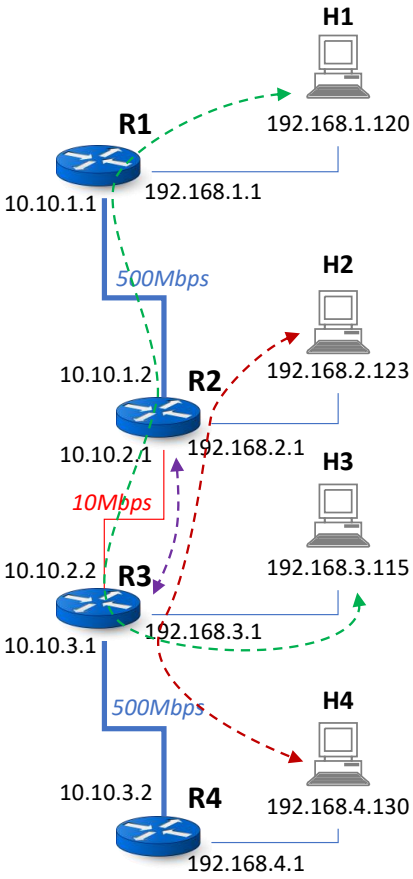
Thiết kế & triển khai mạng IP

Bài thực hành: Quality of Service

Mục lục

1 Chuẩn bị môi trường

Sử dụng các kiến thức của bài trước, dựng môi trường mạng ảo với các kết nối mặc định có tốc độ cao (500Mbps) và 1 kết nối tốc độ thấp (10Mbps). Các luồng dữ liệu cạnh tranh trên kết nối tốc độ thấp này.



1.1 Thiết lập môi trường kết nối mạng

1. Cấu hình R1:

```
R1:~$ sudo nano /etc/netplan/01-network-manager-all.yaml

network:
  ethernets:
    enp0s3:
      dhcp4: true
    enp0s8:
      addresses:
        - 192.168.156.11/24
    enp0s9:
      addresses:
        - 10.10.1.1/24
      routes:
        - to: 192.168.2.0/24
          via: 10.10.1.2
        - to: 192.168.3.0/24
          via: 10.10.1.2
        - to: 192.168.4.0/24
          via: 10.10.1.2
    enp0s10:
      addresses:
        - 192.168.1.1/24

R1:~$ sudo netplan apply
R1:~$ route -n

Kernel IP routing table
```

```
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
10.10.1.0       0.0.0.0         255.255.255.0   U     0      0        0 enp0s9
10.10.2.0       10.10.1.2       255.255.255.0   UG    0      0        0 enp0s9
192.168.1.0     0.0.0.0         255.255.255.0   U     0      0        0 enp0s10
192.168.2.0     10.10.1.2       255.255.255.0   UG    0      0        0 enp0s9
192.168.3.0     10.10.1.2       255.255.255.0   UG    0      0        0 enp0s9
192.168.4.0     10.10.1.2       255.255.255.0   UG    0      0        0 enp0s9
```

2. Cấu hình H1:

```
R1:~$ sudo nano /etc/netplan/01-network-manager-all.yaml


network:
  ethernets:
    enp0s8:
      addresses:
        - 192.168.156.120/24
    enp0s9:
      addresses:
        - 192.168.1.120/24
      gateway4: 192.168.1.1


H1:~$ sudo netplan apply
H1:~$ route -n

Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         192.168.1.1     0.0.0.0         UG    0      0        0 enp0s9
192.168.1.0     0.0.0.0         255.255.255.0   U     0      0        0 enp0s9
```

3. Cấu hình R2:

```
R2:~$ sudo nano /etc/netplan/01-network-manager-all.yaml

network:
  ethernets:
    enp0s3:
      addresses:
        - 192.168.2.1/24
    enp0s9:
      addresses:
        - 10.10.1.2/24
      routes:
        - to: 192.168.1.0/24
          via: 10.10.1.1
    enp0s10:
      addresses:
        - 10.10.2.1/24
      routes:
        - to: 192.168.3.0/24
          via: 10.10.2.2
        - to: 192.168.4.0/24
          via: 10.10.2.2

R2:~$ sudo netplan apply
R2:~$ route -n

Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
10.10.1.0       0.0.0.0         255.255.255.0   U     0      0        0 enp0s9
10.10.2.0       0.0.0.0         255.255.255.0   U     0      0        0 enp0s10
192.168.1.0     10.10.1.1       255.255.255.0   UG    0      0        0 enp0s9
192.168.2.0     0.0.0.0         255.255.255.0   U     0      0        0 enp0s3
192.168.3.0     10.10.2.2       255.255.255.0   UG    0      0        0 enp0s10
192.168.4.0     10.10.2.2       255.255.255.0   UG    0      0        0 enp0s10
```

4. Tương tự, cấu hình các router khác và các host khác. Kiểm tra các host kết nối được với nhau qua các router:

```
H1:~$ ping 192.168.4.130
```

```
PING 192.168.4.130 (192.168.4.130) 56(84) bytes of data.
64 bytes from 192.168.4.130: icmp_seq=1 ttl=60 time=3.48 ms
64 bytes from 192.168.4.130: icmp_seq=2 ttl=60 time=3.19 ms
^C
--- 192.168.4.130 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1002ms
rtt min/avg/max/mdev = 3.185/3.331/3.478/0.146 ms
```

```
H1:~$ tracepath -n 192.168.4.130
```

```
 1?: [LOCALHOST]                      pmtu 1500
 1:  192.168.1.1                                           0.633ms
 1:  192.168.1.1                                           0.616ms
 2:  10.10.1.2                                             1.270ms
 3:  10.10.2.2                                             2.031ms
 4:  no reply
 5:  192.168.4.130                                         3.352ms reached
     Resume: pmtu 1500 hops 5 back 5
```

1.2 Thiết lập tốc độ kết nối mạng R2 – R3

5. Sử dụng tool VboxManage đi kèm Virtualbox để thiết lập tốc độ cho các link kết nối giữa router R2 và R3. Đầu tiên là liệt kê các máy ảo trong hệ thống. Khi thực hiện thay đổi cấu hình băng thông này cần shutdown router R2:

```
$ VBoxManage list vms
```

```
"R1 (QoS)" {7e034d5a-c892-4107-9383-7db2fea07b37}
"R2 (QoS)" {05d73c23-84bc-49ef-8b83-396cbdde5bdb}
"R3 (QoS)" {597d5326-bdd8-48a3-b2a5-b25b93867cea}
"R4 (QoS)" {6001cf5b-4425-43a3-925b-09488fa9951d}
```

6. Xem chi tiết thông tin các kết nối mạng của máy ảo R2. Kết nối R2-R3 năm ở NIC 4 và không hạn chế băng thông (bandwidth group: none):

```
$ VBoxManage showvminfo "R2 (QoS)" | grep NIC
```

```
NIC 1:                       MAC: 080027870C4C, Attachment: Internal Network 'lan02', Cable
connected: on, Trace: off (file: none), Type: 82540EM, Reported speed: 0 Mbps, Boot priority: 0,
Promisc Policy: deny, Bandwidth group: none
NIC 2:                       MAC: 08002766D4EC, Attachment: Host-only Interface 'vboxnet0', Cable
connected: on, Trace: off (file: none), Type: 82540EM, Reported speed: 0 Mbps, Boot priority: 0,
Promisc Policy: deny, Bandwidth group: none
NIC 3:                       MAC: 08002761DE17, Attachment: Internal Network 'serial1', Cable
connected: on, Trace: off (file: none), Type: 82540EM, Reported speed: 0 Mbps, Boot priority: 0,
Promisc Policy: deny, Bandwidth group: none
NIC 4:                       MAC: 080027F9AF90, Attachment: Internal Network 'serial2', Cable
connected: on, Trace: off (file: none), Type: 82540EM, Reported speed: 0 Mbps, Boot priority: 0,
Promisc Policy: deny, Bandwidth group: none
NIC 5:                       disabled
NIC 6:                       disabled
NIC 7:                       disabled
NIC 8:                       disabled
```

7. Thiết lập hạn chế băng thông của NIC 4 là 10Mbps:

```
$ VBoxManage bandwidthctl "R2 (QoS)" add Limit10m --type network --limit 10m
$ VBoxManage modifyvm "R2 (QoS)" --nicbandwidthgroup4 Limit10m

$ VBoxManage showvminfo "R2 (QoS)" | grep NIC
```

```
NIC 1:                       MAC: 080027870C4C, Attachment: Internal Network 'lan02', Cable
connected: on, Trace: off (file: none), Type: 82540EM, Reported speed: 0 Mbps, Boot priority: 0,
Promisc Policy: deny, Bandwidth group: none
NIC 2:                       MAC: 08002766D4EC, Attachment: Host-only Interface 'vboxnet0', Cable
connected: on, Trace: off (file: none), Type: 82540EM, Reported speed: 0 Mbps, Boot priority: 0,
Promisc Policy: deny, Bandwidth group: none
```

```
NIC 3:                       MAC: 08002761DE17, Attachment: Internal Network 'serial1', Cable
connected: on, Trace: off (file: none), Type: 82540EM, Reported speed: 0 Mbps, Boot priority: 0,
Promisc Policy: deny, Bandwidth group: none
NIC 4:                       MAC: 080027F9AF90, Attachment: Internal Network 'serial2', Cable
connected: on, Trace: off (file: none), Type: 82540EM, Reported speed: 0 Mbps, Boot priority: 0,
Promisc Policy: deny, Bandwidth group: Limit10m
NIC 5:                       disabled
NIC 6:                       disabled
NIC 7:                       disabled
NIC 8:                       disabled
```

8. Để hủy hạn chế băng thông trên một kết nối mạng thì thiết lập với group = none:

```
VBoxManage modifyvm "R2 (QoS)" –nicbandwidthgroup4 none
```

1.3 Kiểm tra ảnh hưởng tốc độ giữa các dòng dữ liệu cạnh tranh

1. Chạy *iperf* trên H4 ở chế độ server (nghe) và trên H3 ở chế độ client (truyền). Thấy tốc độ truyền dữ liệu giữ H3 và H4 là khoảng 500Mbps

```
H4:~$ iperf -s -i 1

------------------------------------------------------------
Server listening on TCP port 5001
TCP window size:  128 KByte (default)
------------------------------------------------------------
[  4] local 192.168.4.130 port 5001 connected with 192.168.3.115 port 51936
[ ID] Interval       Transfer     Bandwidth
[  4]  0.0- 1.0 sec  53.4 MBytes   448 Mbits/sec
[  4]  1.0- 2.0 sec  61.2 MBytes   513 Mbits/sec
[  4]  2.0- 3.0 sec  68.5 MBytes   574 Mbits/sec
[  4]  3.0- 4.0 sec  66.1 MBytes   554 Mbits/sec
[  4]  4.0- 5.0 sec  49.7 MBytes   417 Mbits/sec


H3:~$ iperf -c 192.168.4.130 -i 1 -t 3

------------------------------------------------------------
Client connecting to 192.168.4.130, TCP port 5001
TCP window size:  196 KByte (default)
------------------------------------------------------------
[  3] local 192.168.3.115 port 51940 connected with 192.168.4.130 port 5001
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0- 1.0 sec  69.6 MBytes   584 Mbits/sec
[  3]  1.0- 2.0 sec  50.4 MBytes   423 Mbits/sec
[  3]  2.0- 3.0 sec  61.0 MBytes   511 Mbits/sec
[  3]  0.0- 3.0 sec   181 MBytes   505 Mbits/sec
```

2. Chạy *iperf* trên H2 ở chế độ client (truyền). Thấy tốc độ H4 nhận dữ liệu từ H2 và H4 là khoảng dưới 10Mbps (do dòng dữ liệu này đi qua kết nối R2-R3 chỉ có tốc độ 10Mbps). Trên H2 ban đầu tốc độ truyền cao hơn 10Mbps rồi giảm dần về tương đương với tốc độ nhận dữ liệu trên H4 (nhớ lại cơ chế flow control của giao thức TCP với phương pháp điều khiển kích thước cửa sổ trượt sliding window):

```
H4:~$ iperf -s -i 1

------------------------------------------------------------
Server listening on TCP port 5001
TCP window size:  128 KByte (default)
------------------------------------------------------------
[  4] local 192.168.4.130 port 5001 connected with 192.168.2.123 port 51804
[ ID] Interval       Transfer     Bandwidth
[  4]  0.0- 1.0 sec  1.19 MBytes  9.96 Mbits/sec
[  4]  1.0- 2.0 sec  1.18 MBytes  9.92 Mbits/sec
[  4]  2.0- 3.0 sec  1.17 MBytes  9.82 Mbits/sec


H2:~$ iperf -c 192.168.4.130 -i 1 -t 30

------------------------------------------------------------
```

```
Client connecting to 192.168.4.130, TCP port 5001
TCP window size:  348 KByte (default)
------------------------------------------------------------
[  3] local 192.168.2.123 port 51806 connected with 192.168.4.130 port 5001
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0- 1.0 sec  1.88 MBytes  15.7 Mbits/sec
[  3]  1.0- 2.0 sec  1.50 MBytes  12.6 Mbits/sec
[  3]  2.0- 3.0 sec  1.48 MBytes  12.4 Mbits/sec
[  3]  3.0- 4.0 sec  1.12 MBytes  9.44 Mbits/sec
[  3]  4.0- 5.0 sec  1.11 MBytes  9.33 Mbits/sec
[  3]  5.0- 6.0 sec  1.12 MBytes  9.38 Mbits/sec
[  3]  6.0- 7.0 sec  1.12 MBytes  9.44 Mbits/sec
[  3]  7.0- 8.0 sec  1012 KBytes  8.29 Mbits/sec
```

3. Nếu thiết lập dòng dữ liệu iperf giữa H1 và H3 như trên thì thấy tốc độ cũng đạt mức ổn định ở khoảng 9Mbps giống như dòng H2-H4 bên trên. Trong khi dòng H1-H3 đang được thực hiện thì thiết lập dòng *iperf* từ H2 đến H4. Quan sát tốc độ dòng này thấy chỉ còn 1Mbps, thậm chí là chỉ còn 500Kbps:

```
H1:~$ iperf -c 192.168.3.115 -i 1 -t 300

------------------------------------------------------------
Client connecting to 192.168.3.115, TCP port 5001
TCP window size:  357 KByte (default)
------------------------------------------------------------
[  3] local 192.168.1.120 port 54730 connected with 192.168.3.115 port 5001
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0- 1.0 sec  1.88 MBytes  15.7 Mbits/sec
[  3]  1.0- 2.0 sec  1.38 MBytes  11.5 Mbits/sec
[  3]  2.0- 3.0 sec  1.48 MBytes  12.4 Mbits/sec
[  3]  3.0- 4.0 sec  1.06 MBytes  8.89 Mbits/sec
[  3]  4.0- 5.0 sec  1018 KBytes  8.34 Mbits/sec
[  3]  5.0- 6.0 sec  1.05 MBytes  8.84 Mbits/sec
[  3]  6.0- 7.0 sec  1.06 MBytes  8.86 Mbits/sec
[  3]  7.0- 8.0 sec  1.18 MBytes  9.93 Mbits/sec


H2:~$ iperf -c 192.168.4.130 -i 1 -t 10

------------------------------------------------------------
Client connecting to 192.168.4.130, TCP port 5001
TCP window size: 85.0 KByte (default)
------------------------------------------------------------
[  3] local 192.168.2.123 port 51824 connected with 192.168.4.130 port 5001
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0- 1.0 sec   229 KBytes  1.88 Mbits/sec
[  3]  1.0- 2.0 sec   156 KBytes  1.27 Mbits/sec
[  3]  2.0- 3.0 sec  0.00 Bytes  0.00 bits/sec
[  3]  3.0- 4.0 sec  63.6 KBytes   521 Kbits/sec
[  3]  4.0- 5.0 sec  0.00 Bytes  0.00 bits/sec
[  3]  5.0- 6.0 sec  0.00 Bytes  0.00 bits/sec
[  3]  6.0- 7.0 sec  63.6 KBytes   521 Kbits/sec
[  3]  7.0- 8.0 sec  63.6 KBytes   521 Kbits/sec
[  3]  8.0- 9.0 sec  0.00 Bytes  0.00 bits/sec
[  3]  9.0-10.0 sec  0.00 Bytes  0.00 bits/sec
[  3]  0.0-10.4 sec   576 KBytes   452 Kbits/sec
```

4. Thực nghiệm tốc độ truyền dữ liệu bị ảnh hưởng lẫn nhau càng rõ hơn khi thiết lập trên các dòng truyền dữ liệu cạnh tranh qua R2-R3. Vị dụ nếu thêm dòng *iperf* R2-R3 thì dòng H1-H4 chỉ còn trên dưới 100Kbps.

```
$ iperf -c 10.10.2.2 -i 1 -t 300

------------------------------------------------------------
Client connecting to 10.10.2.2, TCP port 5001
TCP window size:  348 KByte (default)
------------------------------------------------------------
[  3] local 10.10.2.1 port 53472 connected with 10.10.2.2 port 5001
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0- 1.0 sec  1.62 MBytes  13.6 Mbits/sec
```

```
[  3]  1.0- 2.0 sec  1.25 MBytes  10.5 Mbits/sec
[  3]  2.0- 3.0 sec  1.38 MBytes  11.5 Mbits/sec
[  3]  3.0- 4.0 sec  1.12 MBytes  9.44 Mbits/sec
[  3]  4.0- 5.0 sec  1.38 MBytes  11.5 Mbits/sec
[  3]  5.0- 6.0 sec  1.12 MBytes  9.44 Mbits/sec
[  3]  6.0- 7.0 sec  1.12 MBytes  9.44 Mbits/sec
[  3]  7.0- 8.0 sec   617 KBytes  5.06 Mbits/sec
[  3]  8.0- 9.0 sec  63.6 KBytes   521 Kbits/sec
[  3]  9.0-10.0 sec  63.6 KBytes   521 Kbits/sec
```

```
H4:~$ iperf -s -i 1

------------------------------------------------------------
Server listening on TCP port 5001
TCP window size:  128 KByte (default)
------------------------------------------------------------
[  4] local 192.168.4.130 port 5001 connected with 192.168.2.123 port 51828
[ ID] Interval       Transfer     Bandwidth
[  4]  0.0- 1.0 sec  84.8 KBytes   695 Kbits/sec
[  4]  1.0- 2.0 sec  39.6 KBytes   324 Kbits/sec
[  4]  2.0- 3.0 sec  11.3 KBytes  92.7 Kbits/sec
[  4]  3.0- 4.0 sec  11.3 KBytes  92.7 Kbits/sec
[  4]  4.0- 5.0 sec  19.8 KBytes   162 Kbits/sec
[  4]  5.0- 6.0 sec  12.7 KBytes   104 Kbits/sec
```

1.4 Kiểm tra ảnh hưởng mất gói tin giữa các dòng dữ liệu cạnh tranh

5. Thực hiện kịch bản như trên nhưng gửi dòng dữ liệu UDP (để xem độ mất mát gói tin). Khi không có dòng dữ liệu cạnh tranh nào khác, dòng UDP H2 – H4 có độ mất mát gói tin bằng 0:

```
H2:~$ iperf -c 192.168.4.130 -i 1 -t 5 -u

------------------------------------------------------------
Client connecting to 192.168.4.130, UDP port 5001
Sending 1470 byte datagrams, IPG target: 11215.21 us (kalman adjust)
UDP buffer size:  208 KByte (default)
------------------------------------------------------------
[  3] local 192.168.2.123 port 46671 connected with 192.168.4.130 port 5001
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0- 1.0 sec   131 KBytes  1.07 Mbits/sec
[  3]  1.0- 2.0 sec   128 KBytes  1.05 Mbits/sec
[  3]  2.0- 3.0 sec   128 KBytes  1.05 Mbits/sec
[  3]  3.0- 4.0 sec   128 KBytes  1.05 Mbits/sec
[  3]  0.0- 5.0 sec   640 KBytes  1.05 Mbits/sec
[  3] Sent 446 datagrams
[  3] Server Report:
[  3]  0.0- 5.0 sec   640 KBytes  1.05 Mbits/sec   0.053 ms    0/  446 (0%)
```

```
H4:~$ iperf -su -i 1

------------------------------------------------------------
Server listening on UDP port 5001
Receiving 1470 byte datagrams
UDP buffer size:  208 KByte (default)
------------------------------------------------------------
[  3] local 192.168.4.130 port 5001 connected with 192.168.2.123 port 46671
[ ID] Interval       Transfer     Bandwidth        Jitter   Lost/Total Datagrams
[  3]  0.0- 1.0 sec   129 KBytes  1.06 Mbits/sec   0.051 ms    0/   90 (0%)
[  3]  1.0- 2.0 sec   128 KBytes  1.05 Mbits/sec   0.040 ms    0/   89 (0%)
[  3]  2.0- 3.0 sec   128 KBytes  1.05 Mbits/sec   0.045 ms    0/   89 (0%)
[  3]  3.0- 4.0 sec   128 KBytes  1.05 Mbits/sec   0.045 ms    0/   89 (0%)
[  3]  4.0- 5.0 sec   128 KBytes  1.05 Mbits/sec   0.053 ms    0/   89 (0%)
[  3]  0.0- 5.0 sec   640 KBytes  1.05 Mbits/sec   0.053 ms    0/  446 (0%)
```

6. Khi có thêm dòng dữ liệu cạnh tranh H1 – H3, tỷ lệ mất gói tin trên dòng H2 – H4 là 4.7%, nếu thêm dòng cạnh tranh thì tỷ lệ này tăng lên tiếp đến 7.6%:

```
H2:~$ iperf -c 192.168.4.130 -i 1 -t 5 -u
```

```
------------------------------------------------------------
Client connecting to 192.168.4.130, UDP port 5001
Sending 1470 byte datagrams, IPG target: 11215.21 us (kalman adjust)
UDP buffer size:  208 KByte (default)
------------------------------------------------------------
[  3] local 192.168.2.123 port 42655 connected with 192.168.4.130 port 5001
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0- 1.0 sec   131 KBytes  1.07 Mbits/sec
[  3]  1.0- 2.0 sec   128 KBytes  1.05 Mbits/sec
[  3]  2.0- 3.0 sec   128 KBytes  1.05 Mbits/sec
[  3]  3.0- 4.0 sec   128 KBytes  1.05 Mbits/sec
[  3]  0.0- 5.0 sec   640 KBytes  1.05 Mbits/sec
[  3] Sent 446 datagrams
[  3] Server Report:
[  3]  0.0- 5.0 sec   610 KBytes   994 Kbits/sec  21.624 ms   21/  446 (4.7%)
```

```
H4:~$ iperf -su -i 1

------------------------------------------------------------
Server listening on UDP port 5001
Receiving 1470 byte datagrams
UDP buffer size:  208 KByte (default)
------------------------------------------------------------
[  3] local 192.168.4.130 port 5001 connected with 192.168.2.123 port 42655
[ ID] Interval       Transfer     Bandwidth        Jitter   Lost/Total Datagrams
[  3]  0.0- 1.0 sec   115 KBytes   941 Kbits/sec  22.038 ms    4/   84 (4.8%)
[  3]  1.0- 2.0 sec   128 KBytes  1.05 Mbits/sec  21.333 ms    2/   91 (2.2%)
[  3]  2.0- 3.0 sec   113 KBytes   929 Kbits/sec  21.603 ms   10/   89 (11%)
[  3]  3.0- 4.0 sec   128 KBytes  1.05 Mbits/sec  20.626 ms    4/   93 (4.3%)
[  3]  4.0- 5.0 sec   126 KBytes  1.03 Mbits/sec  21.624 ms    1/   89 (1.1%)
[  3]  0.0- 5.0 sec   610 KBytes   994 Kbits/sec  21.624 ms   21/  446 (4.7%)
```

```
H2:~$ iperf -c 192.168.4.130 -i 1 -t 5 -u

------------------------------------------------------------
Client connecting to 192.168.4.130, UDP port 5001
Sending 1470 byte datagrams, IPG target: 11215.21 us (kalman adjust)
UDP buffer size:  208 KByte (default)
------------------------------------------------------------
[  3] local 192.168.2.123 port 42634 connected with 192.168.4.130 port 5001
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0- 1.0 sec   131 KBytes  1.07 Mbits/sec
[  3]  1.0- 2.0 sec   128 KBytes  1.05 Mbits/sec
[  3]  2.0- 3.0 sec   128 KBytes  1.05 Mbits/sec
[  3]  3.0- 4.0 sec   128 KBytes  1.05 Mbits/sec
[  3]  0.0- 5.0 sec   640 KBytes  1.05 Mbits/sec
[  3] Sent 446 datagrams
[  3] Server Report:
[  3]  0.0- 5.0 sec   591 KBytes   963 Kbits/sec  18.582 ms   34/  446 (7.6%)
```

7. Kết luận từ thực nghiệm:
   - Hệ thống kết nối mạng tốc độ cao (khoảng 500Mbps) nhưng có 1 kênh kết nối tốc độ thấp (R2-R3) sẽ hạn chết tốc độ end-to-end của tất cả các dòng dữ liệu qua nó (H2 – H4: 10Mbps)
   - Khi có thêm 1 dòng dữ liệu cạnh tranh (H1 – H3) thì dòng dữ liệu H2 – H4 giảm còn 1/10 (khoảng 1Mbps)
   - Khi có thêm 1 dòng dữ liệu cạnh tranh nữa (R1-R2) thì dòng H2 – H4 giảm tiếp còn 1/10 (100Kbps)
   - Với dòng UDP, tỷ lệ mất gói tin từ 0% lên 4.7% rồi 7.6% khi xuất hiện các dòng dữ liệu cạnh tranh (TCP có cơ chế xác nhận để xử lý truyền lại gói tin bị mất nên ảnh hưởng đến tốc độ)
   - ⇨ Tốc độ dòng dữ liệu cũng như tỷ lệ gói tin bị drop ảnh hưởng rất lớn bởi các dòng cạnh tranh

2 Thực hành DiffServ

2.1 Cài đặt PHB cho router R2

1. Kiểm tra các *qdisc* mặc định *fq_codel* trên các card mạng của R2. Xác định giao diện kết nối mạng R2 - R3 có băng thông hạn chế là *enp0s10*:

```
R2:~$ ifconfig -a

enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.2.1  netmask 255.255.255.0  broadcast 192.168.2.255
        inet6 fe80::a00:27ff:fe87:c4c  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:87:0c:4c  txqueuelen 1000  (Ethernet)
        RX packets 170969  bytes 258448582 (258.4 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 26640  bytes 1883250 (1.8 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

enp0s9: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.10.1.2  netmask 255.255.255.0  broadcast 10.10.1.255
        inet6 fe80::a00:27ff:fe61:de17  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:61:de:17  txqueuelen 1000  (Ethernet)
        RX packets 1770881  bytes 2680606618 (2.6 GB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 238270  bytes 16169172 (16.1 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

enp0s10: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.10.2.1  netmask 255.255.255.0  broadcast 10.10.2.255
        inet6 fe80::a00:27ff:fef9:af90  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:f9:af:90  txqueuelen 1000  (Ethernet)
        RX packets 333126  bytes 22590292 (22.5 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 2202315  bytes 4618803440 (4.6 GB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0


R2:~$ sudo tc qdisc show

qdisc noqueue 0: dev lo root refcnt 2
qdisc fq_codel 0: dev enp0s3 root refcnt 2 limit 10240p flows 1024 quantum 1514 target 5.0ms
interval 100.0ms memory_limit 32Mb ecn
qdisc fq_codel 0: dev enp0s8 root refcnt 2 limit 10240p flows 1024 quantum 1514 target 5.0ms
interval 100.0ms memory_limit 32Mb ecn
qdisc fq_codel 0: dev enp0s9 root refcnt 2 limit 10240p flows 1024 quantum 1514 target 5.0ms
interval 100.0ms memory_limit 32Mb ecn
qdisc fq_codel 0: dev enp0s10 root refcnt 2 limit 10240p flows 1024 quantum 1514 target 5.0ms
interval 100.0ms memory_limit 32Mb ecn


R2:~$ sudo tc qdisc show dev enp0s10

qdisc fq_codel 0: root refcnt 2 limit 10240p flows 1024 quantum 1514 target 5.0ms interval 100.0ms
memory_limit 32Mb ecn
```

2. Nếu trên kết nối mạng này đang có *qdisc* khác, có thể reset về trạng thái mặc định ban đầu:

```
R2:~$ sudo tc qdisc del dev enp0s10 root
```

3. Xem các *filter* đang được áp dụng cho kết nối mạng mạng (ban đầu mặc định là không có filter nào được thiết lập cho tất cả các kết nối mạng của router):

```
R2:~$ sudo tc filter show dev enp0s10

filter parent 1: protocol ip pref 49146 u32 chain 0
filter parent 1: protocol ip pref 49146 u32 chain 0 fh 806: ht divisor 1
filter parent 1: protocol ip pref 49146 u32 chain 0 fh 806::800 order 2048 key ht 806 bkt 0 flowid
1:1 not_in_hw
  match 00120000/00ff0000 at 0
filter parent 1: protocol ip pref 49147 u32 chain 0
```

```
filter parent 1: protocol ip pref 49147 u32 chain 0 fh 805: ht divisor 1
filter parent 1: protocol ip pref 49148 u32 chain 0
filter parent 1: protocol ip pref 49148 u32 chain 0 fh 804: ht divisor 1
filter parent 1: protocol ip pref 49149 u32 chain 0
filter parent 1: protocol ip pref 49149 u32 chain 0 fh 803: ht divisor 1
filter parent 1: protocol ip pref 49150 u32 chain 0
filter parent 1: protocol ip pref 49150 u32 chain 0 fh 802: ht divisor 1
filter parent 1: protocol ip pref 49150 u32 chain 0 fh 802::800 order 2048 key ht 802 bkt 0 flowid
1:3 not_in_hw
  match 0a0a0202/ffffffff at 16
filter parent 1: protocol ip pref 49151 u32 chain 0
filter parent 1: protocol ip pref 49151 u32 chain 0 fh 801: ht divisor 1
filter parent 1: protocol ip pref 49151 u32 chain 0 fh 801::800 order 2048 key ht 801 bkt 0 flowid
1:2 not_in_hw
  match c0a80178/ffffffff at 12
filter parent 1: protocol ip pref 49152 u32 chain 0
filter parent 1: protocol ip pref 49152 u32 chain 0 fh 800: ht divisor 1
```

4. Áp dụng *qdisc prio* vào kết nối mạng *enp0s10*. Lệnh bên dưới gắn *qdisc prio* vào nút gốc (root) của các *qdisc* áp dụng cho card mạng *enp0s10*. Tham số *handle* có ý nghĩa như ID của nút trên cây. (nhắc lại: tổ chức qdisc cho mỗi card mạng là dạng đệ qui, bên trong 1 qdisc có nhiều class và mỗi class có thể được áp dụng với một qdisc khác).

```
R2:~$ sudo tc qdisc add dev enp0s10 root handle 1: prio
R2:~$ sudo tc qdisc show dev enp0s10

qdisc prio 1: root refcnt 2 bands 3 priomap 1 2 2 2 1 2 0 0 1 1 1 1 1 1 1 1
```

5. Qdisc *prio* được cài đặt sẵn trong linux kernel, với 3 hàng đợi (class) mặc định theo mức ưu tiên từ thấp đến cao.

```
R2:~$ sudo tc class show dev enp0s10

class prio 1:1 parent 1:
class prio 1:2 parent 1:
class prio 1:3 parent 1:
```

6. Thiết lập thông số min/max băng thông cũng như các thông số khác cho class thường được thực hiện bằng lệnh *"tc class"*, ở đây ta gắn luôn *qdisc tbf* vào 2 nút class trên cây (handle là 1:2 và 1:3) cùng với thông số giới hạn băng thông cho 2 qdisc này ➔ lượng băng thông còn lại được giành cho class 1:1

```
R2:~$ sudo tc qdisc add dev enp0s10 parent 1:2 handle 20: tbf rate 500kbit burst 3000
limit 5000
R2:~$ sudo tc qdisc add dev enp0s10 parent 1:3 handle 30: tbf rate 1mbit burst 3000
limit 9000
R2:~$ sudo tc qdisc show dev enp0s10

qdisc prio 1: root refcnt 2 bands 3 priomap 1 2 2 2 1 2 0 0 1 1 1 1 1 1 1 1
qdisc tbf 30: parent 1:3 rate 1Mbit burst 3000b lat 48.0ms
qdisc tbf 20: parent 1:2 rate 500Kbit burst 3000b lat 32.0ms
```

7. Khai báo 3 filter dựa trên địa chỉ IP nguồn/đích để quyết định gói tin sẽ được xử lý theo class nào. Cụ thể là dòng H2 – H4 được gán cho class 1:1, dòng H1 – H3 được gán cho class 1:2 và dòng R2 – G3 được gán cho class 1:3

```
R2:~$ sudo tc filter add dev enp0s10 parent 1: protocol ip u32 match \
ip src 192.168.2.123 flowid 1:1
R2:~$ sudo tc filter add dev enp0s10 parent 1: protocol ip u32 match \
ip src 192.168.1.120 flowid 1:2
R2:~$ sudo tc filter add dev enp0s10 parent 1: protocol ip u32 match \
ip dst 10.10.2.2 flowid 1:3

R2:~$ sudo tc filter show dev enp0s10
```

```
filter parent 1: protocol ip pref 49150 u32 chain 0
filter parent 1: protocol ip pref 49150 u32 chain 0 fh 802: ht divisor 1
filter parent 1: protocol ip pref 49150 u32 chain 0 fh 802::800 order 2048 key ht 802 bkt 0 flowid
1:3 not_in_hw
  match 0a0a0202/ffffffff at 16
filter parent 1: protocol ip pref 49151 u32 chain 0
filter parent 1: protocol ip pref 49151 u32 chain 0 fh 801: ht divisor 1
filter parent 1: protocol ip pref 49151 u32 chain 0 fh 801::800 order 2048 key ht 801 bkt 0 flowid
1:2 not_in_hw
  match c0a80178/ffffffff at 12
filter parent 1: protocol ip pref 49152 u32 chain 0
filter parent 1: protocol ip pref 49152 u32 chain 0 fh 800: ht divisor 1
filter parent 1: protocol ip pref 49152 u32 chain 0 fh 800::800 order 2048 key ht 800 bkt 0 flowid
1:1 not_in_hw
  match c0a8027b/ffffffff at 12
```

2.2 Triển khai các luồng cạnh tranh trên router R2 đã được xử lý PHB

8. Chạy lại đồng thời 3 luồng (H2 – H4: class 1, H1 – H3: class 2 và R2 – R3: class 3) thì thấy băng thông H2 – H4 đã được đảm bảo, không bị 2 luồng cạnh tranh kia làm ảnh hưởng. Còn băng thông của luồng H1 – H3 và R2 -0 R3 bị giới hạn đúng như thết lập trong class tương ứng là 500Kbps và 1Mbps.

```
H4:~$ iperf -s -i 1

------------------------------------------------------------
Server listening on TCP port 5001
TCP window size:  128 KByte (default)
------------------------------------------------------------
[  4] local 192.168.4.130 port 5001 connected with 192.168.2.123 port 51872
[ ID] Interval       Transfer     Bandwidth
[  4]  0.0- 1.0 sec  1.07 MBytes  9.00 Mbits/sec
[  4]  1.0- 2.0 sec  1.12 MBytes  9.36 Mbits/sec
[  4]  2.0- 3.0 sec  1.12 MBytes  9.42 Mbits/sec
[  4]  3.0- 4.0 sec  1.12 MBytes  9.41 Mbits/sec
[  4]  4.0- 5.0 sec  1.15 MBytes  9.67 Mbits/sec
[  4]  5.0- 6.0 sec  1.15 MBytes  9.66 Mbits/sec


H3:~$ iperf -s

------------------------------------------------------------
Server listening on TCP port 5001
TCP window size:  128 KByte (default)
------------------------------------------------------------
[  4] local 192.168.3.115 port 5001 connected with 192.168.1.120 port 54754
[ ID] Interval       Transfer     Bandwidth
[  4]  0.0-301.4 sec  16.6 MBytes   462 Kbits/sec


R3:~$ iperf -s

------------------------------------------------------------
Server listening on TCP port 5001
TCP window size:  128 KByte (default)
------------------------------------------------------------
[  4] local 10.10.2.2 port 5001 connected with 10.10.2.1 port 53494
[ ID] Interval       Transfer     Bandwidth
[  4]  0.0-301.5 sec  33.1 MBytes   921 Kbits/sec
```

9. Chuyển luồng H2 – H4 sang UDP thì thấy tỷ lệ drop gói tin giảm còn 0%

```
H4:~$ iperf -s -i 1 -u

------------------------------------------------------------
Server listening on UDP port 5001
Receiving 1470 byte datagrams
UDP buffer size:  208 KByte (default)
------------------------------------------------------------
[  3] local 192.168.4.130 port 5001 connected with 192.168.2.123 port 43093
```

```
[ ID] Interval       Transfer     Bandwidth        Jitter   Lost/Total Datagrams
[  3]  0.0- 1.0 sec   129 KBytes  1.06 Mbits/sec   0.087 ms    0/   90 (0%)
[  3]  1.0- 2.0 sec   128 KBytes  1.05 Mbits/sec   0.067 ms    0/   89 (0%)
[  3]  2.0- 3.0 sec   128 KBytes  1.05 Mbits/sec   0.074 ms    0/   89 (0%)
[  3]  3.0- 4.0 sec   128 KBytes  1.05 Mbits/sec   0.134 ms    0/   89 (0%)
[  3]  4.0- 5.0 sec   128 KBytes  1.05 Mbits/sec   0.110 ms    0/   89 (0%)
```

10. Giữ nguyên cấu hình 2 luồng 1:2 và 1:3, xóa filter của luồng 1:1 (là H2 – H4). Chú ý các tham số *handle* và *pref* phải chính xác như khi show filter (tránh xóa nhầm filter):

```
R2:~$ sudo tc filter show dev enp0s10

filter parent 1: protocol ip pref 49150 u32 chain 0
filter parent 1: protocol ip pref 49150 u32 chain 0 fh 802: ht divisor 1
filter parent 1: protocol ip pref 49150 u32 chain 0 fh 802::800 order 2048 key ht 802 bkt 0 flowid
1:3 not_in_hw
  match 0a0a0202/ffffffff at 16
filter parent 1: protocol ip pref 49151 u32 chain 0
filter parent 1: protocol ip pref 49151 u32 chain 0 fh 801: ht divisor 1
filter parent 1: protocol ip pref 49151 u32 chain 0 fh 801::800 order 2048 key ht 801 bkt 0 flowid
1:2 not_in_hw
  match c0a80178/ffffffff at 12
filter parent 1: protocol ip pref 49152 u32 chain 0
filter parent 1: protocol ip pref 49152 u32 chain 0 fh 800: ht divisor 1
filter parent 1: protocol ip pref 49152 u32 chain 0 fh 800::800 order 2048 key ht 800 bkt 0 flowid
1:1 not_in_hw
  match c0a8027b/ffffffff at 12



R2:~$ sudo tc filter delete dev enp0s10 parent 1: handle 800::800 pref 49152 u32
R2:~$ sudo tc filter show dev enp0s10

filter parent 1: protocol ip pref 49150 u32 chain 0
filter parent 1: protocol ip pref 49150 u32 chain 0 fh 802: ht divisor 1
filter parent 1: protocol ip pref 49150 u32 chain 0 fh 802::800 order 2048 key ht 802 bkt 0 flowid
1:3 not_in_hw
  match 0a0a0202/ffffffff at 16
filter parent 1: protocol ip pref 49151 u32 chain 0
filter parent 1: protocol ip pref 49151 u32 chain 0 fh 801: ht divisor 1
filter parent 1: protocol ip pref 49151 u32 chain 0 fh 801::800 order 2048 key ht 801 bkt 0 flowid
1:2 not_in_hw
  match c0a80178/ffffffff at 12
filter parent 1: protocol ip pref 49152 u32 chain 0
filter parent 1: protocol ip pref 49152 u32 chain 0 fh 800: ht divisor 1
```

11. Chạy lại kịch bản bên trên (luồng H2 – H4 không còn được áp dụng filter để xử lý ưu tiên theo class 1:1): tốc độ luồng H2 – H4 lại bị giảm nhiều khi có 2 luồng cạnh tranh. tỷ lệ mất gói tin luồng H2 – H4 khi chuyển sang UDP cũng rất lớn (trên 50%):

```
H4:~$ iperf -s -i 1

------------------------------------------------------------
Server listening on TCP port 5001
TCP window size:  128 KByte (default)
------------------------------------------------------------
[  4] local 192.168.4.130 port 5001 connected with 192.168.2.123 port 51876
[ ID] Interval       Transfer     Bandwidth
[  4]  0.0- 1.0 sec  39.6 KBytes   324 Kbits/sec
[  4]  1.0- 2.0 sec  31.1 KBytes   255 Kbits/sec
[  4]  2.0- 3.0 sec  39.6 KBytes   324 Kbits/sec
[  4]  3.0- 4.0 sec  29.7 KBytes   243 Kbits/sec
[  4]  4.0- 5.0 sec  39.6 KBytes   324 Kbits/sec

H4:~$ iperf -s -i 1 -u

------------------------------------------------------------
Server listening on UDP port 5001
Receiving 1470 byte datagrams
UDP buffer size:  208 KByte (default)
------------------------------------------------------------
```

```
[  3] local 192.168.4.130 port 5001 connected with 192.168.2.123 port 32968
[ ID] Interval       Transfer     Bandwidth        Jitter   Lost/Total Datagrams
[  3]  0.0- 1.0 sec  61.7 KBytes   506 Kbits/sec   3.201 ms   42/   85 (49%)
[  3]  1.0- 2.0 sec  58.9 KBytes   482 Kbits/sec   2.903 ms   47/   88 (53%)
[  3]  2.0- 3.0 sec  60.3 KBytes   494 Kbits/sec   3.075 ms   49/   91 (54%)
[  3]  3.0- 4.0 sec  58.9 KBytes   482 Kbits/sec   2.866 ms   47/   88 (53%)
```

2.3 Áp dụng DSCP codepoint vào filter thay cho địa chỉ IP

12. Khai báo lại filter số 1: thay vì cố định class 1:1 theo địa chỉ IP như trước, chuyển sang áp dụng DSCP codepoint 0x12 (AF21):

```
R2:~$ sudo tc filter add dev enp0s10 parent 1: protocol ip u32 match \
ip tos 0x12 0xff flowid 1:1
R2:~$ sudo tc filter show dev enp0s10

filter parent 1: protocol ip pref 49147 u32 chain 0
filter parent 1: protocol ip pref 49147 u32 chain 0 fh 805: ht divisor 1
filter parent 1: protocol ip pref 49147 u32 chain 0 fh 805::800 order 2048 key ht 805 bkt 0 flowid
1:1 not_in_hw
  match 00120000/00ff0000 at 0
filter parent 1: protocol ip pref 49148 u32 chain 0
filter parent 1: protocol ip pref 49148 u32 chain 0 fh 804: ht divisor 1
filter parent 1: protocol ip pref 49149 u32 chain 0
filter parent 1: protocol ip pref 49149 u32 chain 0 fh 803: ht divisor 1
filter parent 1: protocol ip pref 49150 u32 chain 0
filter parent 1: protocol ip pref 49150 u32 chain 0 fh 802: ht divisor 1
filter parent 1: protocol ip pref 49150 u32 chain 0 fh 802::800 order 2048 key ht 802 bkt 0 flowid
1:3 not_in_hw
  match 0a0a0202/ffffffff at 16
filter parent 1: protocol ip pref 49151 u32 chain 0
filter parent 1: protocol ip pref 49151 u32 chain 0 fh 801: ht divisor 1
filter parent 1: protocol ip pref 49151 u32 chain 0 fh 801::800 order 2048 key ht 801 bkt 0 flowid
1:2 not_in_hw
  match c0a80178/ffffffff at 12
filter parent 1: protocol ip pref 49152 u32 chain 0
filter parent 1: protocol ip pref 49152 u32 chain 0 fh 800: ht divisor 1
```

13. Chạy động thời 3 luồng cạnh tranh như trên, nhưng với kích bản H2 – H4 không được gán DSCP codepoint 0x12 (iperf -S). Đồng thời dùng *tcpdump* trên R2 để kiểm tra giá trị ToS trong gói tin IP trên kênh truyền H2 – H4. Nhận thấy khi không được gán DSCP codepoing 0x12, luồng H2 – H4 bị ảnh hưởng tốc độ và mất gói tin. Khi gán codepoint 0x12, luồng H2 – H4 ổn định tốc độ cao và độ mất mát gói tin về 0%:

```
R2:~$ sudo tcpdump -i enp0s10 -vv

12:12:54.722778 IP (tos 0x0, ttl 64, id 27209, offset 0, flags [DF], proto TCP (6), length 52)
    10.10.2.2.5001 > R2.53502: Flags [.], cksum 0xe46d (correct), seq 1, ack 149144, win 2125,
options [nop,nop,TS val 1238727438 ecr 2815787360], length 0
12:12:54.728251 IP (tos 0x0, ttl 62, id 44720, offset 0, flags [DF], proto TCP (6), length 1500)
    192.168.1.120.54762 > 192.168.3.115.5001: Flags [.], cksum 0x8c0a (incorrect -> 0x41f7), seq
73848:75296, ack 1, win 502, options [nop,nop,TS val 3865997931 ecr 2270429424], length 1448
12:12:54.729285 IP (tos 0x0, ttl 63, id 5245, offset 0, flags [DF], proto TCP (6), length 52)
    192.168.3.115.5001 > 192.168.1.120.54762: Flags [.], cksum 0xe96e (correct), seq 1, ack 75296,
win 761, options [nop,nop,TS val 2270429472 ecr 3865997882], length 0



H2:~$ iperf -c 192.168.4.130 -i 1 -t 10

------------------------------------------------------------
Client connecting to 192.168.4.130, TCP port 5001
TCP window size:  102 KByte (default)
------------------------------------------------------------
[  3] local 192.168.2.123 port 51882 connected with 192.168.4.130 port 5001
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0- 1.0 sec   139 KBytes  1.14 Mbits/sec
[  3]  1.0- 2.0 sec  62.2 KBytes   510 Kbits/sec
[  3]  2.0- 3.0 sec  62.2 KBytes   510 Kbits/sec
[  3]  3.0- 4.0 sec  62.2 KBytes   510 Kbits/sec
```

```
[  3]  4.0- 5.0 sec  62.2 KBytes   510 Kbits/sec
```

```
H2:~$ iperf -c 192.168.4.130 -i 1 -t 10 -S 0x10

------------------------------------------------------------
Client connecting to 192.168.4.130, TCP port 5001
TCP window size: 93.5 KByte (default)
------------------------------------------------------------
[  3] local 192.168.2.123 port 51884 connected with 192.168.4.130 port 5001
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0- 1.0 sec  1.75 MBytes  14.7 Mbits/sec
[  3]  1.0- 2.0 sec  1.50 MBytes  12.6 Mbits/sec
[  3]  2.0- 3.0 sec  1.12 MBytes  9.44 Mbits/sec
[  3]  3.0- 4.0 sec  1.50 MBytes  12.6 Mbits/sec
```

```
H4:~$ iperf -s -i 1 -u

------------------------------------------------------------
Server listening on UDP port 5001
Receiving 1470 byte datagrams
UDP buffer size:  208 KByte (default)
------------------------------------------------------------
[  3] local 192.168.4.130 port 5001 connected with 192.168.2.123 port 52713
[ ID] Interval       Transfer     Bandwidth        Jitter   Lost/Total Datagrams
[  3]  0.0- 1.0 sec  53.1 KBytes   435 Kbits/sec   3.000 ms   54/   91 (59%)
[  3]  1.0- 2.0 sec  58.9 KBytes   482 Kbits/sec   2.881 ms   47/   88 (53%)
[  3]  2.0- 3.0 sec  51.7 KBytes   423 Kbits/sec   3.646 ms   57/   93 (61%)
[  3]  0.0- 3.5 sec   185 KBytes   430 Kbits/sec   3.261 ms  186/  315 (59%)
[  4] local 192.168.4.130 port 5001 connected with 192.168.2.123 port 44002
[  4]  0.0- 1.0 sec   129 KBytes  1.06 Mbits/sec   0.110 ms    0/   90 (0%)
[  4]  1.0- 2.0 sec   128 KBytes  1.05 Mbits/sec   0.084 ms    0/   89 (0%)
[  4]  2.0- 3.0 sec   128 KBytes  1.05 Mbits/sec   0.055 ms    0/   89 (0%)
[  4]  3.0- 4.0 sec   128 KBytes  1.05 Mbits/sec   0.067 ms    0/   89 (0%)
[  4]  4.0- 5.0 sec   128 KBytes  1.05 Mbits/sec   0.059 ms    0/   89 (0%)
[  4]  0.0- 5.2 sec   672 KBytes  1.05 Mbits/sec   0.058 ms    0/  468 (0%)
```

2.4 DiffServ networking

14. R2 đã được thiết lập đầy đủ PHB để làm nhiệm vụ một core router trong DiffServ netwok. Gả sử DiffServ network cần thiết lập gồm có router R1, R2, R3 trong đó R1 là ingress, R2 là core và R3 là egress. DiffServ network này đã hỗ trợ DSCP codepoint 0x12 và chấp nhận các luồng giao thông đăng ký hoạt động trên DSCP class này. Việc đăng ký thường được thực hiện ở tầng ứng dụng (ví dụ khi người dùng mua một dịch vụ xem phim trả tiền có cam kết chất lượng đường truyền) và có thể dựa trên địa chỉ IP nguồn, IP đích, cổng dịch vụ, v.v.. Router R1 đóng vai trò là ingress router có nhiệm vụ khởi tạo trường ToS cho các gói tin đi vào DiffServ network và R3, với vai trò là egress, sẽ reset trường ToS cho các gói tin đi ra khỏi DiffServ network.
    Sử dụng iptables để triển khai chức năng ingress của R1, dựa trên địa chỉ IP nguồn và IP đích. Các gói IP gửi từ H1 đến H4 khi đi qua R1 sẽ được đặt DSCP codepoint là 0x12 và được xử lý ưu tiên trong DiffServ network:

```
R1:~$ sudo iptables -t mangle -A POSTROUTING \
       -s 192.168.1.120 -d 192.168.4.130 \
       -j TOS --set-tos 0x12
R1:~$ sudo iptables -t mangle -L --line-number

Chain PREROUTING (policy ACCEPT)
num  target     prot opt source               destination

Chain FORWARD (policy ACCEPT)
num  target     prot opt source               destination

Chain POSTROUTING (policy ACCEPT)
num  target     prot opt source               destination
1    TOS        all  --  192.168.1.120        192.168.4.130        TOS set 0x12/0xff
```

15. Trên R2 hiện đang có filter theo địa chỉ IP của H1, cần xóa filter này đi để không xử lý các gói tin gửi từ H1 nữa:

```
R2:~$ sudo tc filter show dev enp0s10

filter parent 1: protocol ip pref 49145 u32 chain 0
filter parent 1: protocol ip pref 49145 u32 chain 0 fh 807: ht divisor 1
filter parent 1: protocol ip pref 49145 u32 chain 0 fh 807::800 order 2048 key ht 807 bkt 0 flowid
1:1 not_in_hw
  match 00120000/00ff0000 at 0
filter parent 1: protocol ip pref 49146 u32 chain 0
filter parent 1: protocol ip pref 49150 u32 chain 0 fh 802: ht divisor 1
filter parent 1: protocol ip pref 49150 u32 chain 0 fh 802::800 order 2048 key ht 802 bkt 0 flowid
1:3 not_in_hw
  match 0a0a0202/ffffffff at 16
filter parent 1: protocol ip pref 49151 u32 chain 0
filter parent 1: protocol ip pref 49151 u32 chain 0 fh 801: ht divisor 1
filter parent 1: protocol ip pref 49151 u32 chain 0 fh 801::800 order 2048 key ht 801 bkt 0 flowid
1:2 not_in_hw
  match c0a80178/ffffffff at 12
filter parent 1: protocol ip pref 49152 u32 chain 0
filter parent 1: protocol ip pref 49152 u32 chain 0 fh 800: ht divisor 1


R2:~$ sudo tc filter delete dev enp0s10 parent 1: handle 801::800 pref 49151 u32
R2:~$ sudo tc filter show dev enp0s10

filter parent 1: protocol ip pref 49145 u32 chain 0
filter parent 1: protocol ip pref 49145 u32 chain 0 fh 807: ht divisor 1
filter parent 1: protocol ip pref 49145 u32 chain 0 fh 807::800 order 2048 key ht 807 bkt 0 flowid
1:1 not_in_hw
  match 00120000/00ff0000 at 0
filter parent 1: protocol ip pref 49146 u32 chain 0
filter parent 1: protocol ip pref 49150 u32 chain 0 fh 802: ht divisor 1
filter parent 1: protocol ip pref 49150 u32 chain 0 fh 802::800 order 2048 key ht 802 bkt 0 flowid
1:3 not_in_hw
  match 0a0a0202/ffffffff at 16
filter parent 1: protocol ip pref 49151 u32 chain 0
filter parent 1: protocol ip pref 49151 u32 chain 0 fh 801: ht divisor 1
filter parent 1: protocol ip pref 49152 u32 chain 0
filter parent 1: protocol ip pref 49152 u32 chain 0 fh 800: ht divisor 1
```

16. Bắt các gói tin ICMP đi qua R2 để kiểm tra ToS đã được xử lý đúng khi ping từ H1 đến H3 và H1 đến H4. Thấy dòng ICMP H1 – H3 có ToS = 0 (mặc định). Dòng ICMP H1 – H4 có ToS 0x12:

```
R2:~$ sudo tcpdump icmp -i enp0s10 -vv

tcpdump: listening on enp0s10, link-type EN10MB (Ethernet), capture size 262144 bytes
14:15:59.887498 IP (tos 0x0, ttl 62, id 62151, offset 0, flags [DF], proto ICMP (1), length 84)
    192.168.1.120 > 192.168.3.115: ICMP echo request, id 8, seq 1, length 64
14:15:59.888773 IP (tos 0x0, ttl 63, id 23023, offset 0, flags [none], proto ICMP (1), length 84)
    192.168.3.115 > 192.168.1.120: ICMP echo reply, id 8, seq 1, length 64
14:16:00.889466 IP (tos 0x0, ttl 62, id 62275, offset 0, flags [DF], proto ICMP (1), length 84)
    192.168.1.120 > 192.168.3.115: ICMP echo request, id 8, seq 2, length 64
14:16:00.890589 IP (tos 0x0, ttl 63, id 23183, offset 0, flags [none], proto ICMP (1), length 84)
    192.168.3.115 > 192.168.1.120: ICMP echo reply, id 8, seq 2, length 64
14:16:27.946485 IP (tos 0x12,ECT(0), ttl 62, id 20994, offset 0, flags [DF], proto ICMP (1), length
84)
    192.168.1.120 > 192.168.4.130: ICMP echo request, id 9, seq 1, length 64
14:16:27.948435 IP (tos 0x12,ECT(0), ttl 62, id 34613, offset 0, flags [none], proto ICMP (1),
length 84)
    192.168.4.130 > 192.168.1.120: ICMP echo reply, id 9, seq 1, length 64
14:16:28.948102 IP (tos 0x12,ECT(0), ttl 62, id 21012, offset 0, flags [DF], proto ICMP (1), length
84)
    192.168.1.120 > 192.168.4.130: ICMP echo request, id 9, seq 2, length 64
14:16:28.950046 IP (tos 0x12,ECT(0), ttl 62, id 34680, offset 0, flags [none], proto ICMP (1),
length 84)
    192.168.4.130 > 192.168.1.120: ICMP echo reply, id 9, seq 2, length 64
```

```
H1:~$ ping 192.168.3.115

PING 192.168.3.115 (192.168.3.115) 56(84) bytes of data.
64 bytes from 192.168.3.115: icmp_seq=1 ttl=61 time=2.58 ms
64 bytes from 192.168.3.115: icmp_seq=2 ttl=61 time=2.34 ms
^C
--- 192.168.3.115 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1002ms
rtt min/avg/max/mdev = 2.341/2.460/2.580/0.119 ms


H1:~$ ping 192.168.4.130

PING 192.168.4.130 (192.168.4.130) 56(84) bytes of data.
64 bytes from 192.168.4.130: icmp_seq=1 ttl=60 time=3.29 ms
64 bytes from 192.168.4.130: icmp_seq=2 ttl=60 time=3.29 ms
^C
--- 192.168.4.130 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1002ms
rtt min/avg/max/mdev = 3.285/3.288/3.291/0.003 ms
```

17. Tạo các dòng iperf từ H1 đến H3 và H1 đến H4. Kết quả thấy dòng H1 – H4 được gán DSCP codepoint 0x12 và được ưu tiên xử lý (tốc độ cao, tỷ lệ drop gói tin 0%)

```
H1:~$ iperf -c 192.168.3.115 -i 1 -t 3

------------------------------------------------------------
Client connecting to 192.168.3.115, TCP port 5001
TCP window size: 85.0 KByte (default)
------------------------------------------------------------
[  3] local 192.168.1.120 port 54772 connected with 192.168.3.115 port 5001
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0- 1.0 sec   139 KBytes  1.14 Mbits/sec
[  3]  1.0- 2.0 sec  62.2 KBytes   510 Kbits/sec
[  3]  2.0- 3.0 sec  31.1 KBytes   255 Kbits/sec
[  3]  0.0- 3.1 sec   232 KBytes   621 Kbits/sec


H1:~$ iperf -c 192.168.4.130 -i 1 -t 3

connect failed: Connection refused
hp@H1:~$ iperf -c 192.168.4.130 -i 1 -t 3
------------------------------------------------------------
Client connecting to 192.168.4.130, TCP port 5001
TCP window size:  314 KByte (default)
------------------------------------------------------------
[  3] local 192.168.1.120 port 50782 connected with 192.168.4.130 port 5001
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0- 1.0 sec  1.88 MBytes  15.7 Mbits/sec
[  3]  1.0- 2.0 sec  1.88 MBytes  15.7 Mbits/sec
[  3]  2.0- 3.0 sec  1.00 MBytes  8.39 Mbits/sec
[  3]  0.0- 3.1 sec  4.75 MBytes  12.8 Mbits/sec


H3:~$ iperf -s -u -i 1

------------------------------------------------------------
Server listening on UDP port 5001
Receiving 1470 byte datagrams
UDP buffer size:  208 KByte (default)
------------------------------------------------------------
[  3] local 192.168.3.115 port 5001 connected with 192.168.1.120 port 54613
[ ID] Interval       Transfer     Bandwidth        Jitter   Lost/Total Datagrams
[  3]  0.0- 1.0 sec  61.7 KBytes   506 Kbits/sec   3.459 ms   42/   85 (49%)
[  3]  1.0- 2.0 sec  58.9 KBytes   482 Kbits/sec   3.130 ms   47/   88 (53%)
[  3]  2.0- 3.0 sec  60.3 KBytes   494 Kbits/sec   3.125 ms   49/   91 (54%)
[  3]  0.0- 3.1 sec   184 KBytes   490 Kbits/sec   2.964 ms  140/  268 (52%)


H4:~$ iperf -s -i 1 -u

------------------------------------------------------------
Server listening on UDP port 5001
```

```
Receiving 1470 byte datagrams
UDP buffer size:  208 KByte (default)
------------------------------------------------------------
[  3] local 192.168.4.130 port 5001 connected with 192.168.1.120 port 41154
[ ID] Interval       Transfer     Bandwidth        Jitter   Lost/Total Datagrams
[  3]  0.0- 1.0 sec   129 KBytes  1.06 Mbits/sec   0.044 ms    0/   90 (0%)
[  3]  1.0- 2.0 sec   128 KBytes  1.05 Mbits/sec   0.065 ms    0/   89 (0%)
[  3]  2.0- 3.0 sec   128 KBytes  1.05 Mbits/sec   0.043 ms    0/   89 (0%)
[  3]  0.0- 3.0 sec   385 KBytes  1.05 Mbits/sec   0.043 ms    0/  268 (0%)
```

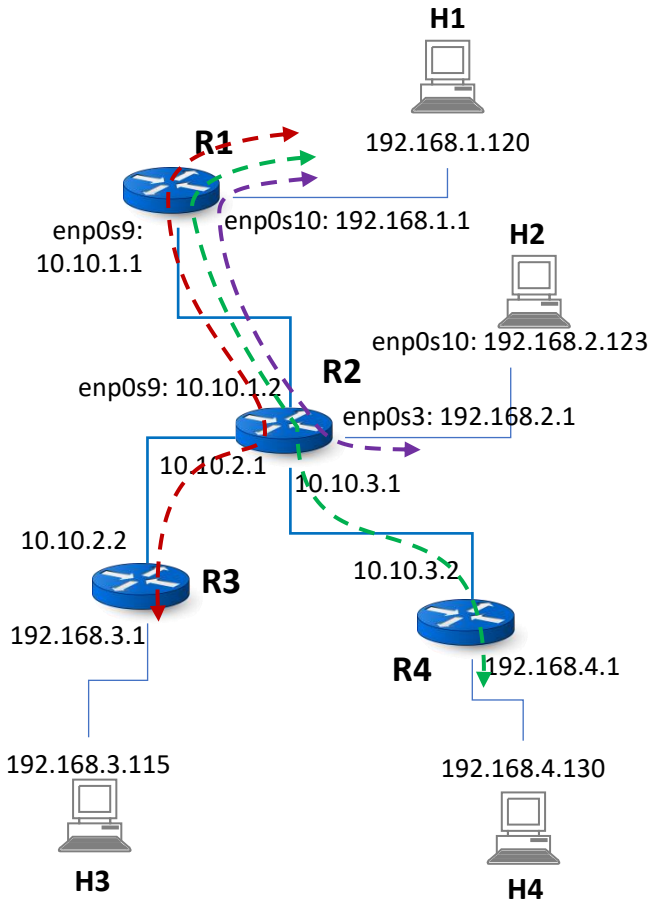
3 MPLS

3.1 LSR implementation by Linux Traffic Control

R1 là ingress LER (Label Edge Router)

R2, R3, R4 là LSR (Label Switch Router). R3 và/hoặc R4 có thể là egress LER.

Có 3 LSP (Label Switch Path): H1 → R1 → R2 → H2, H1 → R1 → R2→ R3 → H3 và H1 → R1 → R2 → R4 → H4



1. Thiết lập qdisc *ingress* LRE trên R1. Qdisc ingress được cài đặt trong linux kernel với chức năng nhận gói tin đi vào kết nối mạng và xử lý filter (các qdisc khác chủ yếu là kiểu egress, tức là xử lý gói tin đi ra kết nối mạng)

```
R1:~$ sudo tc qdisc add dev enp0s10 handle ffff: ingress
```

2. Thiết lập filter cho qdisc *ingress* vừa tạo, đặt label 123 cho các gói tin IP và switch sang kết nối mạng enp0s9 để gửi cho R2. Khi switch sang kết nối để gửi cho R2, cần thiết lập địa chỉ MAC destination cho gói tin (tầng 2) là địa chỉ MAC của R2:

```
R1:~$ sudo tc filter add dev enp0s10 protocol ip parent ffff: \
        flower dst_ip 192.168.2.0/24 \
                action mpls push protocol mpls_uc label 123 \
                action skbmod set dmac 08:00:27:61:de:17 \
                action mirred egress redirect dev enp0s9

R1:~$ tc filter show dev enp0s10 ingress
filter parent ffff: protocol ip pref 49152 flower chain 0
filter parent ffff: protocol ip pref 49152 flower chain 0 handle 0x1
  eth_type ipv4
  dst_ip 192.168.2.0/24
  not_in_hw
        action order 1: mpls  push protocol mpls_uc label 123 ttl 255 pipe
         index 1 ref 1 bind 1

        action order 2: skbmod pipe set dmac 08:00:27:61:de:17
         index 1 ref 1 bind 1

        action order 3: mirred (Egress Redirect to device enp0s9) stolen
        index 1 ref 1 bind 1
```

3. Sau khi thiết lập luật MPLS trên R1, thử ping từ H1 sang H2 thì chưa được nhưng có thể dùng *tcpdump* để thấy các gói tin ICMP/MPLS đã được gửi như sau. Kết nối enp0s10 giữa R1 và H1, có các gói tin ICMP Echo request gửi gửi H1

```
R1~$ sudo tcpdump -i enp0s10 -env
tcpdump: listening on enp0s10, link-type EN10MB (Ethernet), capture size 262144 bytes
11:00:56.071227 08:00:27:1c:c2:15 > 08:00:27:14:05:7e, ethertype IPv4 (0x0800), length 98: (tos
0x0, ttl 64, id 26751, offset 0, flags [DF], proto ICMP (1), length 84)
    192.168.1.120 > 192.168.2.123: ICMP echo request, id 16, seq 998, length 64
11:00:57.095166 08:00:27:1c:c2:15 > 08:00:27:14:05:7e, ethertype IPv4 (0x0800), length 98: (tos
0x0, ttl 64, id 26792, offset 0, flags [DF], proto ICMP (1), length 84)
    192.168.1.120 > 192.168.2.123: ICMP echo request, id 16, seq 999, length 64
11:00:58.119179 08:00:27:1c:c2:15 > 08:00:27:14:05:7e, ethertype IPv4 (0x0800), length 98: (tos
0x0, ttl 64, id 26931, offset 0, flags [DF], proto ICMP (1), length 84)
    192.168.1.120 > 192.168.2.123: ICMP echo request, id 16, seq 1000, length 64
```

4. Kết nối enp0s9 giữa R1 và R2, có các gói tin ICMP Echo request được bọc trong gói MPLS label 123. Địa chỉ MAC cũng được gửi đúng đến MAC của R2 (08:00:27:61:de:17):

```
R1:~$ sudo tcpdump -i enp0s9 -env
tcpdump: listening on enp0s9, link-type EN10MB (Ethernet), capture size 262144 bytes
11:02:25.161445 08:00:27:1c:c2:15 > 08:00:27:61:de:17, ethertype MPLS unicast (0x8847), length 102:
MPLS (label 123, exp 0, [S], ttl 255)
        (tos 0x0, ttl 64, id 37895, offset 0, flags [DF], proto ICMP (1), length 84)
    192.168.1.120 > 192.168.2.123: ICMP echo request, id 16, seq 1085, length 64
11:02:26.185441 08:00:27:1c:c2:15 > 08:00:27:61:de:17, ethertype MPLS unicast (0x8847), length 102:
MPLS (label 123, exp 0, [S], ttl 255)
        (tos 0x0, ttl 64, id 38066, offset 0, flags [DF], proto ICMP (1), length 84)
    192.168.1.120 > 192.168.2.123: ICMP echo request, id 16, seq 1086, length 64
11:02:27.209455 08:00:27:1c:c2:15 > 08:00:27:61:de:17, ethertype MPLS unicast (0x8847), length 102:
MPLS (label 123, exp 0, [S], ttl 255)
        (tos 0x0, ttl 64, id 38268, offset 0, flags [DF], proto ICMP (1), length 84)
    192.168.1.120 > 192.168.2.123: ICMP echo request, id 16, seq 1087, length 64
```

5. Kết nối enp0s3 giữa R2 và H2 chưa nhận được gói tin ICMP gửi từ H1. Lý do là R2 chưa được cấu hình MPLS nên gói tin MPLS gửi từ R1 đến chưa được xử lý

```
R2:~$ sudo tcpdump -i enp0s3 -env
tcpdump: listening on enp0s3, link-type EN10MB (Ethernet), capture size 262144 bytes
```

6. Thiết lập MPLS trên R2 để nhận luồng MPLS 123 gửi từ R1. Tương tự trên R1, cần gắn một qdisc ingress vào kết nối enp0s6 để nhận các gói tin MPLS từ R1

```
R2:~$ sudo tc qdisc add dev enp0s9 handle ffff: ingress
```

7. Đặt filter nhận các gói tin PMLS có label 123 và thực hiện các action sau:
   - Lấy gói IP từ gói MPLS này (nếu chuyển gói MPLS đến H2 thì nó sẽ không hiểu để xử lý)
   - Sửa địa chỉ MAC đích để chuyển đến H2
   - Switch gói tin sang kết nối enp0s3 để gửi cho H2

```
R2:~$ sudo tc filter add dev enp0s9 ingress \
        protocol mpls_uc flower mpls_label 123 \
                action mpls pop protocol ipv4 \
                action skbmod set dmac 08:00:27:75:25:d1 \
                action mirred egress redirect dev enp0s3

R2:~$ tc filter show dev enp0s9 ingress
filter protocol mpls_uc pref 49152 flower chain 0
filter protocol mpls_uc pref 49152 flower chain 0 handle 0x1
  eth_type 8847
  mpls_label 123
  not_in_hw
        action order 1: mpls  pop protocol ip pipe
         index 1 ref 1 bind 1

        action order 2: skbmod pipe set dmac 08:00:27:75:25:d1
         index 1 ref 1 bind 1

        action order 3: mirred (Egress Redirect to device enp0s3) stolen
        index 1 ref 1 bind 1
```

8. Ngay khi luồng MPLS label 123 được thiết lập trên R2, ping H1 – H2 đã thành công

```
H1:~$ ping 192.168.2.123
PING 192.168.2.123 (192.168.2.123) 56(84) bytes of data.
64 bytes from 192.168.2.123: icmp_seq=1 ttl=62 time=2.26 ms
64 bytes from 192.168.2.123: icmp_seq=2 ttl=62 time=2.03 ms
64 bytes from 192.168.2.123: icmp_seq=3 ttl=62 time=2.08 ms
```

9. Kiểm tra tiếp các gói tin được chuyển trên hệ thống bằng *tcpdump* như sau. Kết nối enp0s9 giữa R2 và H1 đã xử lý gói tin MPLS:
   - Chiều R1 ➔ R2: gói tin ICMP Echo request được bọc trong MPLS label 123
   - Chiều R2 ➔ R1: gói tin ICMP Echo reply không được gửi bằng MPLS

```
R2:~$ sudo tcpdump -i enp0s9 -env
tcpdump: listening on enp0s9, link-type EN10MB (Ethernet), capture size 262144 bytes
11:34:58.748397 08:00:27:1c:c2:15 > 08:00:27:61:de:17, ethertype MPLS unicast (0x8847), length 102:
MPLS (label 123, exp 0, [S], ttl 255)
        (tos 0x0, ttl 64, id 28062, offset 0, flags [DF], proto ICMP (1), length 84)
    192.168.1.120 > 192.168.2.123: ICMP echo request, id 18, seq 61, length 64
11:34:58.749055 08:00:27:61:de:17 > 08:00:27:4f:d1:59, ethertype IPv4 (0x0800), length 98: (tos
0x0, ttl 63, id 62230, offset 0, flags [none], proto ICMP (1), length 84)
    192.168.2.123 > 192.168.1.120: ICMP echo reply, id 18, seq 61, length 64
11:34:59.749577 08:00:27:1c:c2:15 > 08:00:27:61:de:17, ethertype MPLS unicast (0x8847), length 102:
MPLS (label 123, exp 0, [S], ttl 255)
        (tos 0x0, ttl 64, id 28269, offset 0, flags [DF], proto ICMP (1), length 84)
    192.168.1.120 > 192.168.2.123: ICMP echo request, id 18, seq 62, length 64
11:34:59.750210 08:00:27:61:de:17 > 08:00:27:4f:d1:59, ethertype IPv4 (0x0800), length 98: (tos
0x0, ttl 63, id 62470, offset 0, flags [none], proto ICMP (1), length 84)
```

10. Các gói tin ICMP gửi ra & vào kết nối enp0s3 giữa R2 với H2 hoàn toàn không có MPLS. Như vậy là với luồng MPLS label 123, R2 đã xử lý như một egress ở chiều kết nối R2 ➔ H2:

```
R2:~$ sudo tcpdump -i enp0s3 -env
tcpdump: listening on enp0s3, link-type EN10MB (Ethernet), capture size 262144 bytes
11:39:54.179109 08:00:27:1c:c2:15 > 08:00:27:75:25:d1, ethertype IPv4 (0x0800), length 98: (tos
0x0, ttl 64, id 64644, offset 0, flags [DF], proto ICMP (1), length 84)
    192.168.1.120 > 192.168.2.123: ICMP echo request, id 18, seq 356, length 64
11:39:54.179699 08:00:27:75:25:d1 > 08:00:27:87:0c:4c, ethertype IPv4 (0x0800), length 98: (tos
0x0, ttl 64, id 33076, offset 0, flags [none], proto ICMP (1), length 84)
    192.168.2.123 > 192.168.1.120: ICMP echo reply, id 18, seq 356, length 64
11:39:55.180384 08:00:27:1c:c2:15 > 08:00:27:75:25:d1, ethertype IPv4 (0x0800), length 98: (tos
0x0, ttl 64, id 64845, offset 0, flags [DF], proto ICMP (1), length 84)
    192.168.1.120 > 192.168.2.123: ICMP echo request, id 18, seq 357, length 64
```

```
11:39:55.180992 08:00:27:75:25:d1 > 08:00:27:87:0c:4c, ethertype IPv4 (0x0800), length 98: (tos
0x0, ttl 64, id 33291, offset 0, flags [none], proto ICMP (1), length 84)
    192.168.2.123 > 192.168.1.120: ICMP echo reply, id 18, seq 357, length 64
11:39:56.181558 08:00:27:1c:c2:15 > 08:00:27:75:25:d1, ethertype IPv4 (0x0800), length 98: (tos
0x0, ttl 64, id 64976, offset 0, flags [DF], proto ICMP (1), length 84)
```

3.2 LS Path by connecting LSR

11. Thiết lập thêm filter để xử lý luồng H1 → R1 → R2 → R3 → H3. Chèn MPLS vào với label 345

```
R1:~$ sudo tc filter add dev enp0s10 protocol ip parent ffff: \
        flower dst_ip 192.168.3.0/24 \
                action mpls push protocol mpls_uc label 456 \
                action skbmod set dmac 08:00:27:61:de:17 \
                action mirred egress redirect dev enp0s9

R1:~$ sudo tc filter show dev enp0s10 ingress
filter parent ffff: protocol ip pref 49151 flower chain 0
filter parent ffff: protocol ip pref 49151 flower chain 0 handle 0x1
  eth_type ipv4
  dst_ip 192.168.3.0/24
  not_in_hw
        action order 1: mpls  push protocol mpls_uc label 456 ttl 255 pipe
         index 2 ref 1 bind 1

        action order 2: skbmod pipe set dmac 08:00:27:61:de:17
         index 2 ref 1 bind 1

        action order 3: mirred (Egress Redirect to device enp0s9) stolen
        index 2 ref 1 bind 1

filter parent ffff: protocol ip pref 49152 flower chain 0
filter parent ffff: protocol ip pref 49152 flower chain 0 handle 0x1
  eth_type ipv4
  dst_ip 192.168.2.0/24
  not_in_hw
        action order 1: mpls  push protocol mpls_uc label 123 ttl 255 pipe
         index 1 ref 1 bind 1

        action order 2: skbmod pipe set dmac 08:00:27:61:de:17
         index 1 ref 1 bind 1

        action order 3: mirred (Egress Redirect to device enp0s9) stolen
        index 1 ref 1 bind 1
```

12. Thiết lập thêm filter để xử lý luồng H1 → R1 → R2 → R3 → H3 trên R2.

```
R2:~$ sudo tc filter add dev enp0s9 ingress \
        protocol mpls_uc flower mpls_label 456 \
                action skbmod set dmac 08:00:27:4a:97:32 \
                action mirred egress redirect dev enp0s10

R2:~$ sudo tc filter show dev enp0s9 ingress
filter protocol mpls_uc pref 49150 flower chain 0
filter protocol mpls_uc pref 49150 flower chain 0 handle 0x1
  eth_type 8847
  mpls_label 456
  not_in_hw
        action order 1: skbmod pipe set dmac 08:00:27:4a:97:32
         index 3 ref 1 bind 1

        action order 2: mirred (Egress Redirect to device enp0s10) stolen
        index 3 ref 1 bind 1

filter protocol mpls_uc pref 49151 flower chain 0
filter protocol mpls_uc pref 49151 flower chain 0 handle 0x1
  eth_type 8847
  mpls_label 123
  not_in_hw
        action order 1: mpls  pop protocol ip pipe
         index 1 ref 1 bind 1
```

```
        action order 2: skbmod pipe set dmac 08:00:27:75:25:d1
         index 2 ref 1 bind 1

        action order 3: mirred (Egress Redirect to device enp0s3) stolen
        index 2 ref 1 bind 1
```

13. Thiết lập thêm filter để xử lý luồng H1 → R1 → R2 → R3 → H3 trên R3.

```
R3:~$ sudo tc qdisc add dev enp0s9 handle ffff: ingress
R3:~$ sudo tc filter add dev enp0s9 ingress \
        protocol mpls_uc flower mpls_label 456 \
                action mpls pop protocol ipv4 action \
                skbmod set dmac 08:00:27:10:11:15 \
                action mirred egress redirect dev enp0s3
```

14. Luồng LSP H1 → R1 → R2 → R3 → H3 đã được thiết lập. Ping thành công

```
H1:~$ ping 192.168.3.115
PING 192.168.3.115 (192.168.3.115) 56(84) bytes of data.
64 bytes from 192.168.3.115: icmp_seq=1 ttl=61 time=4.01 ms
64 bytes from 192.168.3.115: icmp_seq=2 ttl=61 time=2.47 ms
```

15. Gói tin ICMP Echo Request được gửi bằng MPLS label 456, gói tin ICMP Echo Reply trả về bằng IP routing:

```
R2:~$ sudo tcpdump -i enp0s9 -env
tcpdump: listening on enp0s9, link-type EN10MB (Ethernet), capture size 262144 bytes
16:26:22.431816 08:00:27:1c:c2:15 > 08:00:27:61:de:17, ethertype MPLS unicast (0x8847), length 102:
MPLS (label 456, exp 0, [S], ttl 255)
        (tos 0x0, ttl 64, id 32117, offset 0, flags [DF], proto ICMP (1), length 84)
    192.168.1.120 > 192.168.3.115: ICMP echo request, id 6, seq 1, length 64
16:26:22.433124 08:00:27:61:de:17 > 08:00:27:4f:d1:59, ethertype IPv4 (0x0800), length 98: (tos
0x0, ttl 62, id 38947, offset 0, flags [none], proto ICMP (1), length 84)
    192.168.3.115 > 192.168.1.120: ICMP echo reply, id 6, seq 1, length 64
16:26:23.433912 08:00:27:1c:c2:15 > 08:00:27:61:de:17, ethertype MPLS unicast (0x8847), length 102:
MPLS (label 456, exp 0, [S], ttl 255)
        (tos 0x0, ttl 64, id 32164, offset 0, flags [DF], proto ICMP (1), length 84)
    192.168.1.120 > 192.168.3.115: ICMP echo request, id 6, seq 2, length 64
16:26:23.435166 08:00:27:61:de:17 > 08:00:27:4f:d1:59, ethertype IPv4 (0x0800), length 98: (tos
0x0, ttl 62, id 39172, offset 0, flags [none], proto ICMP (1), length 84)
    192.168.3.115 > 192.168.1.120: ICMP echo reply, id 6, seq 2, length 64
16:26:24.436079 08:00:27:1c:c2:15 > 08:00:27:61:de:17, ethertype MPLS unicast (0x8847), length 102:
MPLS (label 456, exp 0, [S], ttl 255)
        (tos 0x0, ttl 64, id 32340, offset 0, flags [DF], proto ICMP (1), length 84)
    192.168.1.120 > 192.168.3.115: ICMP echo request, id 6, seq 3, length 64
16:26:24.437342 08:00:27:61:de:17 > 08:00:27:4f:d1:59, ethertype IPv4 (0x0800), length 98: (tos
0x0, ttl 62, id 39245, offset 0, flags [none], proto ICMP (1), length 84)
    192.168.3.115 > 192.168.1.120: ICMP echo reply, id 6, seq 3, length 64
```

16. Xử lý tương tự để tạo LSP H1 → R1 → R2 → R4→ H4
17. Thiết lập thêm filter để xử lý luồng H1 → R1 → R2 → R3 → H3. Chèn MPLS vào với label 345

3.3 MPLS by Linux kernel: static label setting

1. Bật module MPLS trong linux kernel (có thể đưa *mpls_router* vào */etc/modules* để load khi boot):

```
~$ sudo modprobe mpls_router
~$ lsmod | grep mpls
mpls_router            40960  0
ip_tunnel              24576  1 mpls_router

~$ sudo sysctl -a --pattern mpls
net.mpls.conf.enp0s10.input = 0
net.mpls.conf.enp0s3.input = 0
net.mpls.conf.enp0s8.input = 0
net.mpls.conf.enp0s9.input = 0
net.mpls.conf.lo.input = 0
net.mpls.default_ttl = 255
net.mpls.ip_ttl_propagate = 1
```

```
net.mpls.platform_labels = 0
```

2. Thiết lập các tham số MPLS cho Linux kernel:

```
~$ sudo nano /etc/sysctl.conf
net.mpls.conf.enp0s9.input=1
net.mpls.conf.enp0s3.input=1
net.mpls.platform_labels=10000

~$ sudo sysctl –system

~$ sudo sysctl net.mpls
net.mpls.conf.enp0s10.input = 0
net.mpls.conf.enp0s3.input = 1
net.mpls.conf.enp0s8.input = 0
net.mpls.conf.enp0s9.input = 1
net.mpls.conf.lo.input = 0
net.mpls.default_ttl = 255
net.mpls.ip_ttl_propagate = 1
net.mpls.platform_labels = 10000
```

3. R4 = ingress LSR cho LS path H4→H1. Khai báo áp dụng MPLS label 100 cho kết nối đến mạng 192.168.1.0/24 và chuyển tiếp gói tin đến (next hope) 10.10.3.1:

```
~$ sudo ip route add 192.168.1.0/24 encap mpls 100 via inet 10.10.3.1
~$ sudo ip route
default via 10.0.2.2 dev enp0s3 proto dhcp src 10.0.2.15 metric 100
10.0.2.0/24 dev enp0s3 proto kernel scope link src 10.0.2.15
10.0.2.2 dev enp0s3 proto dhcp scope link src 10.0.2.15 metric 100
10.10.3.0/24 dev enp0s9 proto kernel scope link src 10.10.3.2
192.168.1.0/24  encap mpls  100 via 10.10.3.1 dev enp0s9
```

4. Khi R4 đóng vai trò là ingress LSR, ping từ H4 đến H1 và bắt gói tin trên kết nối R2-R4 thấy xuất hiện gói in ICMP Echo Request được gắn MPLS label 100:

```
@H4:~$ ping 192.168.1.120
PING 192.168.1.120 (192.168.1.120) 56(84) bytes of data...

R2:~$ sudo tcpdump -i enp0s10 -envv
tcpdump: listening on enp0s10, link-type EN10MB (Ethernet), capture size 262144 bytes
20:08:45.316786 08:00:27:15:96:5b > 08:00:27:f9:af:90, ethertype MPLS unicast (0x8847), length 102:
MPLS (label 100, exp 0, [S], ttl 63)
        (tos 0x0, ttl 63, id 24755, offset 0, flags [DF], proto ICMP (1), length 84)
    192.168.4.130 > 192.168.1.120: ICMP echo request, id 6, seq 1, length 64
```

5. Cấu hình R2 là một LSR trong MPLS network (network 192.168.1.0/24 không được khai báo trong bảng routing table, chỉ có MPLS switching với label 100 thì chuyển sang 200 trên kết nối 10.10.1.1 – là R1):

```
R2:~$ sudo ip -f mpls route add 100 as 200 via inet 10.10.1.1
R2:~$ sudo ip -f mpls route
100 as to 200 via inet 10.10.1.1 dev enp0s9

R2:~$ sudo ip route
10.10.1.0/24 dev enp0s9 proto kernel scope link src 10.10.1.2
10.10.2.0/24 dev enp0s3 proto kernel scope link src 10.10.2.1
10.10.3.0/24 dev enp0s10 proto kernel scope link src 10.10.3.1
```

6. Bắt gói tin trên kết nối R1-R2 nhìn thấy gói tin ICMP Echo Request gửi từ H4 đến H1:

```
R1:~$ sudo tcpdump -i enp0s9 -envv
tcpdump: listening on enp0s9, link-type EN10MB (Ethernet), capture size 262144 bytes
20:14:30.438059 08:00:27:61:de:17 > 08:00:27:4f:d1:59, ethertype MPLS unicast (0x8847), length 102:
MPLS (label 200, exp 0, [S], ttl 62)
        (tos 0x0, ttl 63, id 2665, offset 0, flags [DF], proto ICMP (1), length 84)
    192.168.4.130 > 192.168.1.120: ICMP echo request, id 6, seq 339, length 64
```

7. R1 = egress LSR cho LS path H4→H1. Switch gói tin MPLS label 200 đến H1 và không đặt MPLS label nữa:

```
R1:~$ sudo ip -f mpls route add 200 via inet 192.168.1.120
R1:~$ sudo ip -f mpls route
200 via inet 192.168.1.120 dev enp0s10

R1:~$ sudo ip route
10.10.1.0/24 dev enp0s9 proto kernel scope link src 10.10.1.1
192.168.1.0/24 dev enp0s10 proto kernel scope link src 192.168.1.1
```

8. Bắt gói tin trên kết nối H1-R1, đã thấy nhận được ICMP Echo Request và không còn được "bọc" trong gói tin MPLS nữa. H1 nhận được ICMP Echo Request thì xử lý gửi ICMP Echo Reply:

```
H1:~$ sudo tcpdump -i enp0s9 -env
tcpdump: listening on enp0s9, link-type EN10MB (Ethernet), capture size 262144 bytes
20:37:13.459507 08:00:27:14:05:7e > 08:00:27:1c:c2:15, ethertype IPv4 (0x0800), length 98: (tos
0x0, ttl 61, id 40786, offset 0, flags [DF], proto ICMP (1), length 84)
    192.168.4.130 > 192.168.1.120: ICMP echo request, id 6, seq 1669, length 64
20:37:13.459553 08:00:27:1c:c2:15 > 08:00:27:14:05:7e, ethertype IPv4 (0x0800), length 98: (tos
0x0, ttl 64, id 20488, offset 0, flags [none], proto ICMP (1), length 84)
    192.168.1.120 > 192.168.4.130: ICMP echo reply, id 6, seq 1669, length 64
```

9. R1 nhận được ICMP Echo Reply từ H1 gửi cho H4. Thiết lập các cấu hình MPLS tương tự trên các router $1, R2, R4 để tạo LS path từ H1→H4.

```
R1:~$ sudo ip route add 192.168.4.0/24 encap mpls 300 via inet 10.10.1.2
R1:~$ sudo ip route
10.10.1.0/24 dev enp0s9 proto kernel scope link src 10.10.1.1
192.168.1.0/24 dev enp0s10 proto kernel scope link src 192.168.1.1
192.168.4.0/24  encap mpls  300 via 10.10.1.2 dev enp0s9

R2:~$ sudo ip -f mpls route add 300 as 400 via inet 10.10.3.2
R2:~$ sudo ip -f mpls route
100 as to 200 via inet 10.10.1.1 dev enp0s9
300 as to 400 via inet 10.10.3.2 dev enp0s10

R4:~$ sudo ip -f mpls route add 400 via inet 192.168.4.130
R4:~$ sudo ip -f mpls route
400 via inet 192.168.4.130 dev enp0s10

H4:~$ ping 192.168.1.120
PING 192.168.1.120 (192.168.1.120) 56(84) bytes of data.
64 bytes from 192.168.1.120: icmp_seq=149 ttl=61 time=2.66 ms
64 bytes from 192.168.1.120: icmp_seq=150 ttl=61 time=2.45 ms
64 bytes from 192.168.1.120: icmp_seq=151 ttl=61 time=2.64 ms
```

10. Có thể tạo LS path H4→R4→R2→R3→H3 mà R4 là ingress, R2 là egress. Khi H4 ping H3 thì gói tin MPLS (ICMP Echo Request) chuyển từ R2 đến R3 đã được chuyển thành gói IP thuần túy (không có MPLS layer nữa do R2 đã xử lý egress)

```
R4:~$ sudo ip route add 192.168.3.0/24 encap mpls 103 via inet 10.10.3.1
R4:~$ ip route
10.10.3.0/24 dev enp0s9 proto kernel scope link src 10.10.3.2
192.168.1.0/24  encap mpls  100 via 10.10.3.1 dev enp0s9
192.168.3.0/24  encap mpls  103 via 10.10.3.1 dev enp0s9
192.168.4.0/24 dev enp0s10 proto kernel scope link src 192.168.4.1

R2:~$ sudo ip -f mpls route add 103 via inet 10.10.2.2
R2:~$ ip -f mpls route
100 as to 200 via inet 10.10.1.1 dev enp0s9
103 via inet 10.10.2.2 dev enp0s3
300 as to 400 via inet 10.10.3.2 dev enp0s10

R2:~$ sudo tcpdump -i enp0s10 -env
tcpdump: listening on enp0s10, link-type EN10MB (Ethernet), capture size 262144 bytes
09:36:02.854222 08:00:27:15:96:5b > 08:00:27:f9:af:90, ethertype MPLS unicast (0x8847), length 102:
MPLS (label 103, exp 0, [S], ttl 63)
        (tos 0x0, ttl 63, id 50129, offset 0, flags [DF], proto ICMP (1), length 84)
    192.168.4.130 > 192.168.3.1: ICMP echo request, id 11, seq 195, length 64
```

```
R3:~$ sudo tcpdump -i enp0s9 -env
tcpdump: listening on enp0s9, link-type EN10MB (Ethernet), capture size 262144 bytes
09:32:49.108301 08:00:27:87:0c:4c > 08:00:27:ea:db:c7, ethertype IPv4 (0x0800), length 98: (tos
0x0, ttl 62, id 23828, offset 0, flags [DF], proto ICMP (1), length 84)
    192.168.4.130 > 192.168.3.1: ICMP echo request, id 11, seq 9, length 64
```

3.4 Dynamic Label Distribution (LDP)

1. Tìm hiểu Hello messages trường hợp Basic Discovery. R1 và R2 kết nối trực tiếp, start FRR service & khai báo các thông số LDP cần thiết. Bắt gói tin UDP cổng 646 trên kết nối giữa 2 router để xem gói message LDP Hello:

```
R1# show running-config

frr version 7.2.1
mpls ldp
 router-id 1.1.1.1
 !
 address-family ipv4
  discovery transport-address 10.10.1.1
  !
  interface enp0s9
  !
 exit-address-family
 !
!
line vty
!
End

R2# show running-config

frr version 7.2.1
mpls ldp
 router-id 2.2.2.2
 !
 address-family ipv4
  discovery transport-address 10.10.1.2
  !
  interface enp0s9
  !
 exit-address-family
 !
!
line vty
!
End


@R1:~$ sudo tcpdump -i enp0s9 -env udp port 646
tcpdump: listening on enp0s9, link-type EN10MB (Ethernet), capture size 262144 bytes
14:52:08.553043 08:00:27:4f:d1:59 > 01:00:5e:00:00:02, ethertype IPv4 (0x0800), length 84: (tos
0xc0, ttl 1, id 51824, offset 0, flags [DF], proto UDP (17), length 70)
    10.10.1.1.646 > 224.0.0.2.646:
        LDP, Label-Space-ID: 1.1.1.1:0, pdu-length: 38
          Hello Message (0x0100), length: 28, Message ID: 0x00000001, Flags: [ignore if unknown]
            Common Hello Parameters TLV (0x0400), length: 4, Flags: [ignore and don't forward if
unknown]
              Hold Time: 15s, Flags: [Link Hello]
            IPv4 Transport Address TLV (0x0401), length: 4, Flags: [ignore and don't forward if
unknown]
              IPv4 Transport Address: 10.10.1.1
            Configuration Sequence Number TLV (0x0402), length: 4, Flags: [ignore and don't forward
if unknown]
              Sequence Number: 7
14:52:13.557950 08:00:27:4f:d1:59 > 01:00:5e:00:00:02, ethertype IPv4 (0x0800), length 84: (tos
0xc0, ttl 1, id 51991, offset 0, flags [DF], proto UDP (17), length 70)

14:52:15.963824 08:00:27:61:de:17 > 01:00:5e:00:00:02, ethertype IPv4 (0x0800), length 84: (tos
0xc0, ttl 1, id 62530, offset 0, flags [DF], proto UDP (17), length 70)
```

```
    10.10.1.2.646 > 224.0.0.2.646:
        LDP, Label-Space-ID: 2.2.2.2:0, pdu-length: 38
          Hello Message (0x0100), length: 28, Message ID: 0x00000001, Flags: [ignore if unknown]
            Common Hello Parameters TLV (0x0400), length: 4, Flags: [ignore and don't forward if
unknown]
              Hold Time: 15s, Flags: [Link Hello]
            IPv4 Transport Address TLV (0x0401), length: 4, Flags: [ignore and don't forward if
unknown]
              IPv4 Transport Address: 10.10.1.2
            Configuration Sequence Number TLV (0x0402), length: 4, Flags: [ignore and don't forward
if unknown]
              Sequence Number: 7

R1# show mpls ldp neighbor
AF   ID              State       Remote Address    Uptime
ipv4 2.2.2.2         OPERATIONAL 10.10.1.2       00:03:19

R2# show mpls ldp neighbor
AF   ID              State       Remote Address    Uptime
ipv4 1.1.1.1         OPERATIONAL 10.10.1.1       00:03:44
```

2. LDP Sesssion với TCP. Sau khi phát hiện láng giềng, LDP session được thiết lập giữa 2 router (TCP port 646) nhưng không có thông tin LDP gì được trao đổi, chỉ thấy các gói tin TCP "bắt tay":

```
R1:~$ sudo tcpdump -i enp0s9 -env tcp port 646
tcpdump: listening on enp0s9, link-type EN10MB (Ethernet), capture size 262144 bytes
14:55:18.569470 08:00:27:4f:d1:59 > 08:00:27:61:de:17, ethertype IPv4 (0x0800), length 84: (tos
0xc0, ttl 255, id 13207, offset 0, flags [DF], proto TCP (6), length 70)
    10.10.1.1.646 > 10.10.1.2.39721: Flags [P.], cksum 0x164f (incorrect -> 0xe5c8), seq
3278399183:3278399201, ack 554703484, win 509, options [nop,nop,TS val 2394302506 ecr 1056144185],
length 18
14:55:18.570423 08:00:27:61:de:17 > 08:00:27:4f:d1:59, ethertype IPv4 (0x0800), length 66: (tos
0x0, ttl 255, id 57688, offset 0, flags [DF], proto TCP (6), length 52)
    10.10.1.2.39721 > 10.10.1.1.646: Flags [.], cksum 0xffba (correct), ack 18, win 502, options
[nop,nop,TS val 1056204187 ecr 2394302506], length 0
14:55:18.570897 08:00:27:61:de:17 > 08:00:27:4f:d1:59, ethertype IPv4 (0x0800), length 84: (tos
0x0, ttl 255, id 57689, offset 0, flags [DF], proto TCP (6), length 70)
    10.10.1.2.39721 > 10.10.1.1.646: Flags [P.], cksum 0xf957 (correct), seq 1:19, ack 18, win 502,
options [nop,nop,TS val 1056204188 ecr 2394302506], length 18
14:55:18.570925 08:00:27:4f:d1:59 > 08:00:27:61:de:17, ethertype IPv4 (0x0800), length 66: (tos
0xc0, ttl 255, id 13208, offset 0, flags [DF], proto TCP (6), length 52)
    10.10.1.1.646 > 10.10.1.2.39721: Flags [.], cksum 0x163d (incorrect -> 0xff9e), ack 19, win
509, options [nop,nop,TS val 2394302508 ecr 1056204188], length 0
14:56:18.571591 08:00:27:4f:d1:59 > 08:00:27:61:de:17, ethertype IPv4 (0x0800), length 84: (tos
0xc0, ttl 255, id 13209, offset 0, flags [DF], proto TCP (6), length 70)
    10.10.1.1.646 > 10.10.1.2.39721: Flags [P.], cksum 0x164f (incorrect -> 0x10d1), seq 18:36, ack
19, win 509, options [nop,nop,TS val 2394362508 ecr 1056204188], length 18
14:56:18.572376 08:00:27:61:de:17 > 08:00:27:4f:d1:59, ethertype IPv4 (0x0800), length 66: (tos
0x0, ttl 255, id 57690, offset 0, flags [DF], proto TCP (6), length 52)
```

Lý do là FRR (phiên bản 7.2.1) cài đặt phương pháp trao đổi nhãn LDP theo thuật toán "Liberal Label Retention + Downstream Unsolicited + Independent Control" [1]. Downstream Unsolicited có nghĩa là chỉ khi downstream router phát hiện "next hope" (bằng IGP) cho một network nào đó thì sẽ dùng LDP session (TCP connection) yêu cầu upstream router chấp nhận LDB label cho FEC tương ứng với network này.

Sử dụng OSPF (hay IGP nào cũng được) để cấu hình cho R1 và R2. R2 (là downstream router) phát hiện đường đi đến 192.168.1.0/24 qua next hope R1, R2 tự gán nhãn 16 cho FEC network 192.168.1.0/24 và thông báo cho R1 trong LDP session:

```
R1# configure terminal
R1(config)# router ospf
R1(config-router)# network 192.168.1.0/24 area 0
R1(config-router)# network 10.10.1.0/24 area 0
R1(config-router)# end
```

[1] http://docs.frrouting.org/en/latest/ldpd.html

```
R2# configure terminal
R2(config)# router ospf
R2(config-router)# network 10.10.1.0/24 area 0
R2(config-router)# end

R2# show ip route
O   10.10.1.0/24 [110/100] is directly connected, enp0s9, 00:00:16
C>* 10.10.1.0/24 is directly connected, enp0s9, 00:16:46
O>* 192.168.1.0/24 [110/200] via 10.10.1.1, enp0s9, label implicit-null, 00:00:06

R1:~$ sudo tcpdump -i enp0s9 -env tcp port 646
15:10:43.597230 08:00:27:61:de:17 > 08:00:27:4f:d1:59, ethertype IPv4 (0x0800), length 103: (tos
0x0, ttl 255, id 36599, offset 0, flags [DF], proto TCP (6), length 89)
    10.10.1.2.39835 > 10.10.1.1.646: Flags [P.], cksum 0xef35 (correct), seq 255:292, ack 228, win
502, options [nop,nop,TS val 1057129214 ecr 2395218550], length 37
        LDP, Label-Space-ID: 2.2.2.2:0, pdu-length: 33
          Label Mapping Message (0x0400), length: 23, Message ID: 0x00000107, Flags: [ignore if
unknown]
            FEC TLV (0x0100), length: 7, Flags: [ignore and don't forward if unknown]
              Prefix FEC (0x02): IPv4 prefix 192.168.1.0/24
            Generic Label TLV (0x0200), length: 4, Flags: [ignore and don't forward if unknown]
              Label: 16
```

3. R2 áp dụng nhãn 16 cho LDP binding và MPLS table. R1 được thông báo nhãn 16 cho FEC này nhưng không áp dụng (cột Use):

```
R2# show mpls ldp binding
AF   Destination          Nexthop         Local Label Remote Label  In Use
ipv4 10.10.1.0/24         1.1.1.1         imp-null    imp-null          no
ipv4 192.168.1.0/24       1.1.1.1         16          imp-null         yes

R2# show mpls table
 Inbound                            Outbound
   Label     Type           Nexthop     Label
--------  -------  ---------------  --------
      16      LDP        10.10.1.1  implicit-null

R1# show mpls ldp binding
AF   Destination          Nexthop         Local Label Remote Label  In Use
ipv4 10.10.1.0/24         2.2.2.2         imp-null    imp-null          no
ipv4 192.168.1.0/24       2.2.2.2         imp-null    16                no
```

4. Do R2 sử dụng LDP và áp ụng nhãn 16 cho FEC 192.168.1.0/24 nên nó đưa vào bảng MPLS switching của kernel R2:

```
R2:~$ ip -f mpls route
16 via inet 10.10.1.1 dev enp0s9 proto ldp
```

5. Có thể test nhãn 16 bằng cách gửi gói tin *ping* bất kỳ với nhãn 16 (sẽ được forward đến R1: 10.10.1.1 bất kể là ping đến network nào). Ví dụ: thiết lập R4 là một ingress LSR để forward sang R2 với nhãn 16 cho một FEC nào đó (20.20.20.20). Rồi thực hiện *ping* từ R4 đến địa chỉ này. Gói tin ICPM gửi từ R4 đến R2 được gán nhãn 16 nên được forward sang R1. Bắt gói tin trên kết nối R1-R2 thấy có ICMP Echo Request mà không gói trong MPLS (do R2 khi forward sang R1 đã bỏ MPLS layer đi):

```
R4:~$ sudo ip route add 20.20.20.20/32 encap mpls 16 via inet 10.10.3.1
R4:~$ sudo ip route
10.10.3.0/24 dev enp0s9 proto kernel scope link src 10.10.3.2
20.20.20.20  encap mpls  16 via 10.10.3.1 dev enp0s9
192.168.4.0/24 dev enp0s10 proto kernel scope link src 192.168.4.1

R4:~$ ping 20.20.20.20
PING 20.20.20.20 (20.20.20.20) 56(84) bytes of data....

R2:~$ sudo tcpdump -i enp0s10 -env
tcpdump: listening on enp0s10, link-type EN10MB (Ethernet), capture size 262144 bytes
16:03:13.015172 08:00:27:15:96:5b > 08:00:27:f9:af:90, ethertype MPLS unicast (0x8847), length 102:
MPLS (label 16, exp 0, [S], ttl 64)
```

```
        (tos 0x0, ttl 64, id 24064, offset 0, flags [DF], proto ICMP (1), length 84)
    10.10.3.2 > 20.20.20.20: ICMP echo request, id 22, seq 197, length 64


R1:~$ sudo tcpdump -i enp0s9 -env icmp
tcpdump: listening on enp0s9, link-type EN10MB (Ethernet), capture size 262144 bytes
16:10:16.211272 08:00:27:61:de:17 > 08:00:27:4f:d1:59, ethertype IPv4 (0x0800), length 98: (tos
0x0, ttl 63, id 11067, offset 0, flags [DF], proto ICMP (1), length 84)
    10.10.3.2 > 20.20.20.20: ICMP echo request, id 22, seq 610, length 64
```

6. Bổ sung R3 vào MPLS network để tạo LS path R3→R2→R1. Cấu hình MPLS & OSPF R3 và cập nhật MPLS R2 để join được với nhau:

```
R3# show running-config

frr version 7.2.1
hostname R3
!
router ospf
 network 10.10.2.0/24 area 0
!
mpls ldp
 router-id 3.3.3.3
 !
 address-family ipv4
  discovery transport-address 10.10.2.2
  !
  interface enp0s9
  !
 exit-address-family
 !
!
line vty
!
end

R2# show running-config

frr version 7.2.1
hostname R2
router ospf
 network 10.10.1.0/24 area 0
 network 10.10.2.0/24 area 0
!
mpls ldp
 router-id 2.2.2.2
 !
 address-family ipv4
  discovery transport-address 10.10.1.2
  !
  interface enp0s3
  !
  interface enp0s9
  !
 exit-address-family
 !
!
line vty
!
end
```

7. R3 nhận được network 192.168.1.0/24 gửi từ R2 (bằng IGP OSPF) và gán label 17 cho network này. Đồng thời thiết lập MPLS switching rule chuyển sang label 16 khi forward cho R2:

```
R3# show mpls ldp binding
AF   Destination          Nexthop         Local Label Remote Label  In Use
ipv4 10.10.1.0/24         2.2.2.2         16          imp-null        yes
ipv4 10.10.2.0/24         2.2.2.2         imp-null    imp-null         no
ipv4 10.10.3.0/24         2.2.2.2         -           imp-null         no
ipv4 192.168.1.0/24       2.2.2.2         17          16              yes
ipv4 192.168.3.0/24       0.0.0.0         imp-null    -                no
```

(Cần thêm module mpls_iptunnel vào kernel để làm gì???)

LDP (FRR 7.2.1): There are different methods to send label advertisement modes. The implementation actually supports the following : Liberal Label Retention + Downstream Unsolicited + Independent Control. The other advertising modes are depicted below, and compared with the current implementation.

8. Theo IETF spec thì khai báo “discovery transport-address” (địa chỉ IP router sử dụng để kết nối LDP) là optional. Nếu không khai báo, router sẽ sử dụng địa chỉ IP gắn với thủ tục discovery (khai báo bằng “interface”). FRR lại yêu cầu phải khai báo “discovery transport-address” và chỉ join vào 224.0.0.2 chỉ khi có khai báo này:

```
# show running-config
Current configuration:
mpls ldp
 router-id 2.2.2.2
 !
 address-family ipv4
  ! Incomplete config, specify a discovery transport-address
  !
  interface enp0s3
  !
exit-address-family
 !
!
line vty
!
End

# exit
~$ sudo ip -4 maddress
1:      enp0s3
        inet  224.0.0.1

. . . .

# show  running-config
Current configuration:
mpls ldp
 router-id 2.2.2.2
 !
 address-family ipv4
  discovery transport-address 10.10.1.2
  !
  interface enp0s3
 exit-address-family
 !
!
line vty
!
End

# exit
~$ sudo ip -4 maddress
1:      enp0s3
        inet  224.0.0.2
        inet  224.0.0.1
```

inet 224.0.0.1

9. Cần khai báo “interface” và phải có “discovery transport-address” (không nhất thiết địa chỉ discovery transport-address phải cùng nằm trên interface) để enable LDP (gửi/nhận các gói tin

LDP theo địa chỉ multicast 224.0.0.2). Khi thiết lập cho interface nào thì sẽ thấy interface đó join vào địa chỉ 224.0.0.2:

```
~$ ip -4 maddress
1:      lo
        inet  224.0.0.1
2:      enp0s3
        inet  224.0.0.1
3:      enp0s8
        inet  224.0.0.1
4:      enp0s9
        inet  224.0.0.2
        inet  224.0.0.1
```

10.

```
~$ sudo nano /etc/frr/daemon
bgpd=no
ospfd=no
ospf6d=no
ripd=no
ripngd=no
isisd=no
pimd=no
ldpd=yes
nhrpd=no
eigrpd=no
babeld=no
sharpd=no
pbrd=no
bfdd=no
fabricd=no
vrrpd=no

~$ sudo service frr restart
~$ sudo service frr status
● frr.service - FRRouting
     Loaded: loaded (/lib/systemd/system/frr.service; disabled; vendor preset: enabled)
     Active: active (running) since Tue 2021-11-23 10:06:42 +07; 2min 53s ago
       Docs: https://frrouting.readthedocs.io/en/latest/setup.html
    Process: 1918 ExecStart=/usr/lib/frr/frrinit.sh start (code=exited, status=0/SUCCESS)
      Tasks: 11 (limit: 1071)
     Memory: 14.5M
     CGroup: /system.slice/frr.service
             ├─1941 /usr/lib/frr/watchfrr -d zebra ldpd staticd
             ├─1957 /usr/lib/frr/zebra -d -A 127.0.0.1 -s 90000000
             ├─1961 /usr/lib/frr/ldpd -L
             ├─1962 /usr/lib/frr/ldpd -E
             ├─1963 /usr/lib/frr/ldpd -d -A 127.0.0.1
             └─1967 /usr/lib/frr/staticd -d -A 127.0.0.1

Nov 23 10:06:42 R1 watchfrr[1941]: [EC 100663303] Forked background command [pid 1942]:
/usr/lib/frr/>
Nov 23 10:06:42 R1 watchfrr.sh[1951]: Cannot stop ldpd: pid file not found
Nov 23 10:06:42 R1 watchfrr.sh[1953]: Cannot stop zebra: pid file not found
Nov 23 10:06:42 R1 watchfrr.sh[1955]: Cannot stop staticd: pid file not found
Nov 23 10:06:42 R1 watchfrr[1941]: zebra state -> up : connect succeeded
Nov 23 10:06:42 R1 watchfrr[1941]: ldpd state -> up : connect succeeded
Nov 23 10:06:42 R1 watchfrr[1941]: staticd state -> up : connect succeeded
Nov 23 10:06:42 R1 watchfrr[1941]: all daemons up, doing startup-complete notify
Nov 23 10:06:42 R1 frrinit.sh[1918]:  * Started watchfrr
Nov 23 10:06:42 R1 systemd[1]: Started FRRouting.

~$ sudo vtysh
R1# configure terminal
R1(config)# mpls ldp

R1(config-ldp)# address-family ipv4
R1(config-ldp-af)# discovery transport-address
  A.B.C.D  IP address to be used as transport address
```

```
R1(config-ldp-af)# discovery transport-address 10.10.1.1
R1(config-ldp-af)# interface enp0s9
R1(config-ldp-af-if)# end
R1# write
Note: this version of vtysh never writes vtysh.conf
Building Configuration...
Warning: /etc/frr/frr.conf.sav unlink failed
Integrated configuration saved to /etc/frr/frr.conf

R1# show running-config
Building configuration...

Current configuration:
!
frr version 7.2.1
frr defaults traditional
hostname R1
no ipv6 forwarding
service integrated-vtysh-config
!
router-id 1.1.1.1
!
mpls ldp
 router-id 1.1.1.1
 !
 address-family ipv4
  discovery transport-address 10.10.1.1
  !
  interface enp0s9
  !
 exit-address-family
 !
!
line vty
!
end
```