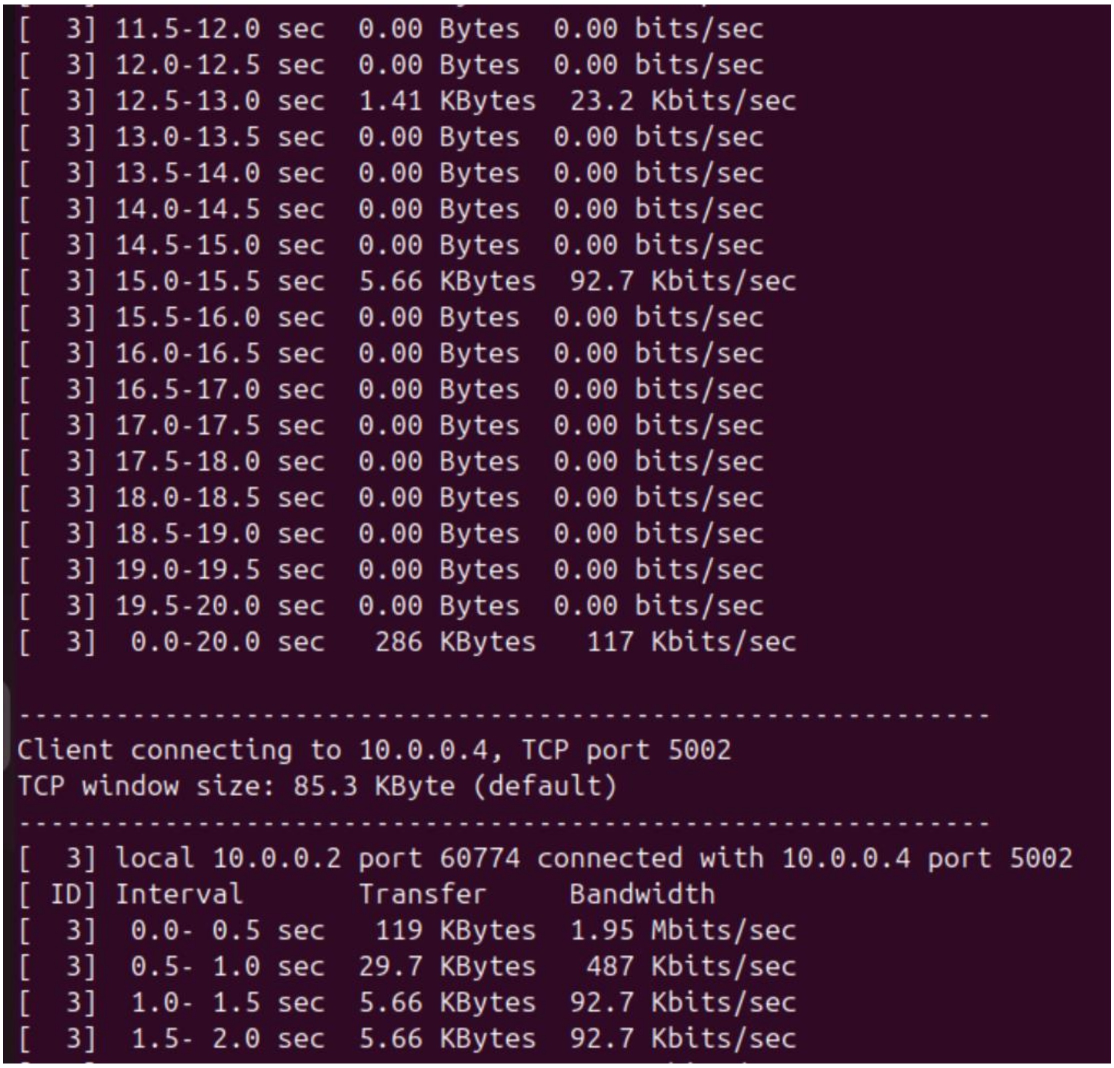
可以看到每 0.5s 所测出的带宽都不太相同，有时（一般是刚开始的时候）会很高，后面还会有很多个 interval 是 0，表面测量过程中网络不稳定，可能是拥塞所导致的。

并且每 0.5s 测量出的 Mbps 都远小于理论值（理论上，13 之间得有 10 左右，24 之间甚至有 20 左右）但是看到平均都是 1 点几。这倒是和上面实验 1 的第二部分只有 1Mbps 左右相似，只要过中间有丢包率的 link，Mbps 就会降低许多。倒应该是和重发、拥塞有关。

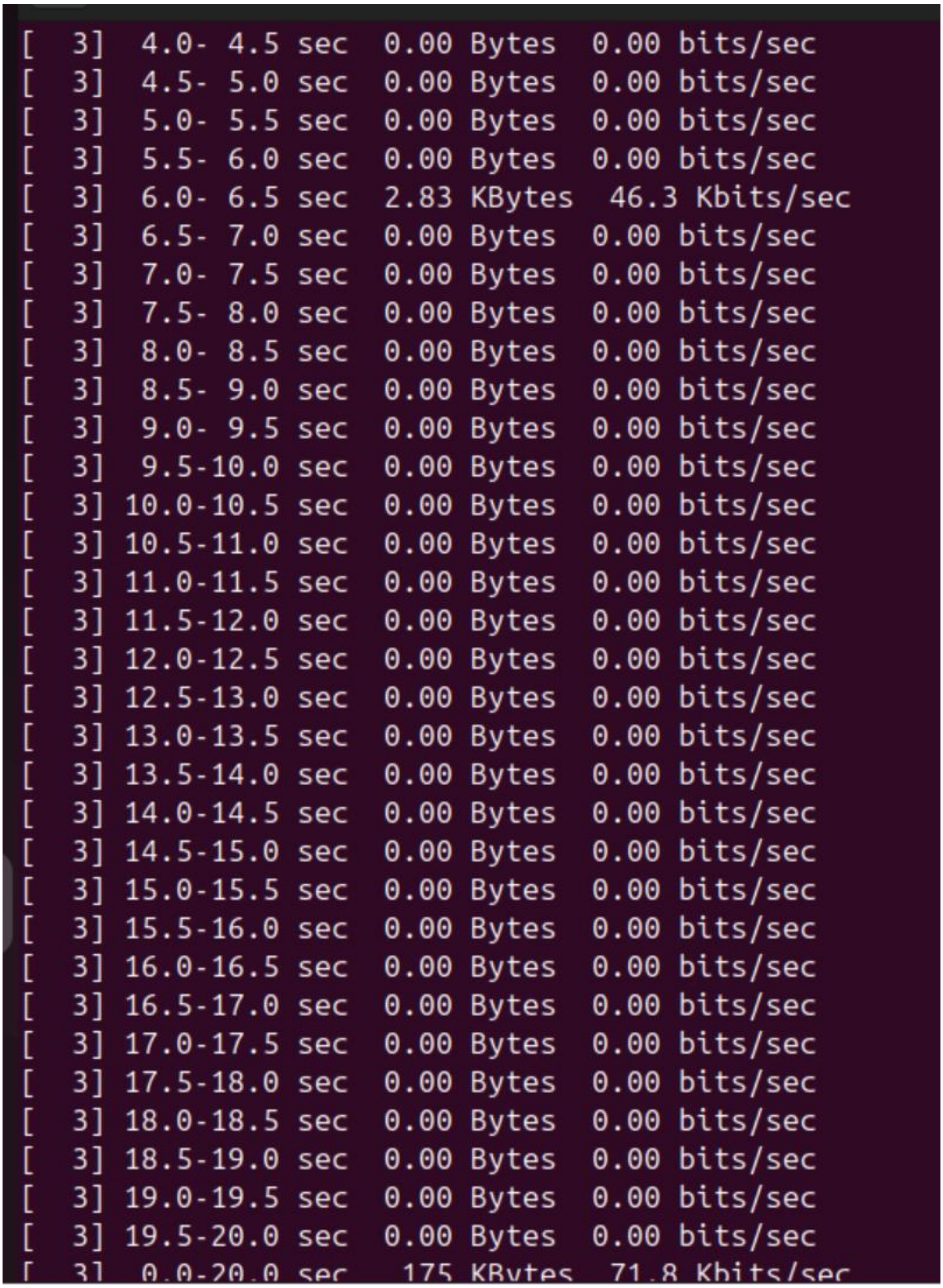
Switch 之间 Loss 设为 30%：



```
------------------------------------------------------------
Client connecting to 10.0.0.3, TCP port 5001
TCP window size: 85.3 KByte (default)
------------------------------------------------------------
[  3] local 10.0.0.1 port 52554 connected with 10.0.0.3 port 5001
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0- 0.5 sec   204 KBytes  3.34 Mbits/sec
[  3]  0.5- 1.0 sec  0.00 Bytes  0.00 bits/sec
[  3]  1.0- 1.5 sec  0.00 Bytes  0.00 bits/sec
[  3]  1.5- 2.0 sec  0.00 Bytes  0.00 bits/sec
[  3]  2.0- 2.5 sec  0.00 Bytes  0.00 bits/sec
[  3]  2.5- 3.0 sec  69.3 KBytes  1.14 Mbits/sec
[  3]  3.0- 3.5 sec  0.00 Bytes  0.00 bits/sec
[  3]  3.5- 4.0 sec  0.00 Bytes  0.00 bits/sec
[  3]  4.0- 4.5 sec  0.00 Bytes  0.00 bits/sec
[  3]  4.5- 5.0 sec  0.00 Bytes  0.00 bits/sec
[  3]  5.0- 5.5 sec  0.00 Bytes  0.00 bits/sec
[  3]  5.5- 6.0 sec  0.00 Bytes  0.00 bits/sec
[  3]  6.0- 6.5 sec  0.00 Bytes  0.00 bits/sec
[  3]  6.5- 7.0 sec  0.00 Bytes  0.00 bits/sec
[  3]  7.0- 7.5 sec  0.00 Bytes  0.00 bits/sec
[  3]  7.5- 8.0 sec  0.00 Bytes  0.00 bits/sec
[  3]  8.0- 8.5 sec  0.00 Bytes  0.00 bits/sec
[  3]  8.5- 9.0 sec  0.00 Bytes  0.00 bits/sec
[  3]  9.0- 9.5 sec  0.00 Bytes  0.00 bits/sec
[  3]  9.5-10.0 sec  0.00 Bytes  0.00 bits/sec
[  3] 10.0-10.5 sec  0.00 Bytes  0.00 bits/sec
[  3] 10.5-11.0 sec  5.66 KBytes  92.7 Kbits/sec
[  3] 11.0-11.5 sec  0.00 Bytes  0.00 bits/sec
```



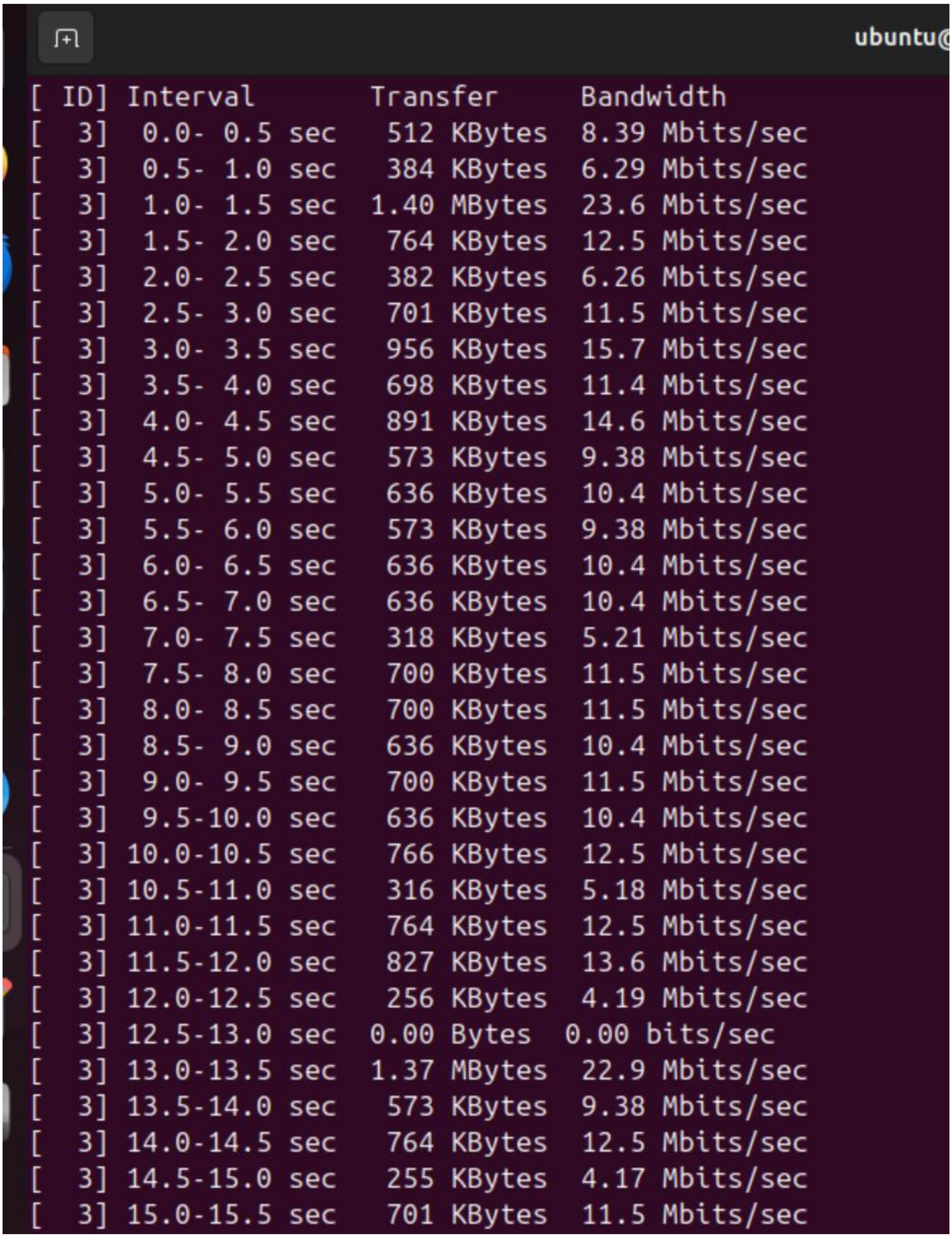
```
[  3] 11.5-12.0 sec  0.00 Bytes  0.00 bits/sec
[  3] 12.0-12.5 sec  0.00 Bytes  0.00 bits/sec
[  3] 12.5-13.0 sec  1.41 KBytes  23.2 Kbits/sec
[  3] 13.0-13.5 sec  0.00 Bytes  0.00 bits/sec
[  3] 13.5-14.0 sec  0.00 Bytes  0.00 bits/sec
[  3] 14.0-14.5 sec  0.00 Bytes  0.00 bits/sec
[  3] 14.5-15.0 sec  0.00 Bytes  0.00 bits/sec
[  3] 15.0-15.5 sec  5.66 KBytes  92.7 Kbits/sec
[  3] 15.5-16.0 sec  0.00 Bytes  0.00 bits/sec
[  3] 16.0-16.5 sec  0.00 Bytes  0.00 bits/sec
[  3] 16.5-17.0 sec  0.00 Bytes  0.00 bits/sec
[  3] 17.0-17.5 sec  0.00 Bytes  0.00 bits/sec
[  3] 17.5-18.0 sec  0.00 Bytes  0.00 bits/sec
[  3] 18.0-18.5 sec  0.00 Bytes  0.00 bits/sec
[  3] 18.5-19.0 sec  0.00 Bytes  0.00 bits/sec
[  3] 19.0-19.5 sec  0.00 Bytes  0.00 bits/sec
[  3] 19.5-20.0 sec  0.00 Bytes  0.00 bits/sec
[  3]  0.0-20.0 sec   286 KBytes   117 Kbits/sec

------------------------------------------------------------
Client connecting to 10.0.0.4, TCP port 5002
TCP window size: 85.3 KByte (default)
------------------------------------------------------------
[  3] local 10.0.0.2 port 60774 connected with 10.0.0.4 port 5002
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0- 0.5 sec   119 KBytes  1.95 Mbits/sec
[  3]  0.5- 1.0 sec  29.7 KBytes   487 Kbits/sec
[  3]  1.0- 1.5 sec  5.66 KBytes  92.7 Kbits/sec
[  3]  1.5- 2.0 sec  5.66 KBytes  92.7 Kbits/sec
```



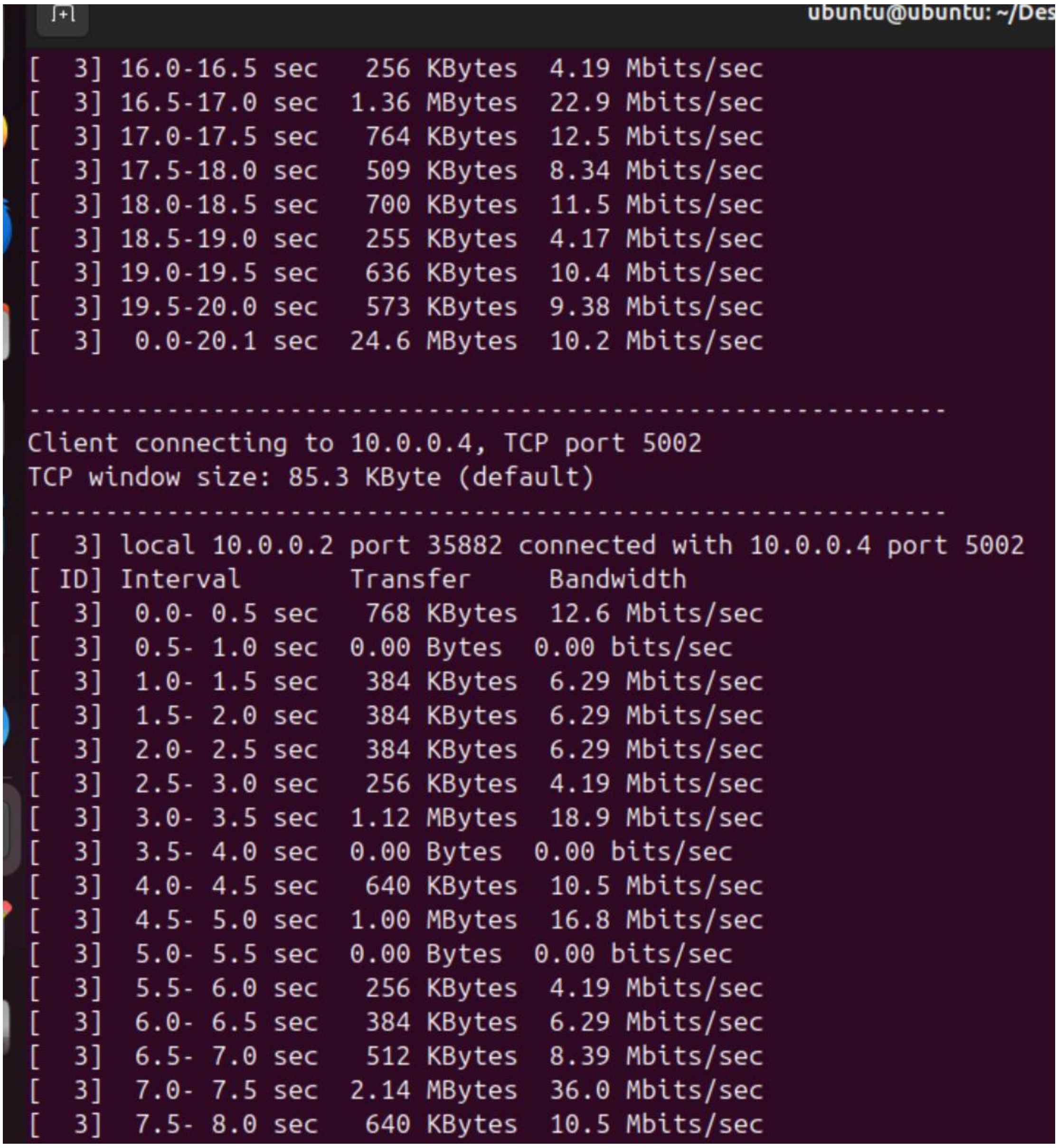
```
[  3]  4.0- 4.5 sec  0.00 Bytes  0.00 bits/sec
[  3]  4.5- 5.0 sec  0.00 Bytes  0.00 bits/sec
[  3]  5.0- 5.5 sec  0.00 Bytes  0.00 bits/sec
[  3]  5.5- 6.0 sec  0.00 Bytes  0.00 bits/sec
[  3]  6.0- 6.5 sec  2.83 KBytes  46.3 Kbits/sec
[  3]  6.5- 7.0 sec  0.00 Bytes  0.00 bits/sec
[  3]  7.0- 7.5 sec  0.00 Bytes  0.00 bits/sec
[  3]  7.5- 8.0 sec  0.00 Bytes  0.00 bits/sec
[  3]  8.0- 8.5 sec  0.00 Bytes  0.00 bits/sec
[  3]  8.5- 9.0 sec  0.00 Bytes  0.00 bits/sec
[  3]  9.0- 9.5 sec  0.00 Bytes  0.00 bits/sec
[  3]  9.5-10.0 sec  0.00 Bytes  0.00 bits/sec
[  3] 10.0-10.5 sec  0.00 Bytes  0.00 bits/sec
[  3] 10.5-11.0 sec  0.00 Bytes  0.00 bits/sec
[  3] 11.0-11.5 sec  0.00 Bytes  0.00 bits/sec
[  3] 11.5-12.0 sec  0.00 Bytes  0.00 bits/sec
[  3] 12.0-12.5 sec  0.00 Bytes  0.00 bits/sec
[  3] 12.5-13.0 sec  0.00 Bytes  0.00 bits/sec
[  3] 13.0-13.5 sec  0.00 Bytes  0.00 bits/sec
[  3] 13.5-14.0 sec  0.00 Bytes  0.00 bits/sec
[  3] 14.0-14.5 sec  0.00 Bytes  0.00 bits/sec
[  3] 14.5-15.0 sec  0.00 Bytes  0.00 bits/sec
[  3] 15.0-15.5 sec  0.00 Bytes  0.00 bits/sec
[  3] 15.5-16.0 sec  0.00 Bytes  0.00 bits/sec
[  3] 16.0-16.5 sec  0.00 Bytes  0.00 bits/sec
[  3] 16.5-17.0 sec  0.00 Bytes  0.00 bits/sec
[  3] 17.0-17.5 sec  0.00 Bytes  0.00 bits/sec
[  3] 17.5-18.0 sec  0.00 Bytes  0.00 bits/sec
[  3] 18.0-18.5 sec  0.00 Bytes  0.00 bits/sec
[  3] 18.5-19.0 sec  0.00 Bytes  0.00 bits/sec
[  3] 19.0-19.5 sec  0.00 Bytes  0.00 bits/sec
[  3] 19.5-20.0 sec  0.00 Bytes  0.00 bits/sec
[  3]  0.0-20.0 sec   175 KBytes  71.8 Kbits/sec
```

可以看到带宽更低了，大部分时候都到 0 了，基本上确定就是由于丢包太多，重发的太多导致拥塞。使得带宽非常低。

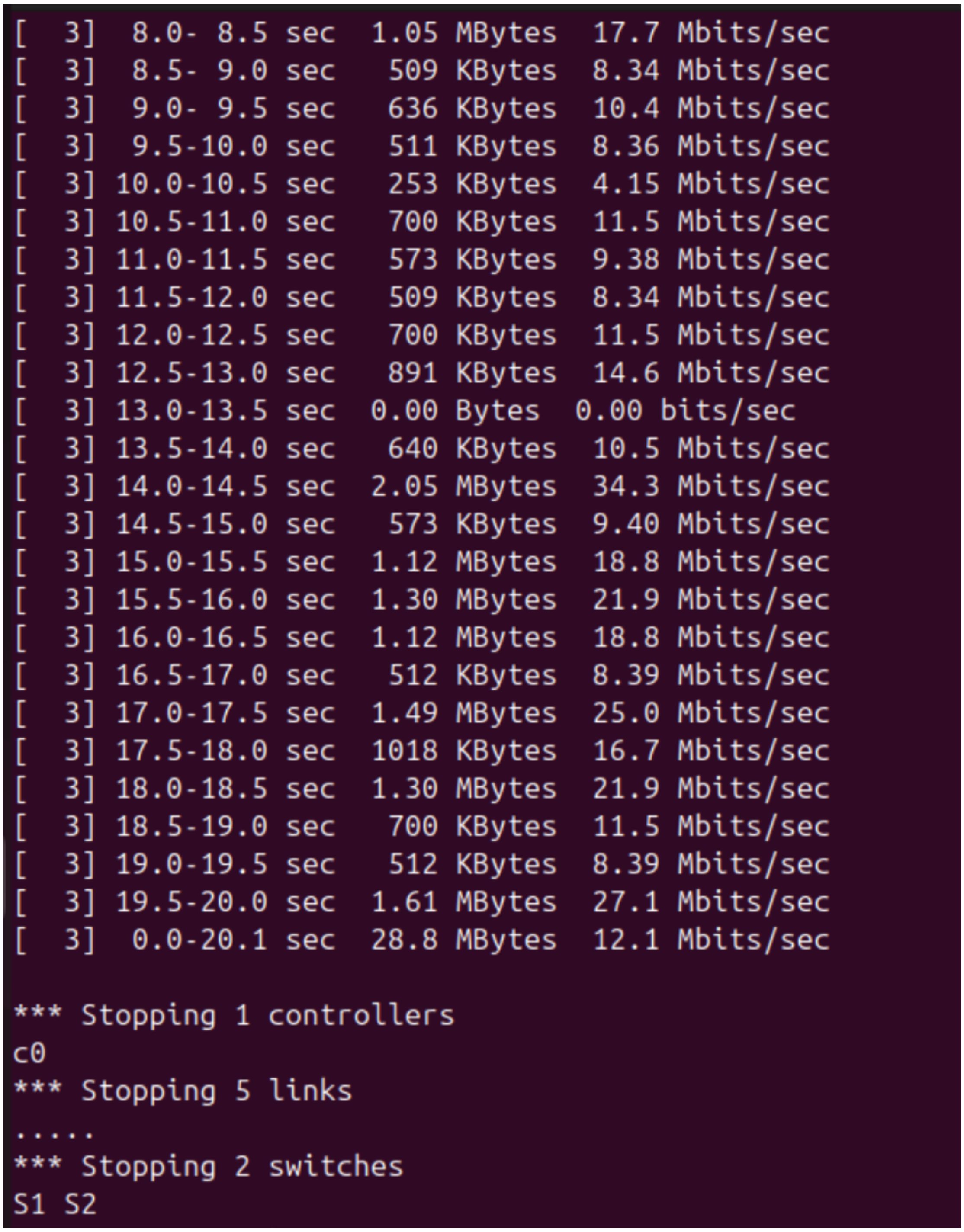
Switch 之间 Loss 设为 0%：



```
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0- 0.5 sec   512 KBytes  8.39 Mbits/sec
[  3]  0.5- 1.0 sec   384 KBytes  6.29 Mbits/sec
[  3]  1.0- 1.5 sec  1.40 MBytes  23.6 Mbits/sec
[  3]  1.5- 2.0 sec   764 KBytes  12.5 Mbits/sec
[  3]  2.0- 2.5 sec   382 KBytes  6.26 Mbits/sec
[  3]  2.5- 3.0 sec   701 KBytes  11.5 Mbits/sec
[  3]  3.0- 3.5 sec   956 KBytes  15.7 Mbits/sec
[  3]  3.5- 4.0 sec   698 KBytes  11.4 Mbits/sec
[  3]  4.0- 4.5 sec   891 KBytes  14.6 Mbits/sec
[  3]  4.5- 5.0 sec   573 KBytes  9.38 Mbits/sec
[  3]  5.0- 5.5 sec   636 KBytes  10.4 Mbits/sec
[  3]  5.5- 6.0 sec   573 KBytes  9.38 Mbits/sec
[  3]  6.0- 6.5 sec   636 KBytes  10.4 Mbits/sec
[  3]  6.5- 7.0 sec   636 KBytes  10.4 Mbits/sec
[  3]  7.0- 7.5 sec   318 KBytes  5.21 Mbits/sec
[  3]  7.5- 8.0 sec   700 KBytes  11.5 Mbits/sec
[  3]  8.0- 8.5 sec   700 KBytes  11.5 Mbits/sec
[  3]  8.5- 9.0 sec   636 KBytes  10.4 Mbits/sec
[  3]  9.0- 9.5 sec   700 KBytes  11.5 Mbits/sec
[  3]  9.5-10.0 sec   636 KBytes  10.4 Mbits/sec
[  3] 10.0-10.5 sec   766 KBytes  12.5 Mbits/sec
[  3] 10.5-11.0 sec   316 KBytes  5.18 Mbits/sec
[  3] 11.0-11.5 sec   764 KBytes  12.5 Mbits/sec
[  3] 11.5-12.0 sec   827 KBytes  13.6 Mbits/sec
[  3] 12.0-12.5 sec   256 KBytes  4.19 Mbits/sec
[  3] 12.5-13.0 sec  0.00 Bytes  0.00 bits/sec
[  3] 13.0-13.5 sec  1.37 MBytes  22.9 Mbits/sec
[  3] 13.5-14.0 sec   573 KBytes  9.38 Mbits/sec
[  3] 14.0-14.5 sec   764 KBytes  12.5 Mbits/sec
[  3] 14.5-15.0 sec   255 KBytes  4.17 Mbits/sec
[  3] 15.0-15.5 sec   701 KBytes  11.5 Mbits/sec
```



```
[  3] 16.0-16.5 sec   256 KBytes  4.19 Mbits/sec
[  3] 16.5-17.0 sec  1.36 MBytes  22.9 Mbits/sec
[  3] 17.0-17.5 sec   764 KBytes  12.5 Mbits/sec
[  3] 17.5-18.0 sec   509 KBytes  8.34 Mbits/sec
[  3] 18.0-18.5 sec   700 KBytes  11.5 Mbits/sec
[  3] 18.5-19.0 sec   255 KBytes  4.17 Mbits/sec
[  3] 19.0-19.5 sec   636 KBytes  10.4 Mbits/sec
[  3] 19.5-20.0 sec   573 KBytes  9.38 Mbits/sec
[  3]  0.0-20.1 sec  24.6 MBytes  10.2 Mbits/sec

------------------------------------------------------------
Client connecting to 10.0.0.4, TCP port 5002
TCP window size: 85.3 KByte (default)
------------------------------------------------------------
[  3] local 10.0.0.2 port 35882 connected with 10.0.0.4 port 5002
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0- 0.5 sec   768 KBytes  12.6 Mbits/sec
[  3]  0.5- 1.0 sec  0.00 Bytes  0.00 bits/sec
[  3]  1.0- 1.5 sec   384 KBytes  6.29 Mbits/sec
[  3]  1.5- 2.0 sec   384 KBytes  6.29 Mbits/sec
[  3]  2.0- 2.5 sec   384 KBytes  6.29 Mbits/sec
[  3]  2.5- 3.0 sec   256 KBytes  4.19 Mbits/sec
[  3]  3.0- 3.5 sec  1.12 MBytes  18.9 Mbits/sec
[  3]  3.5- 4.0 sec  0.00 Bytes  0.00 bits/sec
[  3]  4.0- 4.5 sec   640 KBytes  10.5 Mbits/sec
[  3]  4.5- 5.0 sec  1.00 MBytes  16.8 Mbits/sec
[  3]  5.0- 5.5 sec  0.00 Bytes  0.00 bits/sec
[  3]  5.5- 6.0 sec   256 KBytes  4.19 Mbits/sec
[  3]  6.0- 6.5 sec   384 KBytes  6.29 Mbits/sec
[  3]  6.5- 7.0 sec   512 KBytes  8.39 Mbits/sec
[  3]  7.0- 7.5 sec  2.14 MBytes  36.0 Mbits/sec
[  3]  7.5- 8.0 sec   640 KBytes  10.5 Mbits/sec
```



```
[  3]  8.0- 8.5 sec  1.05 MBytes  17.7 Mbits/sec
[  3]  8.5- 9.0 sec   509 KBytes  8.34 Mbits/sec
[  3]  9.0- 9.5 sec   636 KBytes  10.4 Mbits/sec
[  3]  9.5-10.0 sec   511 KBytes  8.36 Mbits/sec
[  3] 10.0-10.5 sec   253 KBytes  4.15 Mbits/sec
[  3] 10.5-11.0 sec   700 KBytes  11.5 Mbits/sec
[  3] 11.0-11.5 sec   573 KBytes  9.38 Mbits/sec
[  3] 11.5-12.0 sec   509 KBytes  8.34 Mbits/sec
[  3] 12.0-12.5 sec   700 KBytes  11.5 Mbits/sec
[  3] 12.5-13.0 sec   891 KBytes  14.6 Mbits/sec
[  3] 13.0-13.5 sec  0.00 Bytes  0.00 bits/sec
[  3] 13.5-14.0 sec   640 KBytes  10.5 Mbits/sec
[  3] 14.0-14.5 sec  2.05 MBytes  34.3 Mbits/sec
[  3] 14.5-15.0 sec   573 KBytes  9.40 Mbits/sec
[  3] 15.0-15.5 sec  1.12 MBytes  18.8 Mbits/sec
[  3] 15.5-16.0 sec  1.30 MBytes  21.9 Mbits/sec
[  3] 16.0-16.5 sec  1.12 MBytes  18.8 Mbits/sec
[  3] 16.5-17.0 sec   512 KBytes  8.39 Mbits/sec
[  3] 17.0-17.5 sec  1.49 MBytes  25.0 Mbits/sec
[  3] 17.5-18.0 sec  1018 KBytes  16.7 Mbits/sec
[  3] 18.0-18.5 sec  1.30 MBytes  21.9 Mbits/sec
[  3] 18.5-19.0 sec   700 KBytes  11.5 Mbits/sec
[  3] 19.0-19.5 sec   512 KBytes  8.39 Mbits/sec
[  3] 19.5-20.0 sec  1.61 MBytes  27.1 Mbits/sec
[  3]  0.0-20.1 sec  28.8 MBytes  12.1 Mbits/sec

*** Stopping 1 controllers
c0
*** Stopping 5 links
.....
*** Stopping 2 switches
S1 S2
```

丢包率设置为 0 后，一切都正常起来了，第一组理论上在 10Mbps 左右，差不多，第二组可达到 20Mbps。