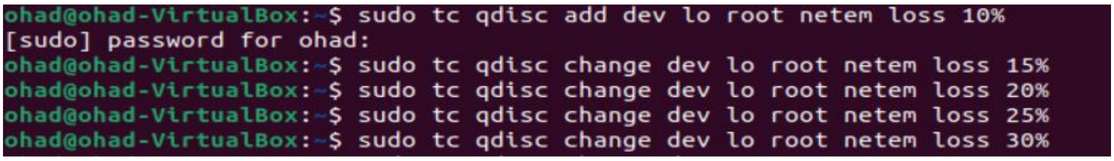
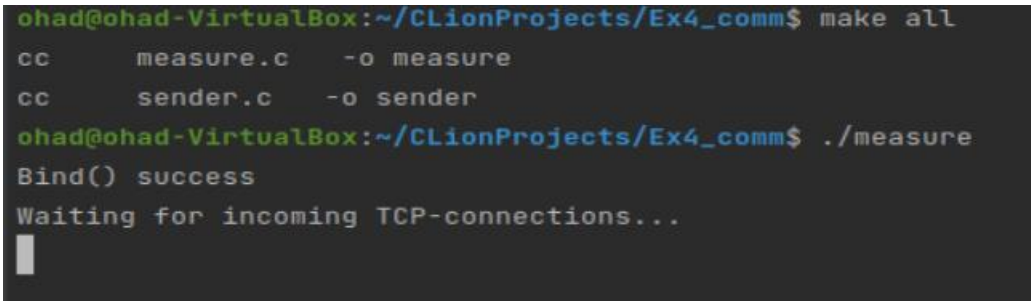
Ex4- Documentation and explanations of Part B: : Ohad Shirazi 318636693

changing of the packet loss by percentage from the terminal:

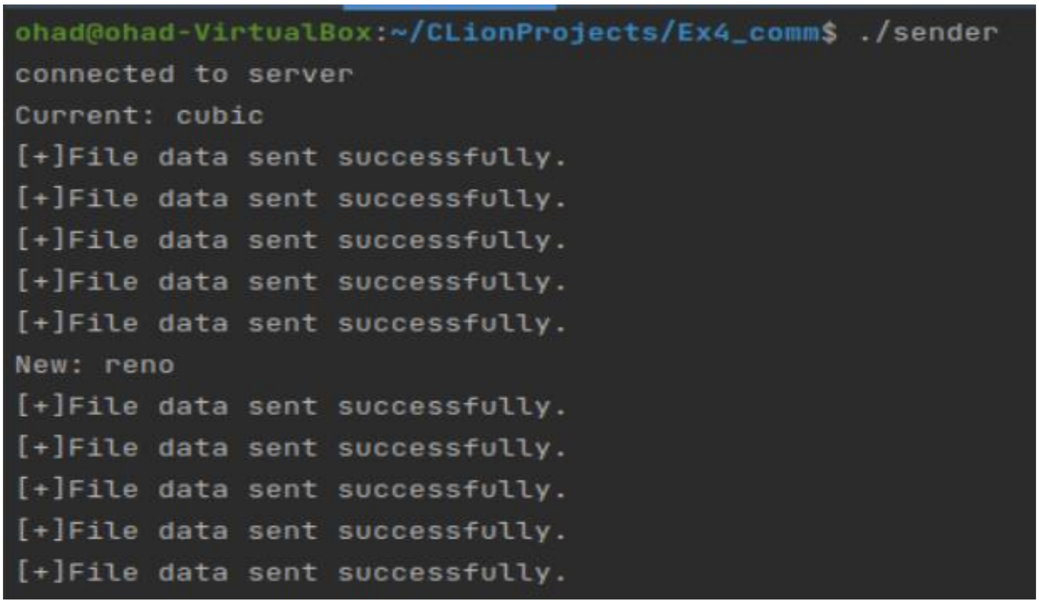


```
ohad@ohad-VirtualBox:~$ sudo tc qdisc add dev lo root netem loss 10%
[sudo] password for ohad:
ohad@ohad-VirtualBox:~$ sudo tc qdisc change dev lo root netem loss 15%
ohad@ohad-VirtualBox:~$ sudo tc qdisc change dev lo root netem loss 20%
ohad@ohad-VirtualBox:~$ sudo tc qdisc change dev lo root netem loss 25%
ohad@ohad-VirtualBox:~$ sudo tc qdisc change dev lo root netem loss 30%
```

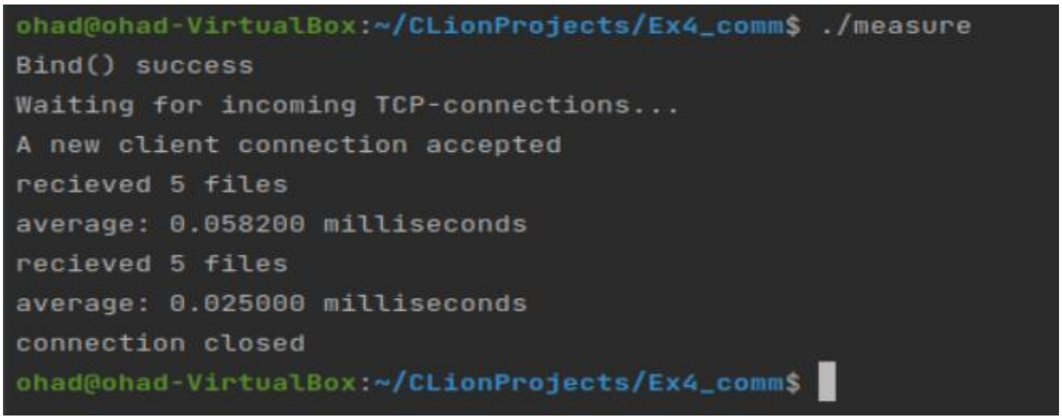
Without any function of packets loss



```
ohad@ohad-VirtualBox:~/CLionProjects/Ex4_comm$ make all
cc     measure.c   -o measure
cc     sender.c   -o sender
ohad@ohad-VirtualBox:~/CLionProjects/Ex4_comm$ ./measure
Bind() success
Waiting for incoming TCP-connections...
```

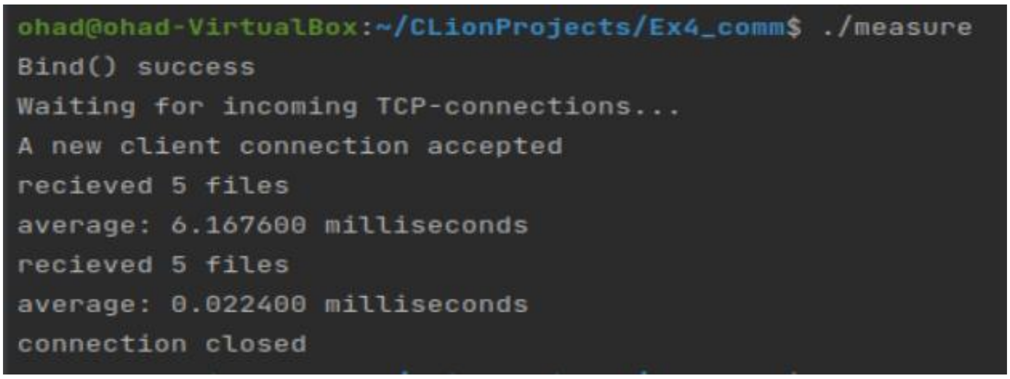


```
ohad@ohad-VirtualBox:~/CLionProjects/Ex4_comm$ ./sender
connected to server
Current: cubic
[+]File data sent successfully.
[+]File data sent successfully.
[+]File data sent successfully.
[+]File data sent successfully.
[+]File data sent successfully.
New: reno
[+]File data sent successfully.
[+]File data sent successfully.
[+]File data sent successfully.
[+]File data sent successfully.
[+]File data sent successfully.
```



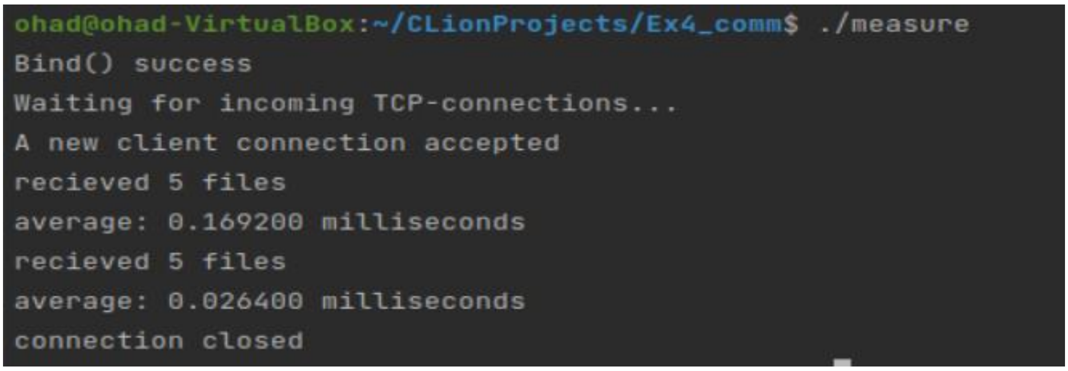
```
ohad@ohad-VirtualBox:~/CLionProjects/Ex4_comm$ ./measure
Bind() success
Waiting for incoming TCP-connections...
A new client connection accepted
recieved 5 files
average: 0.058200 milliseconds
recieved 5 files
average: 0.025000 milliseconds
connection closed
ohad@ohad-VirtualBox:~/CLionProjects/Ex4_comm$
```

Packets loss by 10% percentage



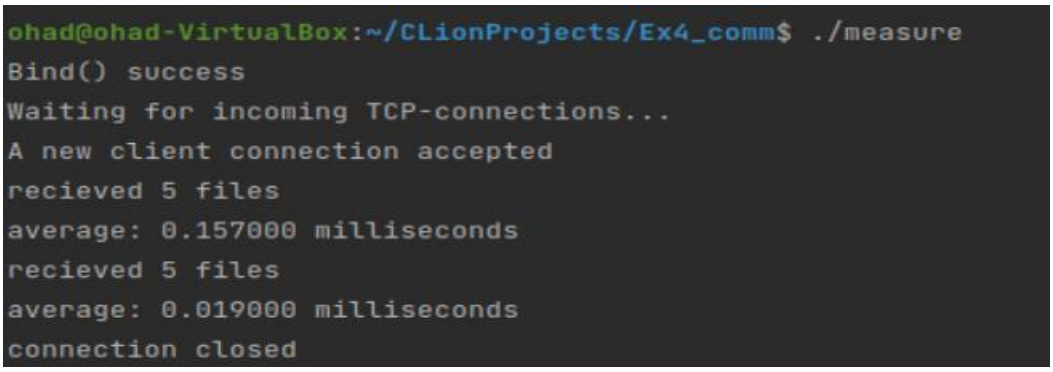
```
ohad@ohad-VirtualBox:~/CLionProjects/Ex4_comm$ ./measure
Bind() success
Waiting for incoming TCP-connections...
A new client connection accepted
recieved 5 files
average: 6.167600 milliseconds
recieved 5 files
average: 0.022400 milliseconds
connection closed
```

Packets loss by 15% percentage



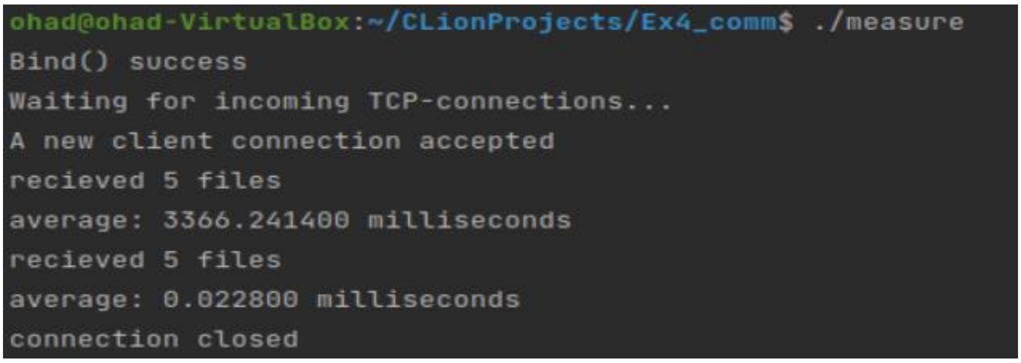
```
ohad@ohad-VirtualBox:~/CLionProjects/Ex4_comm$ ./measure
Bind() success
Waiting for incoming TCP-connections...
A new client connection accepted
recieved 5 files
average: 0.169200 milliseconds
recieved 5 files
average: 0.026400 milliseconds
connection closed
```

Packets loss by 20% percentage



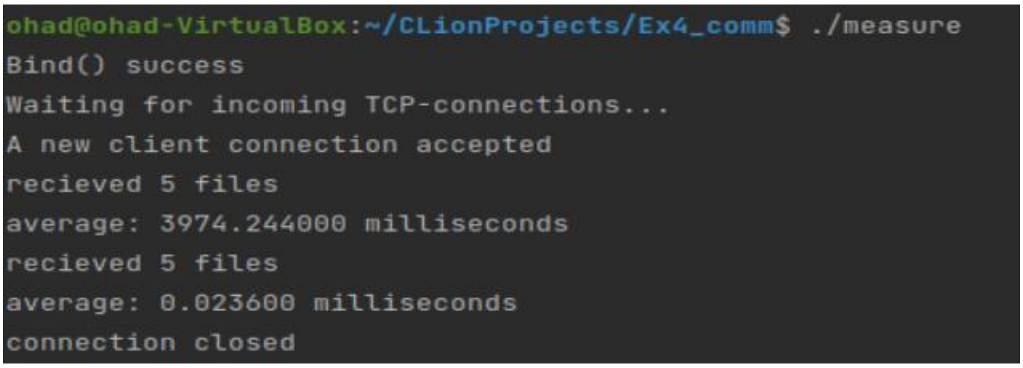
```
ohad@ohad-VirtualBox:~/CLionProjects/Ex4_comm$ ./measure
Bind() success
Waiting for incoming TCP-connections...
A new client connection accepted
recieved 5 files
average: 0.157000 milliseconds
recieved 5 files
average: 0.019000 milliseconds
connection closed
```

Packets loss by 25% percentage



```
ohad@ohad-VirtualBox:~/CLionProjects/Ex4_comm$ ./measure
Bind() success
Waiting for incoming TCP-connections...
A new client connection accepted
recieved 5 files
average: 3366.241400 milliseconds
recieved 5 files
average: 0.022800 milliseconds
connection closed
```

Packets loss by 30% percentage



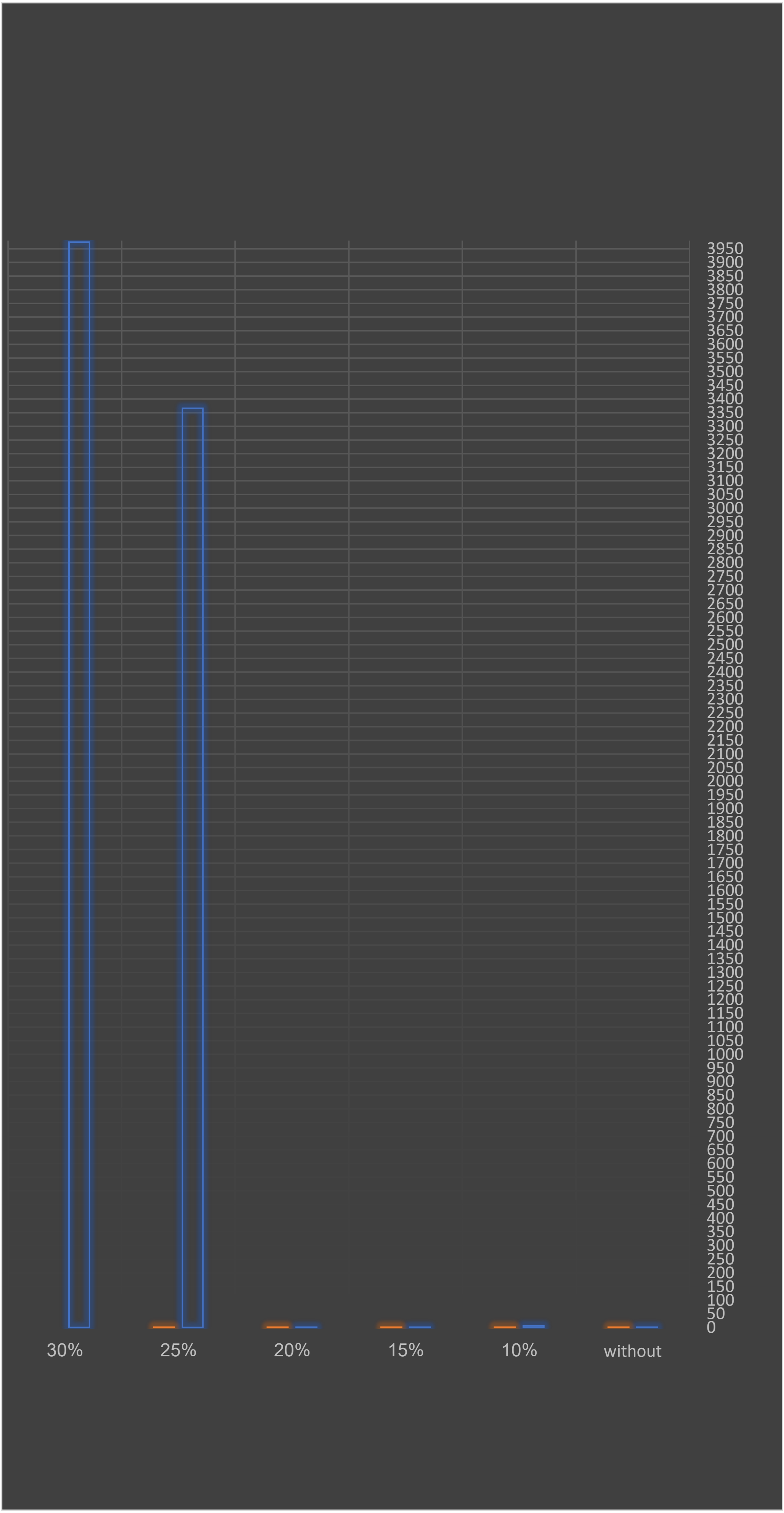
```
ohad@ohad-VirtualBox:~/CLionProjects/Ex4_comm$ ./measure
Bind() success
Waiting for incoming TCP-connections...
A new client connection accepted
recieved 5 files
average: 3974.244000 milliseconds
recieved 5 files
average: 0.023600 milliseconds
connection closed
```

Table that contains all the data above – shows the difference between reno and cubic:

Packets loss by percentage	cubic	reno
without	0.0582 m.s.	0.025 m.s.
10%	6.1676 m.s.	0.0224 m.s.
15%	0.1692 m.s.	0.0264 m.s.
20%	0.157 m.s.	0.019 m.s.
25%	3366.2414 m.s.	0.0228 m.s.
30%	3974.244 m.s.	0.0236 m.s.

This table shows properly the differences between cubic and reno.
As described at the table, we can see on the next page the graph that describes the table, cubic's data is blue, reno's data is orange.
The analysis is an average time per milliseconds of packet loss by percentage on each round.

From the table and also from the graph we can see that in the cases of low packet loss percentage there was a little bit of equality between the algorithms, both are cant be seen, meaning they were very fast relatively. But with the growth of the packet loss percentages, we can see that reno "saves" the same performance, while the cubic- after we pass the 20% shows us a big increase at the average time.
Moreover , when I tried to do an extra round with packet loss of 40% percentage, I gave up because of the long time it tooks.
*** It should be noted- The task was performed with a 1MB file ***


3950
3900
3850
3800
3750
3700
3650
3600
3550
3500
3450
3400
3350
3300
3250
3200
3150
3100
3050
3000
2950
2900
2850
2800
2750
2700
2650
2600
2550
2500
2450
2400
2350
2300
2250
2200
2150
2100
2050
2000
1950
1900
1850
1800
1750
1700
1650
1600
1550
1500
1450
1400
1350
1300
1250
1200
1150
1100
1050
1000
950
900
850
800
750
700
650
600
550
500
450
400
350
300
250
200
150
100
50
0
30%
25%
20%
15%
10%
without

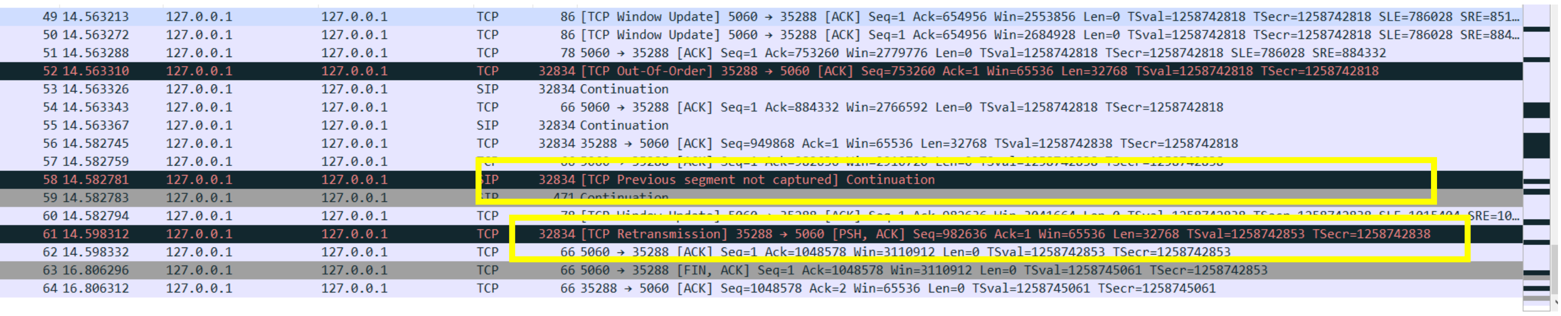
The explanation to these outcomes is obviously the cwnd increase function implementation:

The reno uses slow start, congestion avoidance, and fast retransmit triggered by triple duplicate ACKs. Reno uses packet loss to detect network congestion.

The cubic is less aggressive and more similar to "BIC". (The BIC is binary search increase congestion control implementation of TCP with an optimized congestion control algorithm. It uses binary search algorithm in an attempt to find the largest congestion window that will last maximum amount of time). The cubic function uses 2 components to window growth. The first is a concave portion where the window quickly ramps up to the window size as it was before the previous congestion event. Next is a convex growth where cubic probes for more bandwidth, slowly at first, than very rapidly. Cubic spends a lot of time at a plateau between the concave and convex growth region which allows the network to stabilize before cubic begins looking for more bandwidth.

Another more documentation of the differences for each round from the Wireshark :

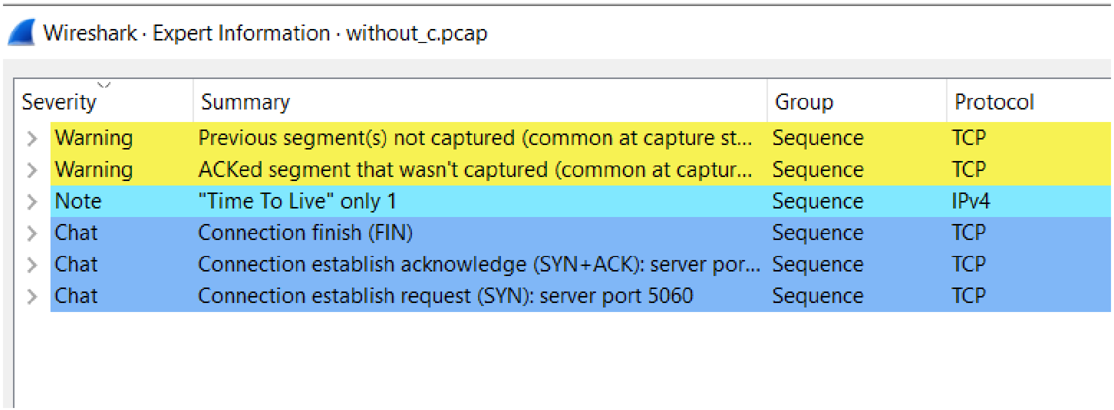
Here we see a screenshot from the 20% packet loss Wireshark pcap:



```
49 14.563213   127.0.0.1   127.0.0.1   TCP      86 [TCP Window Update] 5060 → 35288 [ACK] Seq=1 Ack=654956 Win=2553856 Len=0 TSval=1258742818 TSecr=1258742818 SLE=786028 SRE=851…
50 14.563272   127.0.0.1   127.0.0.1   TCP      86 [TCP Window Update] 5060 → 35288 [ACK] Seq=1 Ack=654956 Win=2684928 Len=0 TSval=1258742818 TSecr=1258742818 SLE=786028 SRE=884…
51 14.563288   127.0.0.1   127.0.0.1   TCP      78 5060 → 35288 [ACK] Seq=1 Ack=753260 Win=2779776 Len=0 TSval=1258742818 TSecr=1258742818 SLE=786028 SRE=884332
52 14.563310   127.0.0.1   127.0.0.1   TCP   32834 [TCP Out-Of-Order] 35288 → 5060 [ACK] Seq=753260 Ack=1 Win=65536 Len=32768 TSval=1258742818 TSecr=1258742818
53 14.563326   127.0.0.1   127.0.0.1   SIP   32834 Continuation
54 14.563343   127.0.0.1   127.0.0.1   TCP      66 5060 → 35288 [ACK] Seq=1 Ack=884332 Win=2766592 Len=0 TSval=1258742818 TSecr=1258742818
55 14.563367   127.0.0.1   127.0.0.1   SIP   32834 Continuation
56 14.582745   127.0.0.1   127.0.0.1   TCP   32834 35288 → 5060 [ACK] Seq=949868 Ack=1 Win=65536 Len=32768 TSval=1258742838 TSecr=1258742818
57 14.582759   127.0.0.1   127.0.0.1   TCP      66 5060 → 35288 [ACK] Seq=1 Ack=982636 Win=2916720 Len=0 TSval=1258742838 TSecr=1258742838
58 14.582781   127.0.0.1   127.0.0.1   SIP   32834 [TCP Previous segment not captured] Continuation
59 14.582783   127.0.0.1   127.0.0.1   SIP     471 Continuation
60 14.582794   127.0.0.1   127.0.0.1   TCP      78 [TCP Window Update] 5060 → 35288 [ACK] Seq=1 Ack=982636 Win=3041664 Len=0 TSval=1258742838 TSecr=1258742838 SLE=1015404 SRE=10…
61 14.598312   127.0.0.1   127.0.0.1   TCP   32834 [TCP Retransmission] 35288 → 5060 [PSH, ACK] Seq=982636 Ack=1 Win=65536 Len=32768 TSval=1258742853 TSecr=1258742838
62 14.598332   127.0.0.1   127.0.0.1   TCP      66 5060 → 35288 [ACK] Seq=1 Ack=1048578 Win=3110912 Len=0 TSval=1258742853 TSecr=1258742853
63 16.806296   127.0.0.1   127.0.0.1   TCP      66 5060 → 35288 [FIN, ACK] Seq=1 Ack=1048578 Win=3110912 Len=0 TSval=1258745061 TSecr=1258742853
64 16.806312   127.0.0.1   127.0.0.1   TCP      66 35288 → 5060 [ACK] Seq=1048578 Ack=2 Win=65536 Len=0 TSval=1258745061 TSecr=1258745061
```

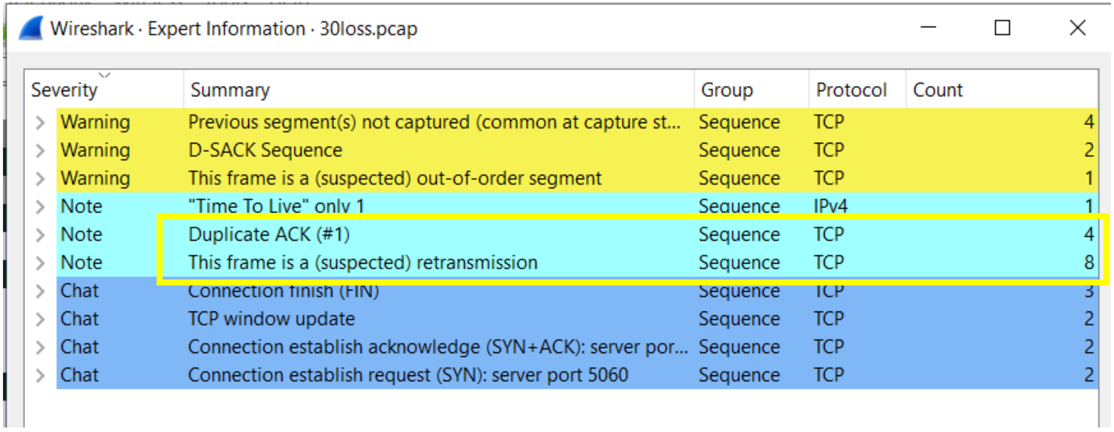
We can see a lost packet and also a retransmission of a lost packet.

We can also look and compare the amount of retransmissions that occurred in round without packet loss and round of 30% packet loss:



Wireshark · Expert Information · without_c.pcap

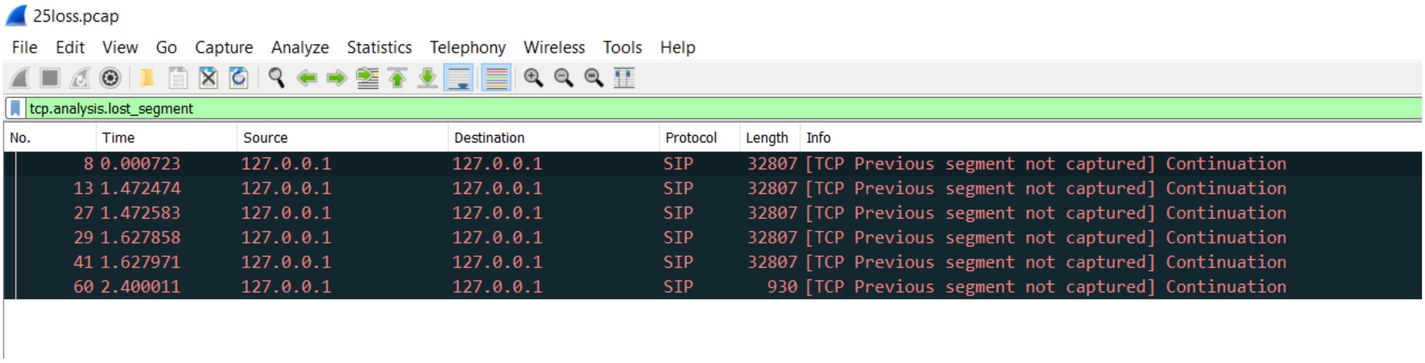
Severity	Summary	Group	Protocol
Warning	Previous segment(s) not captured (common at capture st...	Sequence	TCP
Warning	ACKed segment that wasn't captured (common at captur...	Sequence	TCP
Note	"Time To Live" only 1	Sequence	IPv4
Chat	Connection finish (FIN)	Sequence	TCP
Chat	Connection establish acknowledge (SYN+ACK): server por...	Sequence	TCP
Chat	Connection establish request (SYN): server port 5060	Sequence	TCP



Wireshark · Expert Information · 30loss.pcap

Severity	Summary	Group	Protocol	Count
Warning	Previous segment(s) not captured (common at capture st...	Sequence	TCP	4
Warning	D-SACK Sequence	Sequence	TCP	2
Warning	This frame is a (suspected) out-of-order segment	Sequence	TCP	1
Note	"Time To Live" only 1	Sequence	IPv4	1
Note	Duplicate ACK (#1)	Sequence	TCP	4
Note	This frame is a (suspected) retransmission	Sequence	TCP	8
Chat	Connection finish (FIN)	Sequence	TCP	3
Chat	TCP window update	Sequence	TCP	2
Chat	Connection establish acknowledge (SYN+ACK): server por...	Sequence	TCP	2
Chat	Connection establish request (SYN): server port 5060	Sequence	TCP	2

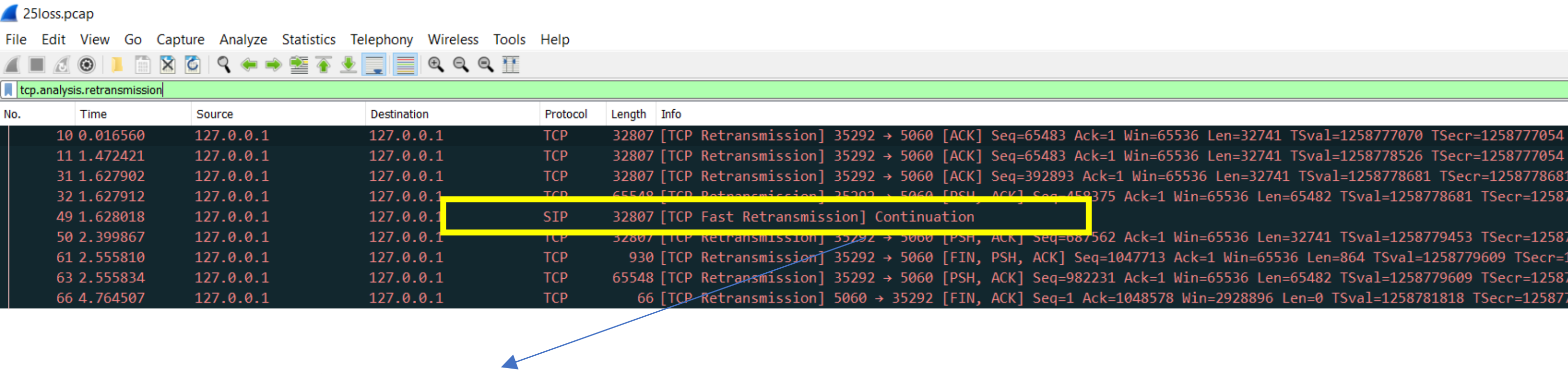
Let's look about some of the packet loss and retransmission occurred at 25% packet loss:



25loss.pcap

tcp.analysis.lost_segment

```
No.     Time        Source      Destination  Protocol  Length Info
      8 0.000723    127.0.0.1   127.0.0.1    SIP        32807 [TCP Previous segment not captured] Continuation
     13 1.472474    127.0.0.1   127.0.0.1    SIP        32807 [TCP Previous segment not captured] Continuation
     27 1.472583    127.0.0.1   127.0.0.1    SIP        32807 [TCP Previous segment not captured] Continuation
     29 1.627858    127.0.0.1   127.0.0.1    SIP        32807 [TCP Previous segment not captured] Continuation
     41 1.627971    127.0.0.1   127.0.0.1    SIP        32807 [TCP Previous segment not captured] Continuation
     60 2.400011    127.0.0.1   127.0.0.1    SIP          930 [TCP Previous segment not captured] Continuation
```



25loss.pcap

tcp.analysis.retransmission

```
No.     Time        Source      Destination  Protocol  Length Info
     10 0.016560    127.0.0.1   127.0.0.1    TCP        32807 [TCP Retransmission] 35292 → 5060 [ACK] Seq=65483 Ack=1 Win=65536 Len=32741 TSval=1258777070 TSecr=1258777054
     11 1.472421    127.0.0.1   127.0.0.1    TCP        32807 [TCP Retransmission] 35292 → 5060 [ACK] Seq=65483 Ack=1 Win=65536 Len=32741 TSval=1258778526 TSecr=1258777054
     31 1.627902    127.0.0.1   127.0.0.1    TCP        32807 [TCP Retransmission] 35292 → 5060 [ACK] Seq=392893 Ack=1 Win=65536 Len=32741 TSval=1258778681 TSecr=125877868
     32 1.627912    127.0.0.1   127.0.0.1    TCP        65548 [TCP Retransmission] 35292 → 5060 [PSH, ACK] Seq=458375 Ack=1 Win=65536 Len=65482 TSval=1258778681 TSecr=12587
     49 1.628018    127.0.0.1   127.0.0.1    SIP        32807 [TCP Fast Retransmission] Continuation
     50 2.399867    127.0.0.1   127.0.0.1    TCP        32807 [TCP Retransmission] 35292 → 5060 [PSH, ACK] Seq=687562 Ack=1 Win=65536 Len=32741 TSval=1258779453 TSecr=12587
     61 2.555810    127.0.0.1   127.0.0.1    TCP          930 [TCP Retransmission] 35292 → 5060 [FIN, PSH, ACK] Seq=1047713 Ack=1 Win=65536 Len=864 TSval=1258779609 TSecr=1
     63 2.555834    127.0.0.1   127.0.0.1    TCP        65548 [TCP Retransmission] 35292 → 5060 [PSH, ACK] Seq=982231 Ack=1 Win=65536 Len=65482 TSval=1258779609 TSecr=12587
     66 4.764507    127.0.0.1   127.0.0.1    TCP           66 [TCP Retransmission] 5060 → 35292 [FIN, ACK] Seq=1 Ack=1048578 Win=2928896 Len=0 TSval=125781818 TSecr=12587
```

Case of duplicate Ack