Cairo University Faculty of Engineering
Dept. of Electronics and Electrical Communications Engineering

ELC 3080 Spring 2025

Computer Networks Project

Submitted to:

Dr. Khaled Fouad, Dr. Rania Osama

& Eng. Ahmed Atef

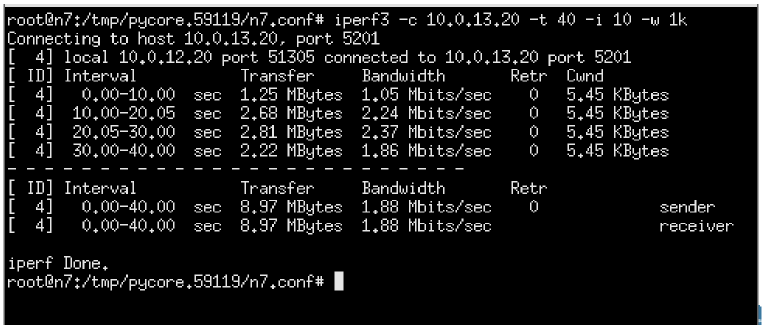
ID	BN	Sec	Name
9220028	6	1	احمد حسن محمد حسن
9220276	8	2	خالد أحمد حامد عبدالعزيز

2 - Effect of TCP Window Size

2-I Test iperf3 with window sizes: 1K, 2K, 3K, 4K, 5K, 6K, then 12K, 16K, 24K, and 32K

For Throughput, It's always equals $= \frac{\# Transfers}{Interval\ Time}$ = Bandwidth

window size 1k



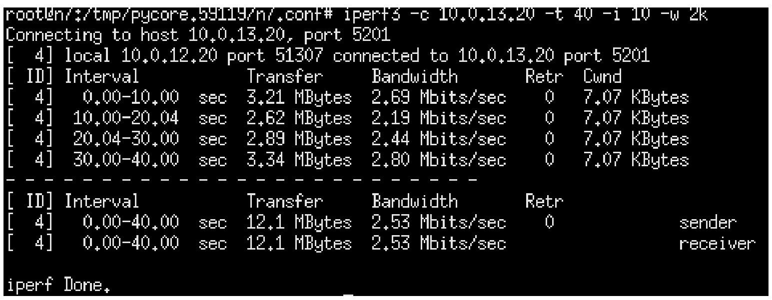
```
root@n7:/tmp/pycore.59119/n7.conf# iperf3 -c 10.0.13.20 -t 40 -i 10 -w 1k
Connecting to host 10.0.13.20, port 5201
[  4] local 10.0.12.20 port 51305 connected to 10.0.13.20 port 5201
[ ID] Interval           Transfer     Bandwidth       Retr  Cwnd
[  4]   0.00-10.00  sec  1.25 MBytes  1.05 Mbits/sec    0   5.45 KBytes
[  4]  10.00-20.05  sec  2.68 MBytes  2.24 Mbits/sec    0   5.45 KBytes
[  4]  20.05-30.00  sec  2.81 MBytes  2.37 Mbits/sec    0   5.45 KBytes
[  4]  30.00-40.00  sec  2.22 MBytes  1.86 Mbits/sec    0   5.45 KBytes
- - - - - - - - - - - - - - - - - - - - - - - - -
[ ID] Interval           Transfer     Bandwidth       Retr
[  4]   0.00-40.00  sec  8.97 MBytes  1.88 Mbits/sec    0             sender
[  4]   0.00-40.00  sec  8.97 MBytes  1.88 Mbits/sec                  receiver

iperf Done.
root@n7:/tmp/pycore.59119/n7.conf#
```

Figure 1: Testing iperf3 with Window size 1K

$\text{Throughput} = Bandwidth = \frac{2^{20}\times 8.97\times 8}{10^6\times 40} = 1.8811\ \text{Mbits/s}$

Average number of retransmissions = 0

window size 2k



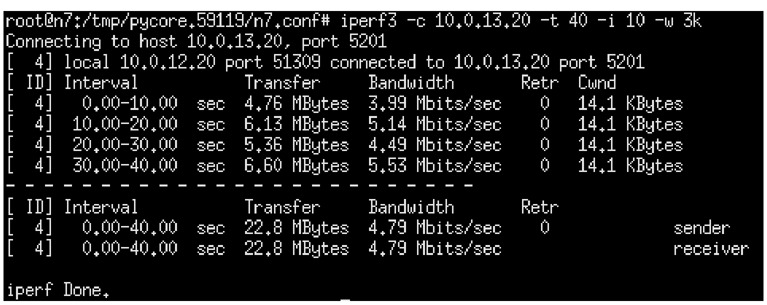
```
root@n7:/tmp/pycore.59119/n7.conf# iperf3 -c 10.0.13.20 -t 40 -i 10 -w 2k
Connecting to host 10.0.13.20, port 5201
[  4] local 10.0.12.20 port 51307 connected to 10.0.13.20 port 5201
[ ID] Interval           Transfer     Bandwidth       Retr  Cwnd
[  4]   0.00-10.00  sec  3.21 MBytes  2.69 Mbits/sec    0   7.07 KBytes
[  4]  10.00-20.04  sec  2.62 MBytes  2.19 Mbits/sec    0   7.07 KBytes
[  4]  20.04-30.00  sec  2.89 MBytes  2.44 Mbits/sec    0   7.07 KBytes
[  4]  30.00-40.00  sec  3.34 MBytes  2.80 Mbits/sec    0   7.07 KBytes
- - - - - - - - - - - - - - - - - - - - - - - - -
[ ID] Interval           Transfer     Bandwidth       Retr
[  4]   0.00-40.00  sec  12.1 MBytes  2.53 Mbits/sec    0             sender
[  4]   0.00-40.00  sec  12.1 MBytes  2.53 Mbits/sec                  receiver

iperf Done.
```

Figure 2: Testing iperf3 with Window size 2K

$\text{Throughput} = Bandwidth = \frac{2^{20}\times 12.1\times 8}{10^6\times 40} = 2.532\ \text{Mbits/s}$

Average number of retransmissions = 0

window size 3k



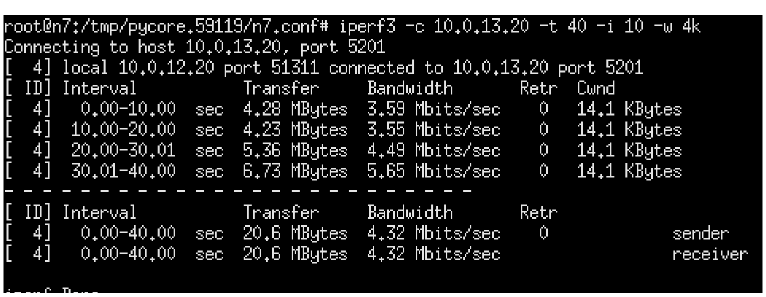
```
root@n7:/tmp/pycore.59119/n7.conf# iperf3 -c 10.0.13.20 -t 40 -i 10 -w 3k
Connecting to host 10.0.13.20, port 5201
[  4] local 10.0.12.20 port 51309 connected to 10.0.13.20 port 5201
[ ID] Interval           Transfer     Bandwidth       Retr  Cwnd
[  4]   0.00-10.00  sec  4.76 MBytes  3.99 Mbits/sec    0   14.1 KBytes
[  4]  10.00-20.00  sec  6.13 MBytes  5.14 Mbits/sec    0   14.1 KBytes
[  4]  20.00-30.00  sec  5.36 MBytes  4.49 Mbits/sec    0   14.1 KBytes
[  4]  30.00-40.00  sec  6.60 MBytes  5.53 Mbits/sec    0   14.1 KBytes
- - - - - - - - - - - - - - - - - - - - - - - - -
[ ID] Interval           Transfer     Bandwidth       Retr
[  4]   0.00-40.00  sec  22.8 MBytes  4.79 Mbits/sec    0             sender
[  4]   0.00-40.00  sec  22.8 MBytes  4.79 Mbits/sec                  receiver

iperf Done.
```

Figure 3: Testing iperf3 with Window size 3K

$\text{Throughput} = Bandwidth = \frac{2^{20}\times 22.8\times 8}{10^6\times 40} = 4.7815\ \text{Mbits/s}$

Average number of retransmissions = 0

window size 4k



```
root@n7:/tmp/pycore.59119/n7.conf# iperf3 -c 10.0.13.20 -t 40 -i 10 -w 4k
Connecting to host 10.0.13.20, port 5201
[  4] local 10.0.12.20 port 51311 connected to 10.0.13.20 port 5201
[ ID] Interval           Transfer     Bandwidth       Retr  Cwnd
[  4]   0.00-10.00  sec  4.28 MBytes  3.59 Mbits/sec    0   14.1 KBytes
[  4]  10.00-20.00  sec  4.23 MBytes  3.55 Mbits/sec    0   14.1 KBytes
[  4]  20.00-30.01  sec  5.36 MBytes  4.49 Mbits/sec    0   14.1 KBytes
[  4]  30.01-40.00  sec  6.73 MBytes  5.65 Mbits/sec    0   14.1 KBytes
- - - - - - - - - - - - - - - - - - - - - - - - -
[ ID] Interval           Transfer     Bandwidth       Retr
[  4]   0.00-40.00  sec  20.6 MBytes  4.32 Mbits/sec    0             sender
[  4]   0.00-40.00  sec  20.6 MBytes  4.32 Mbits/sec                  receiver
```

Figure 4: Testing iperf3 with Window size 4K

$\text{Throughput} = Bandwidth = \frac{2^{20}\times 20.6\times 8}{10^6\times 40} = 4.3201\ \text{Mbits/s}$

Average number of retransmissions = 0

window size 5k

```
root@n7:/tmp/pycore.59119/n7.conf# iperf3 -c 10.0.13.20 -t 40 -i 10 -w 5k
Connecting to host 10.0.13.20, port 5201
[  4] local 10.0.12.20 port 51313 connected to 10.0.13.20 port 5201
[ ID] Interval           Transfer     Bandwidth       Retr  Cwnd
[  4]   0.00-10.00  sec  5.96 MBytes  5.00 Mbits/sec    0   14.1 KBytes
[  4]  10.00-20.03  sec  5.11 MBytes  4.27 Mbits/sec    0   14.1 KBytes
[  4]  20.03-30.00  sec  4.03 MBytes  3.39 Mbits/sec    0   14.1 KBytes
[  4]  30.00-40.00  sec  5.09 MBytes  4.27 Mbits/sec    1   8.48 KBytes
- - - - - - - - - - - - - - - - - - - - - - - - -
[ ID] Interval           Transfer     Bandwidth       Retr
[  4]   0.00-40.00  sec  20.2 MBytes  4.23 Mbits/sec    1             sender
[  4]   0.00-40.00  sec  20.2 MBytes  4.23 Mbits/sec                  receiver

iperf Done.
```

Figure 5: Testing iperf3 with Window size 5K

Throughput $= Bandwidth = \frac{2^{20}\times 20.2\times 8}{10^{6}\times 40} = 4.236$ Mbits/s

Average number of retransmissions = 1

window size 6k

```
root@n7:/tmp/pycore.59119/n7.conf# iperf3 -c 10.0.13.20 -t 40 -i 10 -w 6k
Connecting to host 10.0.13.20, port 5201
[  4] local 10.0.12.20 port 51315 connected to 10.0.13.20 port 5201
[ ID] Interval           Transfer     Bandwidth       Retr  Cwnd
[  4]   0.00-10.00  sec   154 KBytes   126 Kbits/sec   86   4.24 KBytes
[  4]  10.00-20.00  sec   143 KBytes   117 Kbits/sec   88   4.24 KBytes
[  4]  20.00-30.00  sec   144 KBytes   118 Kbits/sec   87   4.24 KBytes
[  4]  30.00-40.00  sec   143 KBytes   117 Kbits/sec   87   4.24 KBytes
- - - - - - - - - - - - - - - - - - - - - - - - -
[ ID] Interval           Transfer     Bandwidth       Retr
[  4]   0.00-40.00  sec   584 KBytes   120 Kbits/sec  348             sender
[  4]   0.00-40.00  sec   574 KBytes   118 Kbits/sec                  receiver

iperf Done.
```

Figure 6: Testing iperf3 with Window size 6K

Throughput $= Bandwidth = \frac{2^{10}\times 584\times 8}{10^{3}\times 40} = 119.6032$ Kbits/s

Average number of retransmissions = 348

window size 12k

```
root@n7:/tmp/pycore.59119/n7.conf# iperf3 -c 10.0.13.20 -t 40 -i 10 -w 12k
Connecting to host 10.0.13.20, port 5201
[  4] local 10.0.12.20 port 51319 connected to 10.0.13.20 port 5201
[ ID] Interval           Transfer     Bandwidth       Retr  Cwnd
[  4]   0.00-10.00  sec  1.77 MBytes  1.49 Mbits/sec  323   4.24 KBytes
[  4]  10.00-20.00  sec   749 KBytes   614 Kbits/sec  138   4.24 KBytes
[  4]  20.00-30.00  sec  2.32 MBytes  1.95 Mbits/sec  420   4.24 KBytes
[  4]  30.00-40.00  sec  1.87 MBytes  1.57 Mbits/sec  341   2.83 KBytes
- - - - - - - - - - - - - - - - - - - - - - - - -
[ ID] Interval           Transfer     Bandwidth       Retr
[  4]   0.00-40.00  sec  6.69 MBytes  1.40 Mbits/sec  1222            sender
[  4]   0.00-40.00  sec  6.67 MBytes  1.40 Mbits/sec                  receiver

iperf Done.
```

Figure 7: Testing iperf3 with Window size 12K

Throughput $= Bandwidth = \frac{2^{20}\times 6.69\times 8}{10^{6}\times 40} = 1.3701$ Mbits/s

Average number of retransmissions = 1222

window size 16k

```
root@n7:/tmp/pycore.59119/n7.conf# iperf3 -c 10.0.13.20 -t 40 -i 10 -w 16k
Connecting to host 10.0.13.20, port 5201
[  4] local 10.0.12.20 port 51321 connected to 10.0.13.20 port 5201
[ ID] Interval           Transfer     Bandwidth       Retr  Cwnd
[  4]   0.00-10.01  sec   574 KBytes   470 Kbits/sec  110   4.24 KBytes
[  4]  10.01-20.00  sec   389 KBytes   319 Kbits/sec  103   5.66 KBytes
[  4]  20.00-30.05  sec   276 KBytes   225 Kbits/sec  106   5.66 KBytes
[  4]  30.05-40.00  sec   283 KBytes   233 Kbits/sec  110   8.48 KBytes
- - - - - - - - - - - - - - - - - - - - - - - - -
[ ID] Interval           Transfer     Bandwidth       Retr
[  4]   0.00-40.00  sec  1.49 MBytes   312 Kbits/sec  429             sender
[  4]   0.00-40.00  sec  1.46 MBytes   307 Kbits/sec                  receiver

iperf Done.
```

Figure 8: Testing iperf3 with Window size 16K

Throughput $= Bandwidth = \frac{2^{20}\times 1.49\times 8}{10^{3}\times 40} = 312.475$ Kbits/s

Average number of retransmissions = 429

window size 24k

```
root@n7:/tmp/pycore.59119/n7.conf# iperf3 -c 10.0.13.20 -t 40 -i 10 -w 24k
Connecting to host 10.0.13.20, port 5201
[  4] local 10.0.12.20 port 51323 connected to 10.0.13.20 port 5201
[ ID] Interval           Transfer     Bandwidth       Retr  Cwnd
[  4]   0.00-10.00  sec   221 KBytes   181 Kbits/sec   82   8.48 KBytes
[  4]  10.00-20.05  sec   218 KBytes   178 Kbits/sec   83   7.07 KBytes
[  4]  20.05-30.00  sec   178 KBytes   147 Kbits/sec   72   5.66 KBytes
[  4]  30.00-40.04  sec   198 KBytes   162 Kbits/sec   80   5.66 KBytes
- - - - - - - - - - - - - - - - - - - - - - - - -
[ ID] Interval           Transfer     Bandwidth       Retr
[  4]   0.00-40.04  sec   814 KBytes   167 Kbits/sec  317             sender
[  4]   0.00-40.04  sec   772 KBytes   158 Kbits/sec                  receiver

iperf Done.
```

Figure 9: Testing iperf3 with Window size 24K

Throughput $= Bandwidth = \frac{2^{20}\times 814\times 8}{10^6\times 40} = 166.702$ Kbits/s

Average number of retransmissions = 317

window size 32k

```
root@n7:/tmp/pycore.59119/n7.conf# iperf3 -c 10.0.13.20 -t 40 -i 10 -w 32k
Connecting to host 10.0.13.20, port 5201
[  4] local 10.0.12.20 port 51329 connected to 10.0.13.20 port 5201
[ ID] Interval           Transfer     Bandwidth       Retr  Cwnd
[  4]   0.00-10.00  sec   194 KBytes   159 Kbits/sec   80   22.6 KBytes
[  4]  10.00-20.00  sec   188 KBytes   154 Kbits/sec   97   29.7 KBytes
[  4]  20.00-30.00  sec   242 KBytes   198 Kbits/sec  115   9.90 KBytes
[  4]  30.00-40.00  sec   161 KBytes   132 Kbits/sec   92   18.4 KBytes
- - - - - - - - - - - - - - - - - - - - - - - - -
[ ID] Interval           Transfer     Bandwidth       Retr
[  4]   0.00-40.00  sec   785 KBytes   161 Kbits/sec  384             sender
[  4]   0.00-40.00  sec   738 KBytes   151 Kbits/sec                  receiver

iperf Done.
```

Throughput $= Bandwidth = \frac{2^{10}\times 785\times 8}{10^3\times 40} = 160.768$ Kbits/s

Average number of retransmissions = 384

Figure 10: Testing iperf3 with Window size 32K

Average throughput and the Average number of retransmissions as a function of window size:

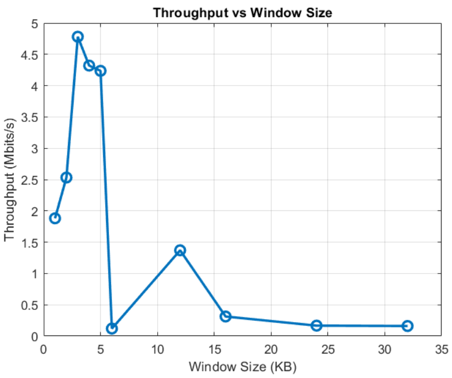


Figure 11: Throughput vs Window size plot

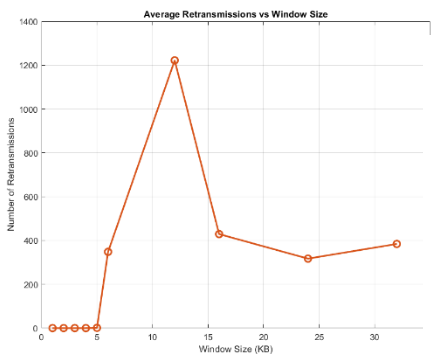


Figure 12: Avg. Retransmissions vs Window size plot

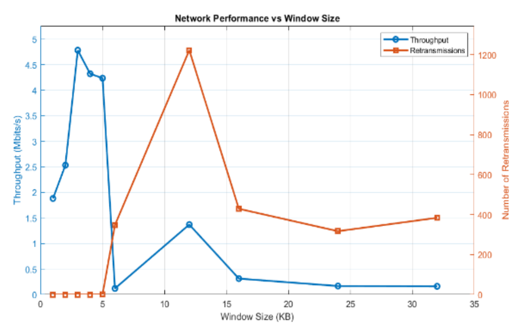


Figure 13: Network Performance vs Window size plot

Comment:

The test results show a Zig-Zag pattern in network speed and lost packets. When using small window sizes (1K-4K), everything works smoothly and data flows fast with no lost packets. But when we increase the window size to 5K or more, the receiver gets overwhelmed and starts losing data, which slows things down and retrasmissions. Sometimes (like at 12K), it looks like things improve briefly when some space opens up, but then the receiver gets overloaded again. With very large windows (16K+), the system stays stuck in slow mode with lots of lost packets.The best performance happens with smaller windows (1K-4K) that don't overload the receiver. This shows we need to find the right balance - not too small to be slow, but not too big to cause problems.

2.II - identifying the TCP options used for TCP a data segment whose source is node n7.

Total Frame Length = 72 Bytes, TCP Header + IP Header = 52 Bytes

Data = 4 bytes

Ethernet Header is 72 – 52 – 4 =16 bytes.

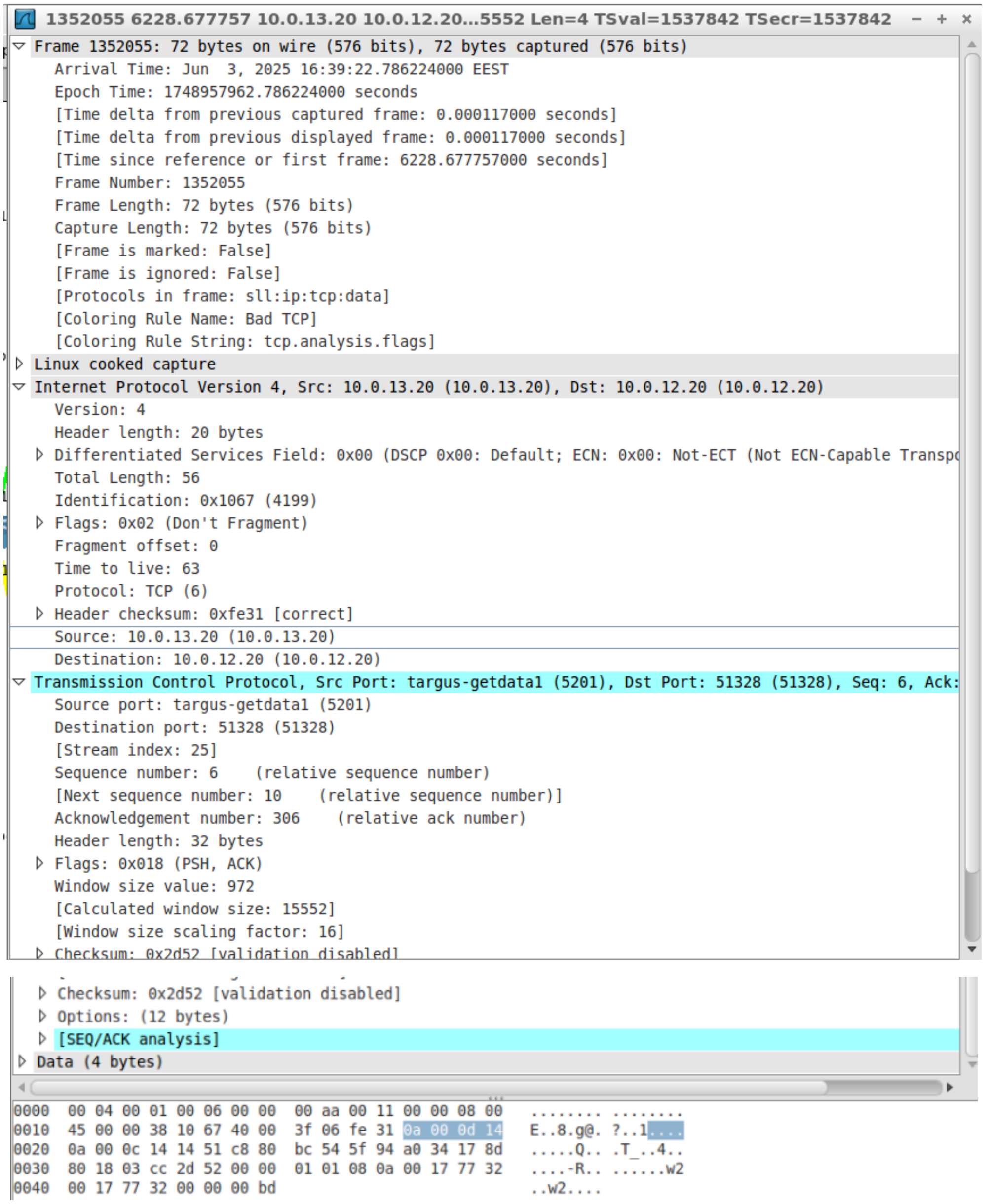
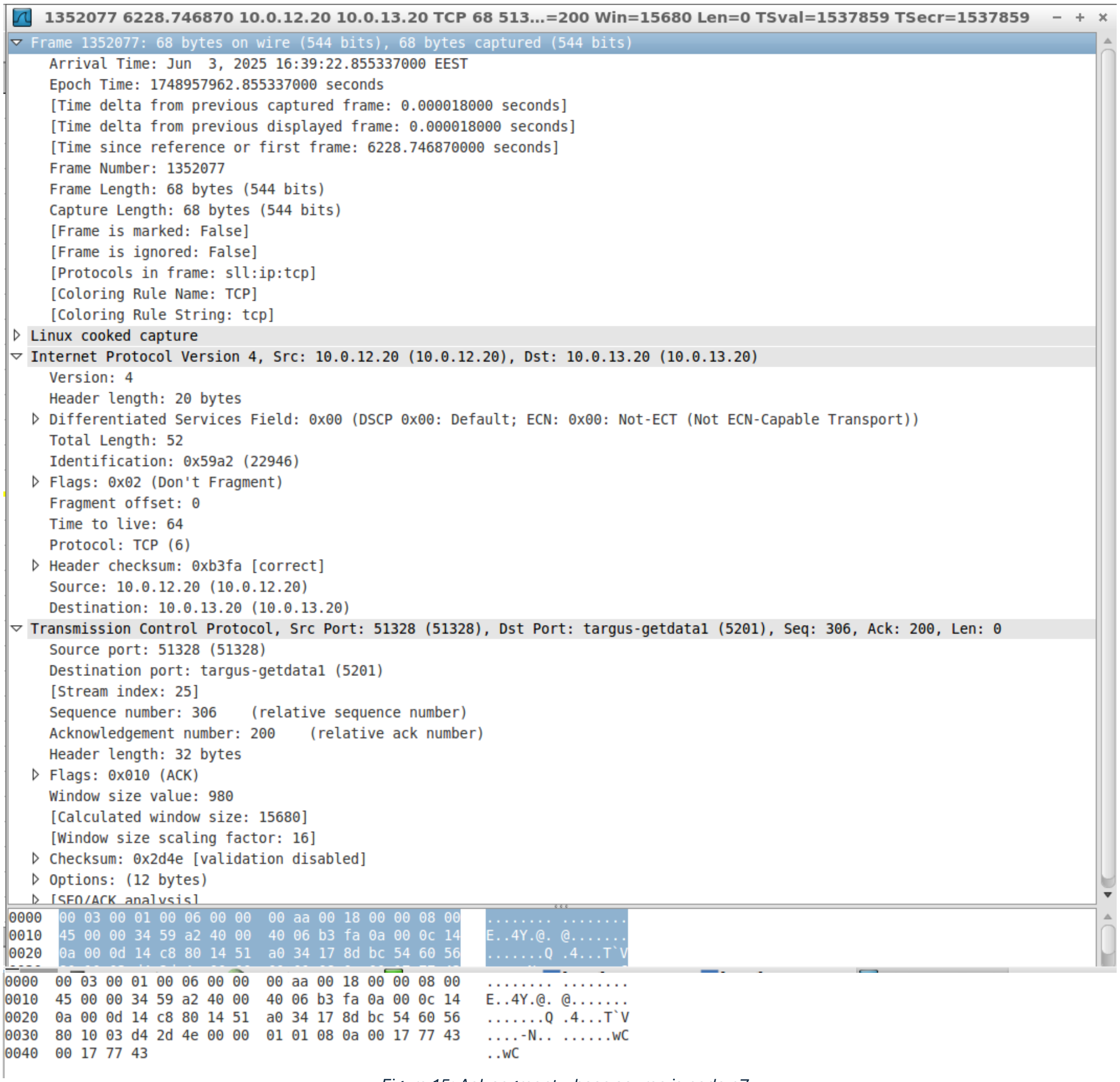


Figure 14: data segment whose source is node n7

2.III - Repeating for an ACK packet sent from node n11.

Total Frame Length = 68 Bytes, TCP Header + IP Header= 52 Bytes

Ethernet Header is 68 - 52 =16 bytes .



```
1352077 6228.746870 10.0.12.20 10.0.13.20 TCP 68 513...=200 Win=15680 Len=0 TSval=1537859 TSecr=1537859
Frame 1352077: 68 bytes on wire (544 bits), 68 bytes captured (544 bits)
    Arrival Time: Jun  3, 2025 16:39:22.855337000 EEST
    Epoch Time: 1748957962.855337000 seconds
    [Time delta from previous captured frame: 0.000018000 seconds]
    [Time delta from previous displayed frame: 0.000018000 seconds]
    [Time since reference or first frame: 6228.746870000 seconds]
    Frame Number: 1352077
    Frame Length: 68 bytes (544 bits)
    Capture Length: 68 bytes (544 bits)
    [Frame is marked: False]
    [Frame is ignored: False]
    [Protocols in frame: sll:ip:tcp]
    [Coloring Rule Name: TCP]
    [Coloring Rule String: tcp]
Linux cooked capture
Internet Protocol Version 4, Src: 10.0.12.20 (10.0.12.20), Dst: 10.0.13.20 (10.0.13.20)
    Version: 4
    Header length: 20 bytes
    Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00: Not-ECT (Not ECN-Capable Transport))
    Total Length: 52
    Identification: 0x59a2 (22946)
    Flags: 0x02 (Don't Fragment)
    Fragment offset: 0
    Time to live: 64
    Protocol: TCP (6)
    Header checksum: 0xb3fa [correct]
    Source: 10.0.12.20 (10.0.12.20)
    Destination: 10.0.13.20 (10.0.13.20)
Transmission Control Protocol, Src Port: 51328 (51328), Dst Port: targus-getdata1 (5201), Seq: 306, Ack: 200, Len: 0
    Source port: 51328 (51328)
    Destination port: targus-getdata1 (5201)
    [Stream index: 25]
    Sequence number: 306    (relative sequence number)
    Acknowledgement number: 200    (relative ack number)
    Header length: 32 bytes
    Flags: 0x010 (ACK)
    Window size value: 980
    [Calculated window size: 15680]
    [Window size scaling factor: 16]
    Checksum: 0x2d4e [validation disabled]
    Options: (12 bytes)
    [SEQ/ACK analysis]
0000  00 03 00 01 00 06 00 00  00 aa 00 18 00 00 08 00   ........ ........
0010  45 00 00 34 59 a2 40 00  40 06 b3 fa 0a 00 0c 14   E..4Y.@. @.......
0020  0a 00 0d 14 c8 80 14 51  a0 34 17 8d bc 54 60 56   .......Q .4...T`V
0000  00 03 00 01 00 06 00 00  00 aa 00 18 00 00 08 00   ........ ........
0010  45 00 00 34 59 a2 40 00  40 06 b3 fa 0a 00 0c 14   E..4Y.@. @.......
0020  0a 00 0d 14 c8 80 14 51  a0 34 17 8d bc 54 60 56   .......Q .4...T`V
0030  80 10 03 d4 2d 4e 00 00  01 01 08 0a 00 17 77 43   ....-N.. ......wC
0040  00 17 77 43                                        ..wC
```

Figure 15: Ack segment whose source is node n7

3.TCP short versus long paths

```
root@n7:/tmp/pycore.59150/n7.conf# iperf3 -c 10.0.13.20 -t 40 -i 10 -w 4k
Connecting to host 10.0.13.20, port 5201
[  4] local 10.0.12.20 port 51336 connected to 10.0.13.20 port 5201
[ ID] Interval           Transfer     Bandwidth       Retr  Cwnd
[  4]   0.00-10.00  sec  4.38 MBytes  3.67 Mbits/sec    0   14.1 KBytes
[  4]  10.00-20.00  sec  4.12 MBytes  3.45 Mbits/sec    1   8.48 KBytes
[  4]  20.00-30.02  sec  5.04 MBytes  4.22 Mbits/sec    0   8.48 KBytes
[  4]  30.02-40.00  sec  3.47 MBytes  2.92 Mbits/sec    0   8.48 KBytes
- - - - - - - - - - - - - - - - - - - - - - - - -
[ ID] Interval           Transfer     Bandwidth       Retr
[  4]   0.00-40.00  sec  17.0 MBytes  3.57 Mbits/sec    1             sender
[  4]   0.00-40.00  sec  17.0 MBytes  3.57 Mbits/sec                  receiver

iperf Done.
```

Figure 16: TCP Short path as n11 as a server and n7 as a client

```
root@n7:/tmp/pycore.59119/n7.conf# iperf3 -c 10.0.11.20 -t 40 -i 10 -w 4k
Connecting to host 10.0.11.20, port 5201
[  4] local 10.0.12.20 port 46407 connected to 10.0.11.20 port 5201
[ ID] Interval           Transfer     Bandwidth       Retr  Cwnd
[  4]   0.00-10.00  sec  3.29 MBytes  2.76 Mbits/sec    0   14.1 KBytes
[  4]  10.00-20.00  sec  3.69 MBytes  3.10 Mbits/sec    0   14.1 KBytes
[  4]  20.00-30.00  sec  3.68 MBytes  3.09 Mbits/sec    1   8.48 KBytes
[  4]  30.00-40.00  sec  2.77 MBytes  2.32 Mbits/sec    0   8.48 KBytes
- - - - - - - - - - - - - - - - - - - - - - - - -
[ ID] Interval           Transfer     Bandwidth       Retr
[  4]   0.00-40.00  sec  13.4 MBytes  2.82 Mbits/sec    1             sender
[  4]   0.00-40.00  sec  13.4 MBytes  2.82 Mbits/sec                  receiver

iperf Done.
```

Figure 17: TCP Long path as n8 as a server and n7 as a client

$$\text{Throughput for node n8 as a server and n7 as client } = \text{Bandwidth} = \frac{2^{20} \times 13.4 \times 8}{10^6 \times 40} = \ 2.82 \text{ Mbits/s}$$

$$\text{Throughput for node n11 as a server and n7 as client } = \text{Bandwidth} = \frac{2^{20} \times 17 \times 8}{10^6 \times 40} = \ 3.57 \text{ Mbits/s}$$

Comment:

The test shows slower network speeds when transferring data between node n8 (server) and node n7 (client) compared to transfers between n11 (server) and n7 (client). This happens because the network path between n8 and n7 is longer than between n11 and n7. The longer distance means packets take more time to travel, causing delays that reduce the overall throughput. Essentially, the farther apart two nodes are in the network, the slower their connection tends to be.

4.Higher Link Capacity with Drops versus Reliable Lower Capacity

Select the link between n4 and n5, configure it to have the following:

A. capacity of 10 Mbps with zero loss in both directions.

```
root@n7:/tmp/pycore.59153/n7.conf# iperf3 -c 10.0.13.20 -t 40 -i 10 -w 4k
Connecting to host 10.0.13.20, port 5201
[  4] local 10.0.12.20 port 51339 connected to 10.0.13.20 port 5201
[ ID] Interval           Transfer     Bandwidth       Retr  Cwnd
[  4]   0.00-10.02  sec  4.70 MBytes  3.93 Mbits/sec    1   7.07 KBytes
[  4]  10.02-20.02  sec  3.23 MBytes  2.71 Mbits/sec    1   8.48 KBytes
[  4]  20.02-30.00  sec  4.82 MBytes  4.05 Mbits/sec    0   8.48 KBytes
[  4]  30.00-40.06  sec  5.25 MBytes  4.38 Mbits/sec    1   8.48 KBytes
- - - - - - - - - - - - - - - - - - - - - - - - -
[ ID] Interval           Transfer     Bandwidth       Retr
[  4]   0.00-40.06  sec  18.0 MBytes  3.77 Mbits/sec    3             sender
[  4]   0.00-40.06  sec  18.0 MBytes  3.77 Mbits/sec                  receiver

iperf Done.
```

Figure 18: Testing Capacity 10Mbps with zero loss

$$\text{Throughput} = Bandwidth = \frac{2^{20} \times 18 \times 8}{10^6 \times 40} = 3.77 \text{ Mbits/s}$$

B. capacity of 3 Mbps with zero loss in both directions.

```
root@n7:/tmp/pycore.59153/n7.conf# iperf3 -c 10.0.13.20 -t 40 -i 10 -w 4k
Connecting to host 10.0.13.20, port 5201
[  4] local 10.0.12.20 port 51343 connected to 10.0.13.20 port 5201
[ ID] Interval           Transfer     Bandwidth       Retr  Cwnd
[  4]   0.00-10.00  sec  1.83 MBytes  1.53 Mbits/sec    1   8.48 KBytes
[  4]  10.00-20.00  sec  2.51 MBytes  2.11 Mbits/sec    0   8.48 KBytes
[  4]  20.00-30.00  sec  2.38 MBytes  2.00 Mbits/sec    0   8.48 KBytes
[  4]  30.00-40.00  sec  1.76 MBytes  1.47 Mbits/sec    1   8.48 KBytes
- - - - - - - - - - - - - - - - - - - - - - - - -
[ ID] Interval           Transfer     Bandwidth       Retr
[  4]   0.00-40.00  sec  8.48 MBytes  1.78 Mbits/sec    2             sender
[  4]   0.00-40.00  sec  8.48 MBytes  1.78 Mbits/sec                  receiver

iperf Done.
```

Figure 19: Testing Capacity 3 Mbps with zero loss

$$\text{Throughput} = Bandwidth = \frac{2^{20} \times 8.48 \times 8}{10^6 \times 40} = 1.78 \text{ Mbits/s}$$

C. capacity of 10 Mbps with 5% loss in both directions.

```
root@n7:/tmp/pycore.59153/n7.conf# iperf3 -c 10.0.13.20 -t 40 -i 10 -w 4k
Connecting to host 10.0.13.20, port 5201
[  4] local 10.0.12.20 port 51347 connected to 10.0.13.20 port 5201
[ ID] Interval           Transfer     Bandwidth       Retr  Cwnd
[  4]   0.00-10.00  sec   300 KBytes   246 Kbits/sec   15   4.24 KBytes
[  4]  10.00-20.02  sec   269 KBytes   220 Kbits/sec   20   8.48 KBytes
[  4]  20.02-30.00  sec   484 KBytes   397 Kbits/sec   24   5.66 KBytes
[  4]  30.00-40.00  sec   625 KBytes   512 Kbits/sec   29   4.24 KBytes
- - - - - - - - - - - - - - - - - - - - - - - - -
[ ID] Interval           Transfer     Bandwidth       Retr
[  4]   0.00-40.00  sec  1.64 MBytes   343 Kbits/sec   88             sender
[  4]   0.00-40.00  sec  1.63 MBytes   342 Kbits/sec                  receiver

iperf Done.
```

Figure 20: Testing Capacity 10Mbps with 5% loss

$$\text{Throughput} = Bandwidth = \frac{2^{20} \times 1.64 \times 8}{10^6 \times 40} = 0.343 \text{ Mbits/s}$$

D. capacity of 100 Mbps with 10% loss in both directions.

```
root@n7:/tmp/pycore.59153/n7.conf# iperf3 -c 10.0.13.20 -t 40 -i 10 -w 4k
Connecting to host 10.0.13.20, port 5201
[  4] local 10.0.12.20 port 51349 connected to 10.0.13.20 port 5201
[ ID] Interval           Transfer     Bandwidth       Retr  Cwnd
[  4]   0.00-10.00  sec   310 KBytes   254 Kbits/sec   26   4.24 KBytes
[  4]  10.00-20.00  sec   168 KBytes   138 Kbits/sec   16   4.24 KBytes
[  4]  20.00-30.00  sec   281 KBytes   231 Kbits/sec   24   4.24 KBytes
[  4]  30.00-40.00  sec   556 KBytes   455 Kbits/sec   40   5.66 KBytes
- - - - - - - - - - - - - - - - - - - - - - - - -
[ ID] Interval           Transfer     Bandwidth       Retr
[  4]   0.00-40.00  sec  1.28 MBytes   269 Kbits/sec  106             sender
[  4]   0.00-40.00  sec  1.28 MBytes   268 Kbits/sec                  receiver

iperf Done.
```

Figure 21: Testing Capacity 100 Mbps with 10 % loss

$$\text{Throughput} = Bandwidth = \frac{2^{20} \times 1.28 \times 8}{10^6 \times 40} = 0.269 \text{ Mbits/s}$$

E. Select the link between n4 and n5, configure it to have capacity of 10 Mbps with 1% loss in direction from n4 to n5 and 0% loss in the other direction.

```
root@n7:/tmp/pycore.59153/n7.conf# iperf3 -c 10.0.13.20 -t 40 -i 10 -w 4k
Connecting to host 10.0.13.20, port 5201
[  4] local 10.0.12.20 port 51351 connected to 10.0.13.20 port 5201
[ ID] Interval           Transfer     Bandwidth       Retr  Cwnd
[  4]   0.00-10.00  sec  2.90 MBytes  2.43 Mbits/sec   11   8.48 KBytes
[  4]  10.00-20.03  sec  1.93 MBytes  1.62 Mbits/sec    4   8.48 KBytes
[  4]  20.03-30.01  sec  4.55 MBytes  3.83 Mbits/sec    8   8.48 KBytes
[  4]  30.01-40.00  sec  3.09 MBytes  2.60 Mbits/sec    4   8.48 KBytes
- - - - - - - - - - - - - - - - - - - - - - - - -
[ ID] Interval           Transfer     Bandwidth       Retr
[  4]   0.00-40.00  sec  12.5 MBytes  2.62 Mbits/sec   27             sender
[  4]   0.00-40.00  sec  12.5 MBytes  2.62 Mbits/sec                  receiver

iperf Done.
```

Figure 22: Testing Capacity 10Mbps with 1% loss in direction of n4 to n5

$$\text{Throughput} = Bandwidth = \frac{2^{20} \times 12.5 \times 8}{10^6 \times 40} = 2.62 \text{ Mbits/s}$$

F. Select link between n4 and n5, configure it to have capacity of 10 Mbps with 0% loss in direction from n4 to n5 and 1% loss in the other direction.

```
root@n7:/tmp/pycore.59153/n7.conf# iperf3 -c 10.0.13.20 -t 40 -i 10 -w 4k
Connecting to host 10.0.13.20, port 5201
[  4] local 10.0.12.20 port 51353 connected to 10.0.13.20 port 5201
[ ID] Interval           Transfer     Bandwidth       Retr  Cwnd
[  4]   0.00-10.00  sec  2.21 MBytes  1.86 Mbits/sec   21   5.66 KBytes
[  4]  10.00-20.00  sec  1.81 MBytes  1.52 Mbits/sec   11   5.66 KBytes
[  4]  20.00-30.00  sec  2.67 MBytes  2.24 Mbits/sec   15   5.66 KBytes
[  4]  30.00-40.00  sec  3.05 MBytes  2.56 Mbits/sec   25   4.24 KBytes
- - - - - - - - - - - - - - - - - - - - - - - - -
[ ID] Interval           Transfer     Bandwidth       Retr
[  4]   0.00-40.00  sec  9.74 MBytes  2.04 Mbits/sec   72             sender
[  4]   0.00-40.00  sec  9.74 MBytes  2.04 Mbits/sec                  receiver

iperf Done.
```

Figure 23: Testing Capacity 10Mbps with 1% loss in direction of n5 to n4

$$\text{Throughput} = Bandwidth = \frac{2^{20} \times 9.74 \times 8}{10^6 \times 40} = 2.04 \text{ Mbits/s}$$

- **Comment on throughputs in cases a, b, c. Why b is better than c?**

We can see that the throughput of A (3.77 Mbps) is higher than Throughput in B (1.78Mbps) and Throughput in B is higher than Throughput in C (0.343 Mbps) Although the capacity of the link in C is higher than B but the losses in C are higher than B and that will make packets need to be retransmitted and that will decrease the throughput and that's why B is better than C and A is better than both because it has higher link capacity than B and have no losses like C.

- **Comment on throughputs in cases b, c and d. Which is better? Why?**

We can see that the throughput of B (1.78 Mbps) is higher than Throughput in C (0.343 Mbps) and Throughput in C is higher than Throughput in D (0.269 Mbps) so B is the best and that's because B has no losses Although the capacity of the link in C and D is higher than B but the losses in C and D is higher than B and that will make packets need to be retransmitted and that will decrease the throughput for the two cases.

- **Comment on throughputs in cases e and f. Which is better? Why?**

We can see that the throughput of E (2.62 Mbps) is higher than Throughput in F (2.04 Mbps) and that's because packets in uplink is most likely bigger than packets in downlink and that's because these massages are most likely acks so losses in uplink will affect more than in downlink.

OSPF_Part

5 - OSPF Link Cost Changes

5-I What happens to the path between n7 and n11 (as seen after steps 3 and 6).

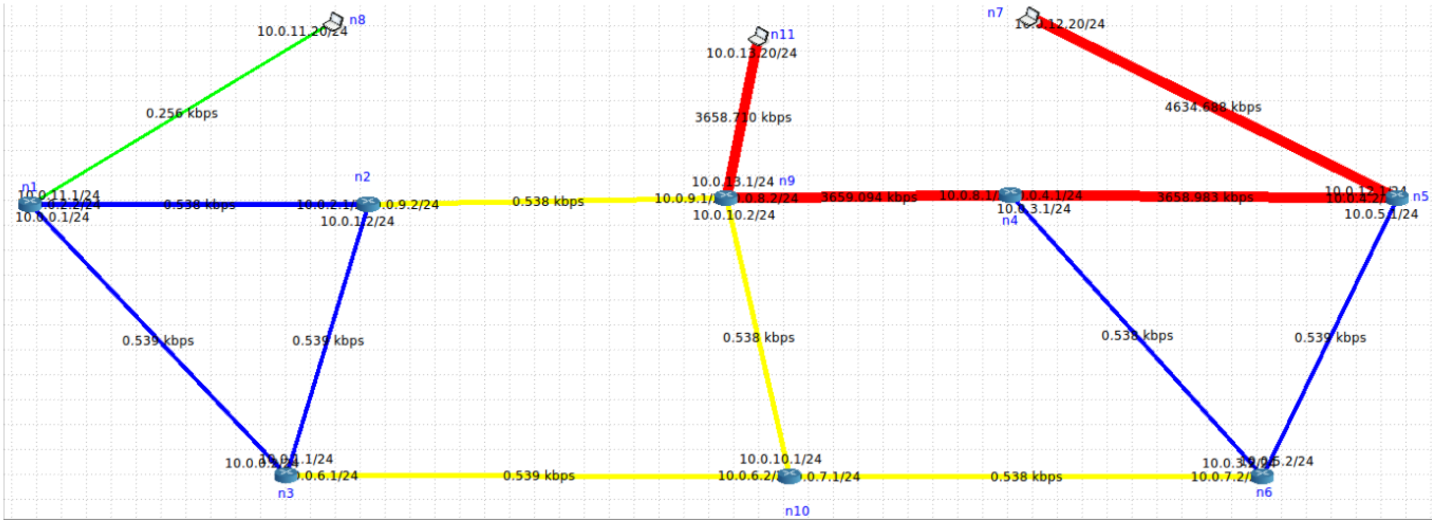


Figure 24: Path from n7 to n11 with links have the same cost

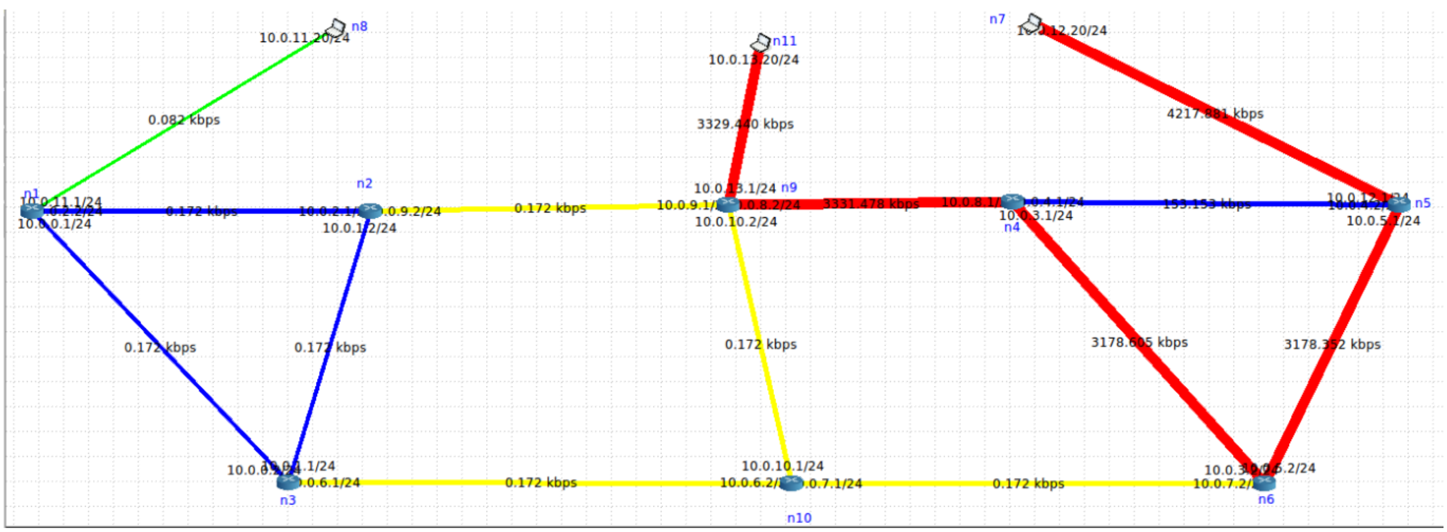


Figure 25: Path from n7 to n11 when n5 eth1 has cost=40

- **Comment on the path from n7 to n11**

As shown in Figure 24, each node selects the path with the lowest cost. In this case, since all links have the same cost, the nodes choose the shortest path.

As shown in Figure 25, after increasing the cost of the link (n5 → n4) from 10 to 40, node 5 changes its path to be (n5 → n6 → n4) which has a lower total cost of 20.

5- II What happens in the paths of the two connections (n7 → n11) and (n11 → n7) before and after increasing link (n4 → n5) cost from 10 to 40.

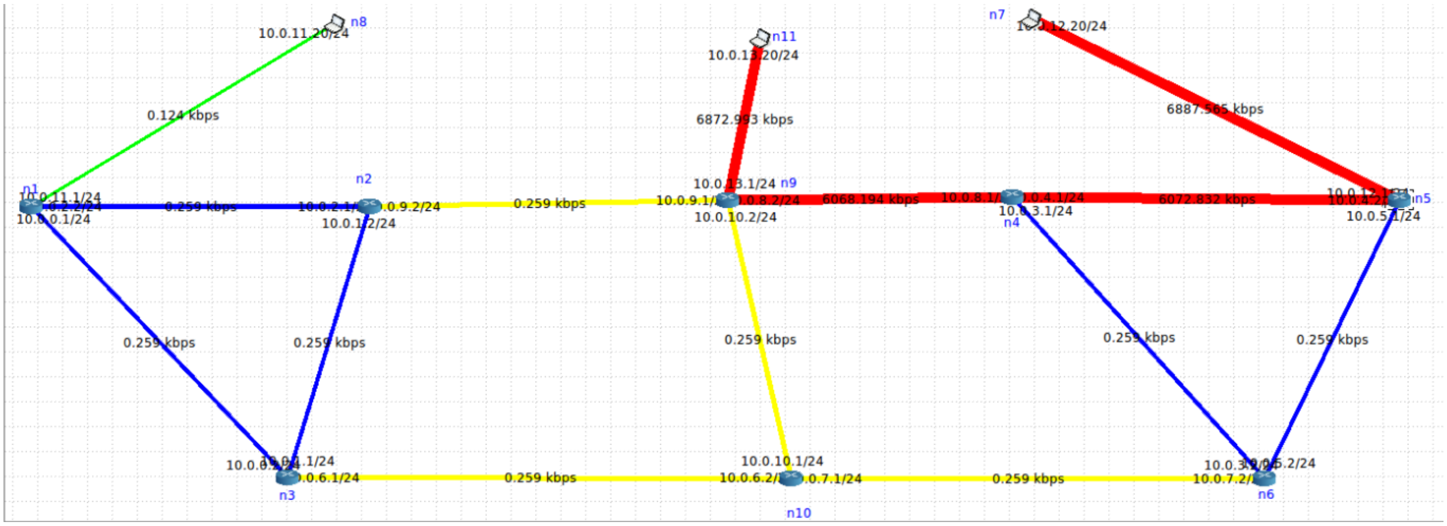


Figure 26: Paths from n7 to n11 and from n11 to n7

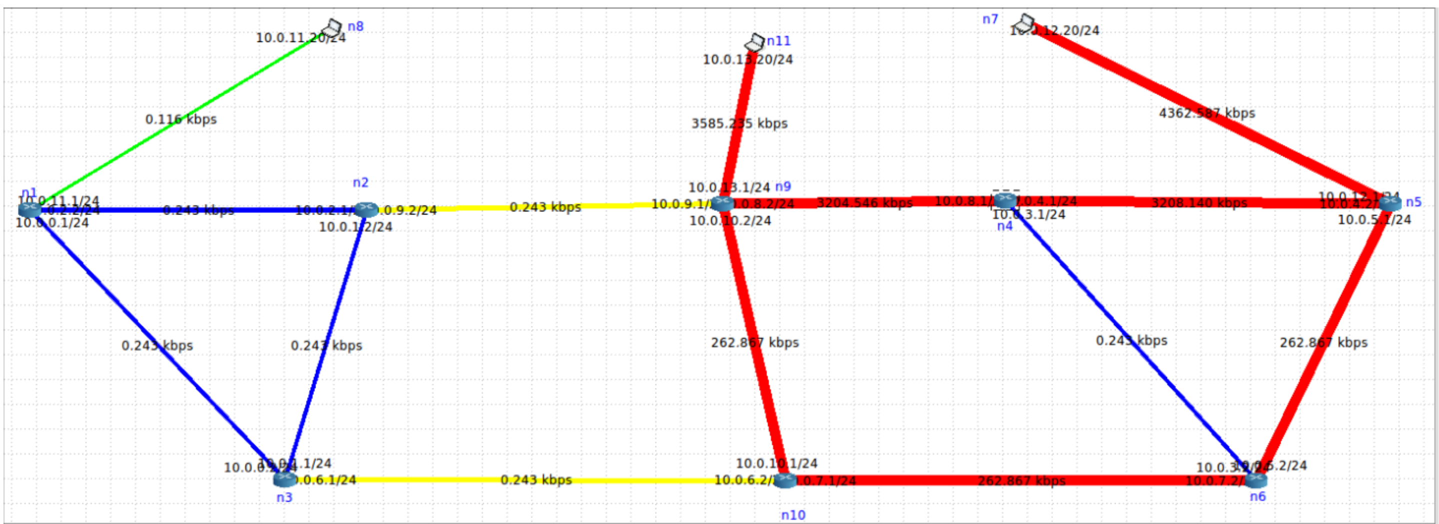


Figure 27: Paths from n7 to n11 and from n11 to n7 when link (n4 to n5) has cost=40

- **Comment on the paths from n7 to n11 and from n11 to n7**

When all links have the same cost, both connections follow the same path, as shown in Figure 26.

However, when the cost of the link from n4 to n5 is increased from 10 to 40, the path from n7 to n11 remains unchanged, but the path from n11 to n7 is changed to take a lower-cost path.

As shown in Figure 27, the path from n11 to n7 changed from (n9 → n4 → n5) with total cost = 50 to (n9 → n10 → n6 → n5) with total cost=30.

- **Conclusion**

The routing path can change based on link costs, similar to how traffic adjusts to road conditions. As a result, network routing can be asymmetric, with different paths used in each direction

6 - OSPF Database Updates

6-I Database and routing tables for router n2.

```
n2# show ip ospf database

       OSPF Router with ID (10.0.1.2)

                Router Link States (Area 0.0.0.0)

Link ID         ADV Router      Age  Seq#       CkSum  Link count
10.0.0.1        10.0.0.1         521 0x8000000e 0xeac1 3
10.0.0.2        10.0.0.2         471 0x8000000d 0x940b 3
10.0.1.2        10.0.1.2         460 0x8000000e 0x850e 3
10.0.3.1        10.0.3.1         672 0x80000010 0x3b30 3
10.0.3.2        10.0.3.2         523 0x8000000e 0xa5e2 3
10.0.5.1        10.0.5.1        1553 0x8000000c 0x8c09 3
10.0.6.2        10.0.6.2         502 0x8000000e 0xc2b0 3
10.0.8.2        10.0.8.2         581 0x8000000f 0x1c18 4

                Net Link States (Area 0.0.0.0)

Link ID         ADV Router      Age  Seq#       CkSum
10.0.0.2        10.0.0.2        1501 0x80000004 0x59ce
10.0.1.2        10.0.1.2         480 0x80000005 0x5cc6
10.0.2.1        10.0.1.2        1490 0x80000004 0x4bd9
10.0.3.2        10.0.3.2        1473 0x80000004 0x65b6
10.0.4.2        10.0.5.1        1313 0x80000005 0x5cbb

...skipping one line
10.0.6.2        10.0.6.2         412 0x80000005 0x43d0
10.0.7.1        10.0.6.2        1352 0x80000005 0x51bf
10.0.8.2        10.0.8.2        1371 0x80000005 0x4ac1
10.0.9.1        10.0.8.2         501 0x80000005 0x25e7
10.0.10.2       10.0.8.2         211 0x80000005 0x59ac
```

Figure 28: Database for Router n2

```
n2# show ip ospf route
============ OSPF network routing table ============
N    10.0.0.0/24           [20] area: 0.0.0.0
                           via 10.0.1.1, eth0
                           via 10.0.2.2, eth1
N    10.0.1.0/24           [10] area: 0.0.0.0
                           directly attached to eth0
N    10.0.2.0/24           [10] area: 0.0.0.0
                           directly attached to eth1
N    10.0.3.0/24           [30] area: 0.0.0.0
                           via 10.0.9.1, eth2
N    10.0.4.0/24           [50] area: 0.0.0.0
                           via 10.0.1.1, eth0
                           via 10.0.9.1, eth2
N    10.0.5.0/24           [40] area: 0.0.0.0
                           via 10.0.1.1, eth0
                           via 10.0.9.1, eth2
N    10.0.6.0/24           [20] area: 0.0.0.0
                           via 10.0.1.1, eth0
N    10.0.7.0/24           [30] area: 0.0.0.0
                           via 10.0.1.1, eth0
                           via 10.0.9.1, eth2
N    10.0.8.0/24           [20] area: 0.0.0.0
                           via 10.0.9.1, eth2
--More--
```

Figure 29: Routing table for Router n2

- **Comment on the database and the routing table for Router n2**

As shown in Figure 28, the data includes information for each node, such as its age and the number of links connected to it.

In Figure 29, some destinations have more than one route because all links have the same cost, except for the link from n4 to n5, which has a higher cost of 40.

6- II How long does the network take to update its routing tables after changing the cost of n2's eth1 interface to 20?

335	20.794647	fe80::200:ff:feaa:a	ff02::5	OSPF	96 Hello Packet
336	20.794696	fe80::200:ff:feaa:1b	ff02::5	OSPF	92 Hello Packet
337	21.158010	10.0.3.1	224.0.0.5	OSPF	124 LS Update
338	21.158023	10.0.3.1	224.0.0.5	OSPF	124 LS Update
339	21.158042	10.0.4.1	224.0.0.5	OSPF	124 LS Update
340	21.158045	10.0.4.1	224.0.0.5	OSPF	124 LS Update
341	21.158053	10.0.8.1	224.0.0.5	OSPF	124 LS Update
342	21.158058	10.0.8.1	224.0.0.5	OSPF	124 LS Update
343	21.158266	10.0.8.1	224.0.0.5	OSPF	124 LS Update
344	21.158270	10.0.3.1	224.0.0.5	OSPF	124 LS Update

Figure 30: Wireshark response after changing the network cost

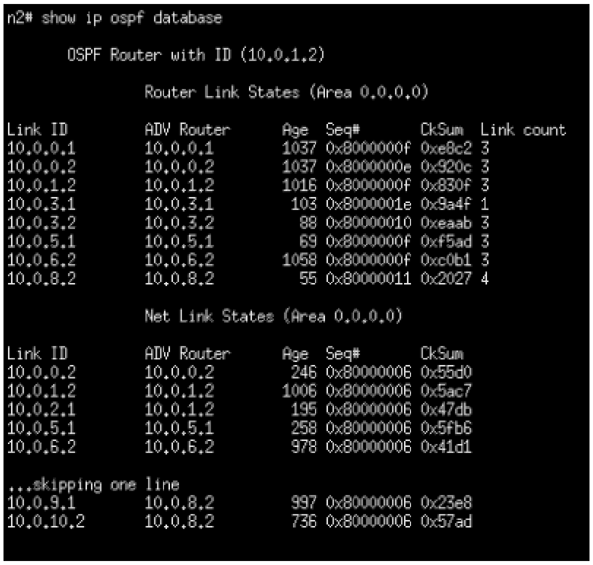
411	22.111092	10.0.7.2	224.0.0.5	OSPF	80 LS Acknowledge
412	22.112125	10.0.7.2	224.0.0.5	OSPF	80 LS Acknowledge
413	22.112129	10.0.3.2	224.0.0.5	OSPF	80 LS Acknowledge
414	22.112131	10.0.5.2	224.0.0.5	OSPF	80 LS Acknowledge
417	24.752968	10.0.0.1	224.0.0.5	OSPF	84 Hello Packet
418	24.752980	10.0.0.1	224.0.0.5	OSPF	84 Hello Packet
419	24.752982	10.0.5.1	224.0.0.5	OSPF	84 Hello Packet

Figure 31: Wireshark response to the ACK following the change

- **Comment on the time taken to update routing tables**

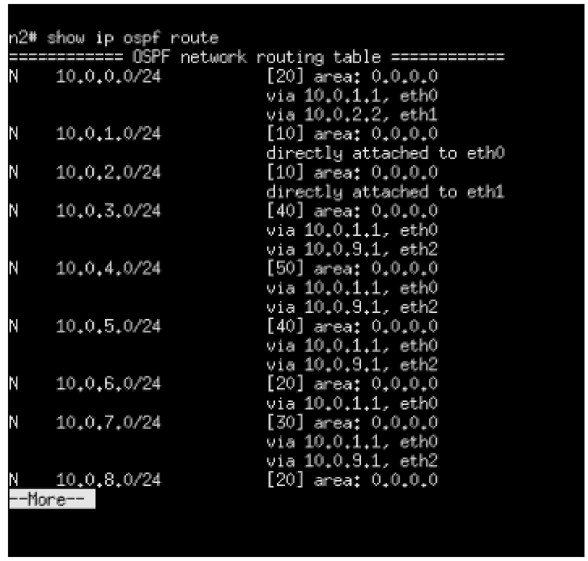
As shown in Figures 30–31, the network takes time to exchange link-state packets and update its routing tables. The total convergence time is 22.112 - 21.158 = 0.954 seconds.

6- III Database and routing tables for router n2 after disconnecting router n4.



```
n2# show ip ospf database

      OSPF Router with ID (10.0.1.2)

              Router Link States (Area 0.0.0.0)

Link ID         ADV Router      Age  Seq#       CkSum  Link count
10.0.0.1        10.0.0.1        1037 0x8000000f 0xe8c2 3
10.0.0.2        10.0.0.2        1037 0x8000000e 0x920c 3
10.0.1.2        10.0.1.2        1016 0x8000000f 0x830f 3
10.0.3.1        10.0.3.1         103 0x8000001e 0x9a4f 1
10.0.3.2        10.0.3.2          88 0x80000010 0xeaab 3
10.0.5.1        10.0.5.1          69 0x8000000f 0xf5ad 3
10.0.6.2        10.0.6.2        1058 0x8000000f 0xc0b1 3
10.0.8.2        10.0.8.2          55 0x80000011 0x2027 4

              Net Link States (Area 0.0.0.0)

Link ID         ADV Router      Age  Seq#       CkSum
10.0.0.2        10.0.0.2         246 0x80000006 0x55d0
10.0.1.2        10.0.1.2        1006 0x80000006 0x5ac7
10.0.2.1        10.0.1.2         195 0x80000006 0x47db
10.0.5.1        10.0.5.1         258 0x80000006 0x5fb6
10.0.6.2        10.0.6.2         978 0x80000006 0x41d1

...skipping one line
10.0.9.1        10.0.8.2         997 0x80000006 0x23e8
10.0.10.2       10.0.8.2         736 0x80000006 0x57ad
```

Figure 32: Database for Router n2 and disconnect n4



```
n2# show ip ospf route
============ OSPF network routing table ============
N    10.0.0.0/24          [20] area: 0.0.0.0
                          via 10.0.1.1, eth0
                          via 10.0.2.2, eth1
N    10.0.1.0/24          [10] area: 0.0.0.0
                          directly attached to eth0
N    10.0.2.0/24          [10] area: 0.0.0.0
                          directly attached to eth1
N    10.0.3.0/24          [40] area: 0.0.0.0
                          via 10.0.1.1, eth0
                          via 10.0.9.1, eth2
N    10.0.4.0/24          [50] area: 0.0.0.0
                          via 10.0.1.1, eth0
                          via 10.0.9.1, eth2
N    10.0.5.0/24          [40] area: 0.0.0.0
                          via 10.0.1.1, eth0
                          via 10.0.9.1, eth2
N    10.0.6.0/24          [20] area: 0.0.0.0
                          via 10.0.1.1, eth0
N    10.0.7.0/24          [30] area: 0.0.0.0
                          via 10.0.1.1, eth0
                          via 10.0.9.1, eth2
N    10.0.8.0/24          [20] area: 0.0.0.0
--More--
```

Figure 33: Routing tables for Router n2 without n2

- **Comment on the database and the routing table for Router n2 after disconnecting Router n4.**

As shown in Figure 32, the number of links for 10.0.3.1 decreased from 3 to 1 due to the disconnection of node n4.
As shown in Figure 33, the cost to reach 10.0.3.0 increased from 30 to 40 as a result of n4 being disconnected.