Reliable Transport

Spencer Wood

February 24, 2017

I implemented TCP and performed analysis on the relationship between speed and reliability. The first series of tests were basic transmission tests between the two nodes. The second involved an implementation of fast retransmit and how that affected file transfers between the two nodes. The third series of tests examines the relationship between window size, throughput, and average queueing delay.

All tests were done on a simple 2 node network with a bi-directional link. The bandwidth of the links was 10Mbps and the propagation delay was 10ms.

1 Basic Tests

1.1 Test #1

This test involved transmitting a 10,000 byte text file from the first node to the second node with a window size of 3,000 bytes and with various values of packet loss (0%, 10%, 20%, and 50%).

Table 1 shows an overview of the results.

Table 1: Test #1 results

Packet Loss	Time to Transfer File
0%	0.0832s
10%	1.1048s
20%	3.1056s
50%	11.1456s

Listing 1: Truncated output for 0% packet loss

```
...
0.0624 n1 (1) sending TCP segment to 2 for 9000
0.0632 n1 (1) handling ACK 8000
0.064 n1 (1) handling ACK 9000
0.0732 n2 (2) received TCP segment from 1 for 9000
0.0732 application got 1000 bytes
0.0732 n2 (2) sending TCP ACK to 1 for 10000
0.0832 n1 (1) handling ACK 10000
File transfer correct!
```

Listing 2: Truncated output for 10% packet loss

```
...
1.074 n2 (2) sending TCP ACK to 1 for 9000
1.084 n1 (1) handling ACK  9000
1.084 n1 (1) sending TCP segment to 2 for 9000
1.0948 n2 (2) received TCP segment from 1 for 9000
1.0948 application got 1000 bytes
1.0948 n2 (2) sending TCP ACK to 1 for 10000
1.1048 n1 (1) handling ACK  10000
File transfer correct!
```

Listing 3: Truncated output for 20% packet loss

```
...
2.0848 n1 (1) sending TCP segment to 2 for 9000
3.0848 n1 (1) sending TCP segment to 2 for 9000
3.0848 n1 (1) retransmission timer fired
3.0956 n2 (2) received TCP segment from 1 for 9000
3.0956 application got 1000 bytes
3.0956 n2 (2) sending TCP ACK to 1 for 10000
3.1056 n1 (1) handling ACK  10000
File transfer correct!
```

Listing 4: Truncated output for 50% packet loss

```
...
11.1248 n1 (1) handling ACK  8000
11.1248 n1 (1) sending TCP segment to 2 for 8000
11.1248 n1 (1) sending TCP segment to 2 for 9000
11.1356 n2 (2) received TCP segment from 1 for 8000
11.1356 application got 2000 bytes
11.1356 n2 (2) sending TCP ACK to 1 for 10000
11.1456 n1 (1) handling ACK  10000
File transfer correct!
```

1.2 Test #2

This test involved transmitting a 514,520 byte pdf from the first node to the second node with a window size of 10,000 bytes and with packet loss of 0% and 50%.

Table 2 shows an overview of the results.

Table 2: Test #2 results

Packet Loss	Time to Transfer File
0%	1.084416s
50%	329.0624s

Listing 5: Truncated output for 0% packet loss

```
...
1.074416 application got 520 bytes
1.074416 n2 (2) sending TCP ACK to 1 for 514520
1.0816 n1 (1) handling ACK 511000
1.0824 n1 (1) handling ACK 512000
1.0832 n1 (1) handling ACK 513000
1.084 n1 (1) handling ACK 514000
1.084416 n1 (1) handling ACK 514520
File transfer correct!
```

Listing 6: Truncated output for 50% packet loss

```
...
328.0416 n1 (1) retransmission timer fired
329.0416 n1 (1) sending TCP segment to 2 for 511000
329.0416 n1 (1) retransmission timer fired
329.0524 n2 (2) received TCP segment from 1 for 511000
329.0524 application got 0 bytes
329.0524 n2 (2) sending TCP ACK to 1 for 514520
329.0624 n1 (1) handling ACK 514520
File transfer correct!
```

2 Fast Retransmit

I implemented a simple fast transport protocol that resends dropped packets after 4 duplicate ACKs are received in a row. To measure the impact this has on file transfer speed, I did two tests, each with a window size of 10,000 bytes and a loss rate of 20%. For the first test fast retransmit was turned off and for the second in t was turned on. In each case the file transfered was a 514,520 byte pdf.

Table 3 shows an overview of the results. Fast retransmit cut down the transfer time by 90%.

Table 3: Test #2 results

Fast Retransmit	Packet Loss	Time to Transfer File
No	20%	67.91402s
Yes	20%	6.74282s

Listing 7: Truncated output for no fast retransmit

```
...
66.8936 n1 (1) handling ACK 514000
67.8936 n1 (1) sending TCP segment to 2 for 514000
67.8936 n1 (1) retransmission timer fired
67.90401600000001 n2 (2) received TCP segment from 1 for 514000
67.90401600000001 application got 520 bytes
67.90401600000001 n2 (2) sending TCP ACK to 1 for 514520
67.91401600000002 n1 (1) handling ACK 514520
File transfer correct!
```

Listing 8: Truncated output for fast retransmit

```
...
6.730415999999962 n2 (2) received TCP segment from 1 for 514000
6.730415999999962 application got 0 bytes
6.730415999999962 n2 (2) sending TCP ACK to 1 for 514520
6.7328159999999615 n2 (2) received TCP segment from 1 for 514000
6.7328159999999615 application got 0 bytes
6.7328159999999615 n2 (2) sending TCP ACK to 1 for 514520
6.742815999999961 n1 (1) handling ACK 514520
File transfer correct!
```

3 Experiments

The final series of tests examined the relationship between window size and queueing delay / throughput. Two experiments were performed.

The first measured the affect of window size on average packet queueing delay. The PDF was transferred from the first node to the second node using window sizes of 1000, 2000, 5000, 10000, 15000, and 20000 bytes. Packet loss was set to 0%. The average queueing delay was measured during each run. The results are shown in figure 1. As window size goes up, queueing delay appears to increase exponentially.

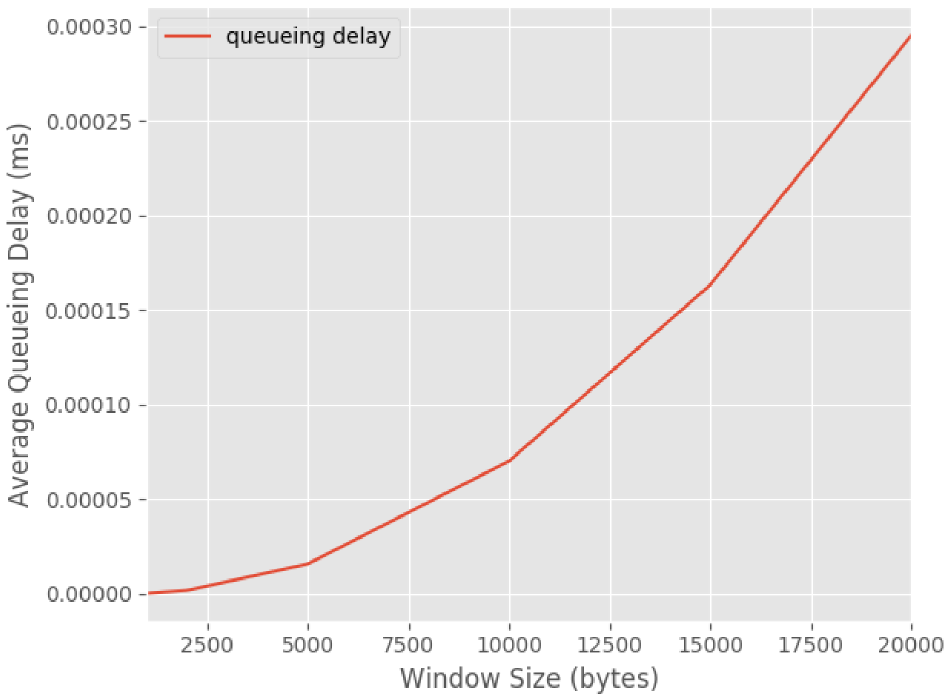

Figure 1: Average queueing delay

The second test measured the affect of window size on throughput. Throughput is measured as the number of bits transferred over the total number of seconds the transfer took. The PDF was transferred from the first node to the second node using window sizes of 1000, 2000, 5000, 10000, 15000, and 20000 bytes. Packet loss was set to 0%. The throughput was measured during each run. The results are shown in figure 2. As window size increases, throughput appears to increase linearly.

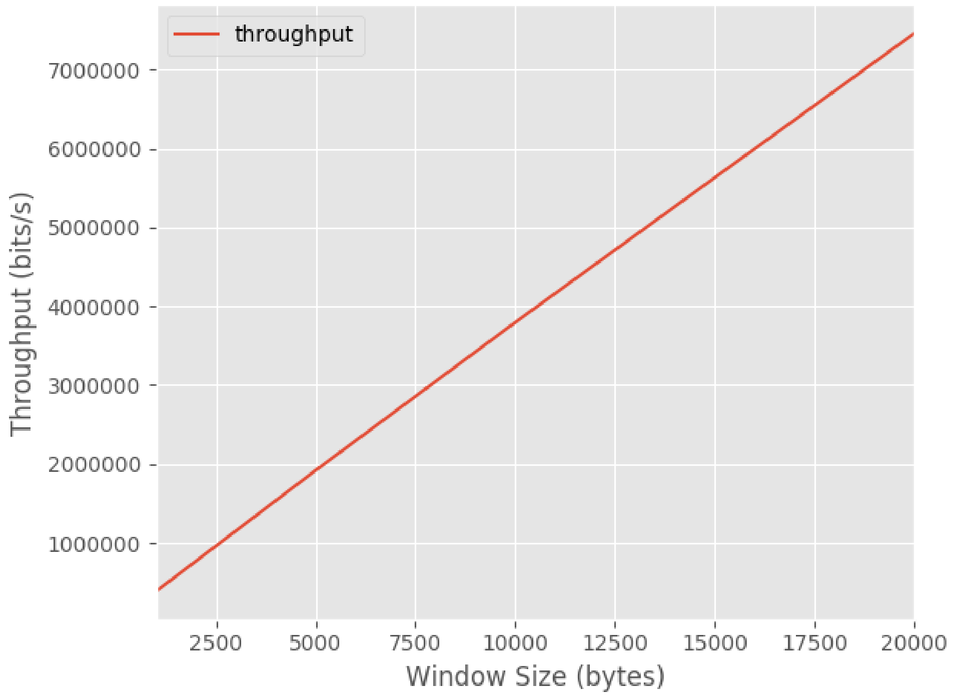

Figure 2: Throughput