# Performance Analysis on TCP Variants

Tuoxin Li

Khoury College of CS

Northeastern University

Roux Institute

li.tuox@northeastern.edu

## ABSTRACT

The performance of different TCP variants, including Tahoe, Reno, NewReno, and Vegas, were analyzed in the presence of a Constant Bit Rate (CBR) flow using ns2. The aim of this study was to investigate the performance of different TCP variants under different network conditions, including varying queuing disciplines used by nodes.

The study involved using different TCP variants in the ns2 simulator to evaluate their performance in the presence of a CBR flow. The results showed that the different TCP variants had different performance characteristics under different network conditions, and that the choice of TCP variant can have a significant impact on overall network performance. Additionally, the study examined the influence of queuing discipline used by nodes on the overall throughput of flows, and found that the choice of queuing discipline can also have a significant impact on network performance.

Overall, the study provides valuable insights into the performance of different TCP variants in different network conditions, and highlights the importance of careful selection of TCP variant and queuing discipline for optimal network performance.

## 1 INTRODUCTION

This report will exam the performance of the following TCP variants.

Tahoe TCP is the oldest of the TCP variants, and was designed to provide reliable data transfer in the presence of packet loss.

Reno TCP is a modification of Tahoe TCP, which aims to reduce the effects of TCP's slow start mechanism.

NewReno TCP is a further modification of Reno TCP, and aims to provide better performance in the presence of multiple packet losses.

Vegas TCP is a newer TCP variant, which uses a different congestion control algorithm to maintain high throughput while minimizing packet loss.

## 2 Performance under congestion

In this experiment, the performance of the four TCP variants are discussed(Tahoe, Reno, NewReno and Vegas) in the presence of CBR flow. Figure 1 shows the topology that is used.

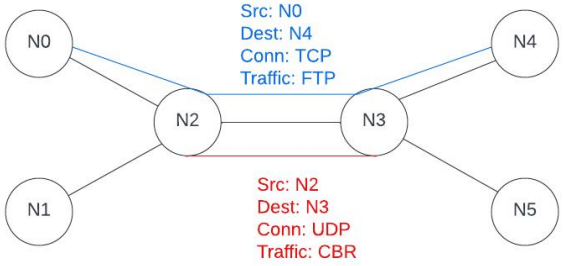


Figure 1

By using network simulator, a single stream from N0 to a sink at N4 is added, at meanwhile, a CBR source at N2 and a sink at N3 is added. By default, the packet size of TCP is 1000 and at rate of 1 Mbps. CBR flow will be increased to exam the performance of the TCP variants. Three sub-experiments regarding to changing the CBR flow are discussed in the following sections.

### 2.1 Steadily increase CBR flow

In this experiment, 5 CBR are set and will be added during the time when running the program.

| CBR name | Packet size | CBR flow | Added time |
|---|---|---|---|
| Cbr1 | 1000 | 5 MB | At 1.0 |
| Cbr2 | 1000 | 10 MB | At 4.0 |
| Cbr3 | 1000 | 20 MB | At 7.0 |
| Cbr4 | 1000 | 40 MB | At 10.0 |
| Cbr5 | 1000 | 80 MB | At 13.0 |

Table 1

*2.1.1 Throughput*

Based on figure 2, Reno and NewReno have been shown to perform better in this scenario. Tahoe TCP, which was one of the first TCP variants developed, has shown to have poor performance in congested networks. It is discussed by other researchers that Vegas TCP uses a different approach to congestion control by monitoring the queuing delay instead of packet loss, and it performs worse when the middle rate of CBR is added but performs better when high rate of CBR is added.

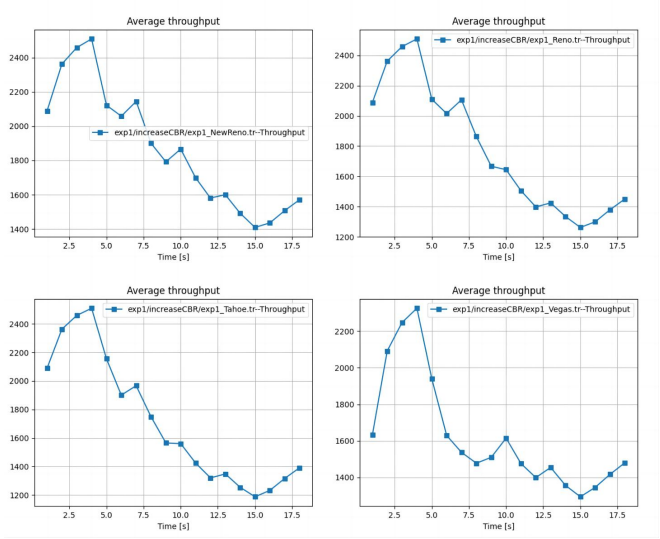


Figure 2

*2.1.2 Package drop*

During the experiment, there are 101734 records regarding package send or receive for the 4 TCP variants. The number of drop in each scenario is shown in table 2. According to the table, the Reno and NewReno TCP have the fewest drop.

| TCP variants | Reno | Tahoe | Vegas | NewReno |
|---|---|---|---|---|
| Number | 13728 | 13762 | 13816 | 13750 |

Table 2

*2.1.3 Latency*

According to figure 3, the difference of latency among the variant is not obvious. Generally, Vegas TCP has the lowest latency.

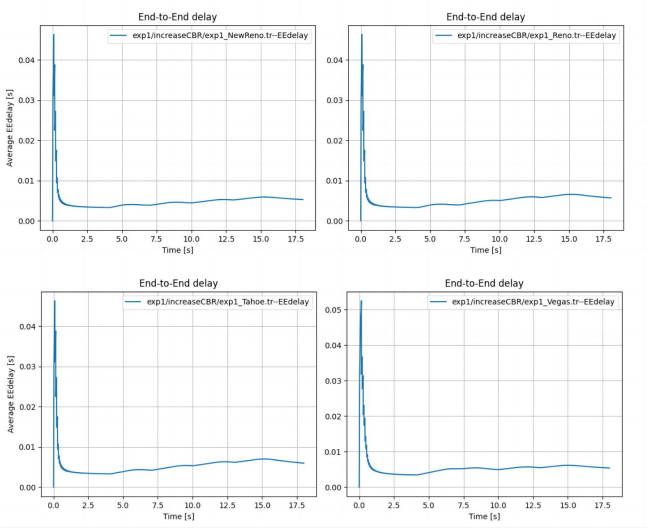


Figure 3

## 2.2 High CBR flow

In this experiment, there is only one CBR with 100 Mbps is added at time of 1.0.

*2.2.1 Throughput*

Based on figure 4, it is surprisingly that Vegas TCP has shown to perform the worst among the variants.

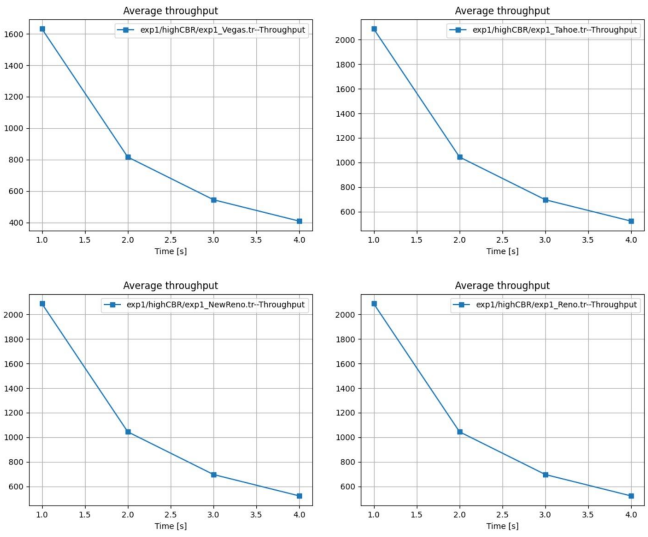


Figure 4

*2.2.2 Package drop*

As the congestion is awful, the four variants all have more than 20000 drops in terms of the total 85347 records.

*2.2.3 Latency*

As is shown in figure 5, Vegas TCP has shown to perform the worst latency, while the other 3 variants haven't shown to vary.

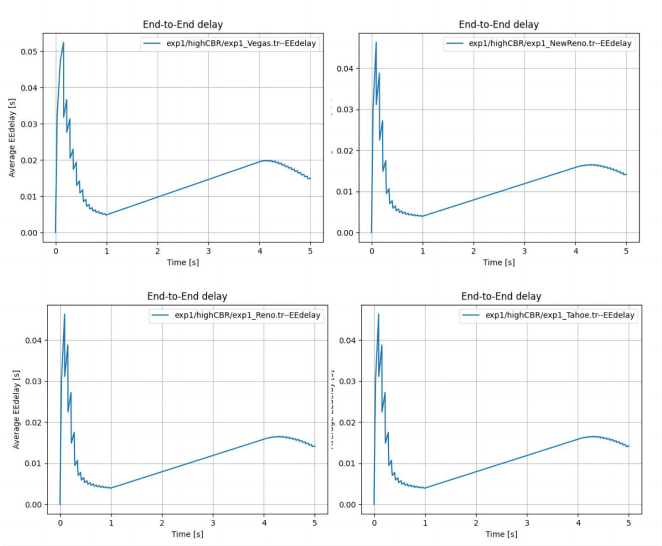


Figure 5

## 2.3 Low CBR flow

In this experiment, there is only one CBR with 10 Mbps is added at time of 1.0.

*2.3.1 Throughput*

When congestion is low, Vegas TCP performs the worst throughput, and the throughput among the other variants do not significantly vary.

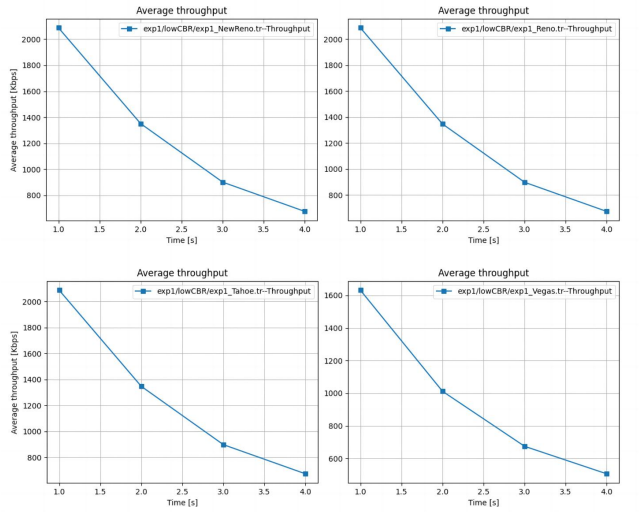


Figure 6

*2.3.2 Package drop*

During the experiment, there are 17373 records regarding package send or receive for the 4 TCP variants. The number of drop in each scenario is shown in table 3. According to the table, Vegas TCP has shown the best performance to occur package drop.

| TCP variants | Reno | Tahoe | Vegas | NewReno |
|---|---|---|---|---|
| Number | 46 | 50 | 26 | 47 |

Table 3

*2.2.3 Latency*

As is shown in figure 7, Reno TCP has shown to perform the best latency.

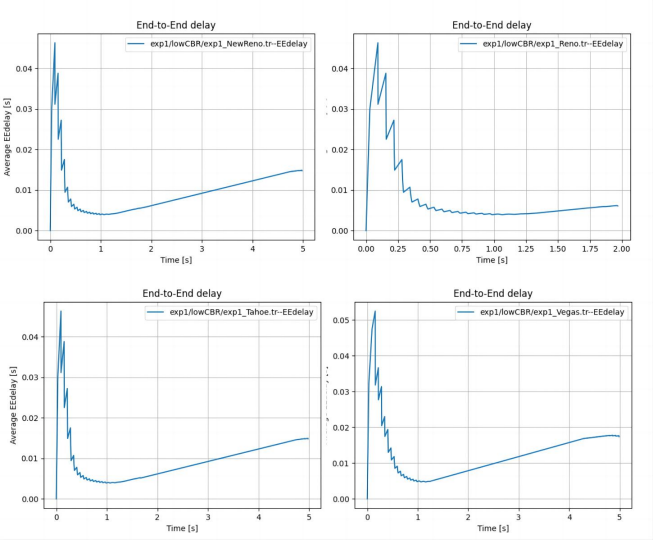


Figure 7

## 2.4 Conclusion

The result of the performance can be very differed in different traffic situations. During the three experiments, we can observe that Vegas can have low package drop while show unsatisfactory on throughput and latency when congestion is low. Reno and NewReno seem to always perform good in terms of throughput and latency when congestion occurs. Also, Tahoe TCP, which was one of the first TCP variants developed, is known to have poor performance in congested networks due to its slow response to congestion.

In conclusion, it is difficult to determine which TCP variant will perform better in different scenarios. The performance of each TCP variant depends on various factors such as the network topology, the amount of traffic, and the queuing discipline used by nodes.

# 3 Fairness Between TCP Variants

In this experiment, fairness between different TCP variants are discussed(Reno/Reno, NewReno/Reno, Vegas/Vegas and NewReno/Vegas). Figure 8 shows the topology that is used.

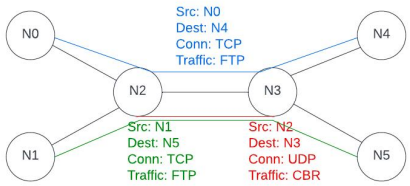


Figure 8

By using network simulator, stream from N0 to a sink at N4 and stream from N1 to a sink at N5 are added, at meanwhile, a CBR source at N2 and a sink at N3 is added. By default, the packet size of TCP is 1000 and at rate of 1 Mbps. CBR flow will keep increasing as is used in sub-experiment 2.1 during this experiment. Four sub-experiments regarding to different combinations of the variants are discussed in the following sections.

## 3.1 Reno/Reno

Figure 9 shows the throughput of Reno TCP 1 and Reno TCP 2(from top). The result shows the two TCP share very similar throughput, and the combination of this variant is fair.

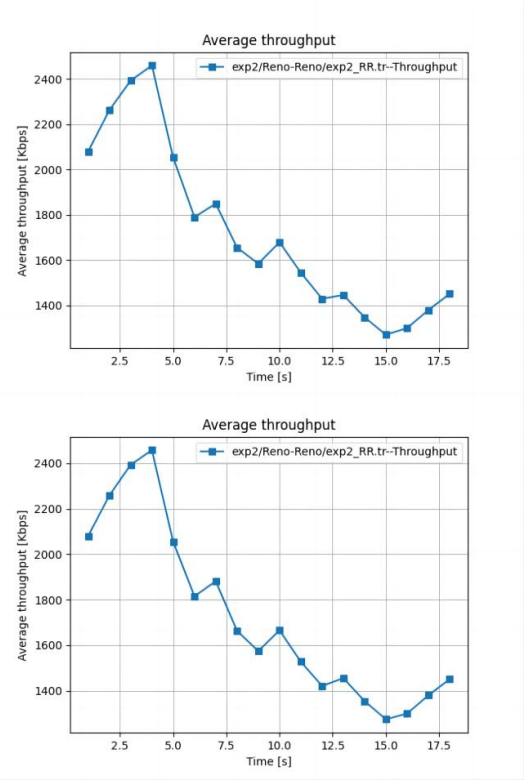


Figure 9

## 3.2 NewReno/Reno

Due to the page limit, the graphs can be found in my github repository. Result shows that when congestion is low, the two TCP are in a very fair situation, while when congestion is high, Reno TCP will use more traffic. Generally, the two TCP share very similar throughput, the combination of these variants is fair.

## 3.3 Vegas/Vegas

The result shows when congestion is low, the two TCP are in a very fair situation, while when congestion is high, one Vegas TCP could use more traffic. Generally, the two TCP share very similar throughput, the combination of these variants is fair.

## 3.4 NewReno/Vegas

In this scenario, the result shows the unfairness between this combination. Figure 10 shows the throughput of the NewReno TCP, and figure 11 shows the throughput of the Vegas TCP. When congestion occurs, NewReno TCP generally uses more traffic when congestion is at low level. While at high level congestion, Vegas shows the aggressiveness to use more network resources.

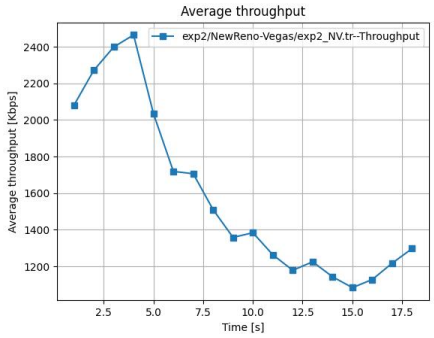


Figure 10

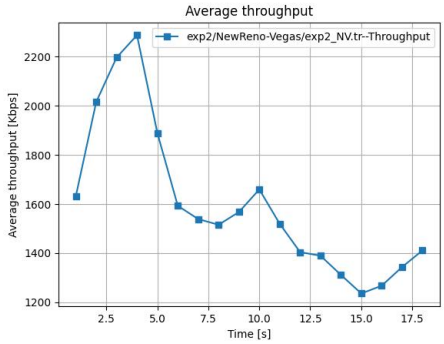


Figure 11

Unfairness between NewReno TCP and Vegas TCP can occur due to differences in their congestion control mechanisms. NewReno TCP uses a conservative congestion control mechanism that reduces the congestion window size by half upon detecting a loss event. In contrast, Vegas TCP uses a more aggressive congestion control mechanism that increases the congestion window size at a faster rate and maintains a higher congestion window than NewReno TCP.

When both NewReno and Vegas TCP flows compete for the same network resources, Vegas TCP may obtain a larger share of the available bandwidth due to its more aggressive congestion control mechanism, resulting in unfairness towards NewReno TCP flows.

## 3.5 Conclusion

The unfairness among the combinations could happen due the congestion control mechanisms and it depends on the specific

scenario. For example, when the congestion is at high level, Vegas TCP could use more network resources than NewReno TCP due to its aggressive congestion control mechanism.

## 4 Influence of Queuing

In this experiment, queuing disciplines like DropTail and Random Early Drop (RED) are implemented in ns2 with TCP Reno and SACK. The fairness and performance of the network by the different queuing algorithm are discussed. Figure 8 shows the topology that is used. Similarly with the previous experiments, we have from N0 to a sink at N4 using TCP Reno or SACK in the sub-experiments, and from N1 to a sink at N5 using UDP. The whole program finishes at time of 5.0, FTP is added at time of 0.1 and CBR is added at time of 0.3.

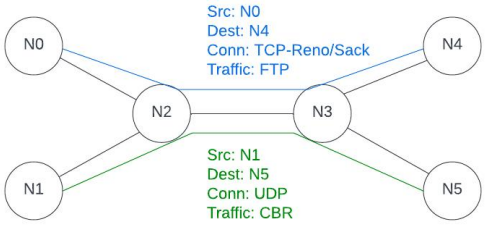


Figure 12

### 4.1 DropTail/Red comparison under TCP Reno

*4.1.1 fair or unfair bandwidth*

Figure 13 is the result of the throughput of the 4 sub-experiments using TCP Reno. It can be observed that when using drop tail queuing algorithm, the bandwidth usage is not fair, since the average throughput of TCP significantly drops after introducing CBR. While by taking RED queuing algorithm, TCP is not noticeably affected after introducing CBR. Thus, the bandwidth usage is fair in this scenario.

*4.1.2 latency difference between DropTail and RED*

Figure 14 and figure 16 show the latency when using TCP Reno and TCP SACK. RED algorithm has shown to contribute to decreasing the latency to TCP comparing to drop tail algorithm in this scenario. RED can provide better end-to-end latency compared to DropTail in congested networks because it can prevent buffer overflow and prevent the retransmission of packets, which reduces the overall end-to-end latency for the flows.

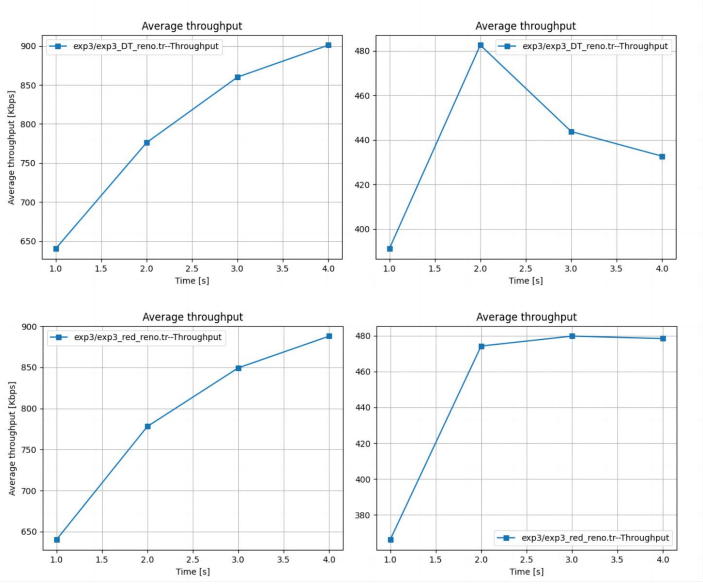


Figure 13 *Left-right: CBR-TCP, Top-down: Droptail-RED*

*4.1.3 TCP reaction to the creation of the CBR flow*

The TCP reaction to the creation of the CBR flow can really depend on the specific scenario. For example, Figure 13 shows that the average throughput of TCP Reno significantly drops after the creation of CBR taking DropTail algorithm, and the throughput of TCP Reno is not noticeably affected to the creation of CBR taking RED algorithm. However, the situation can differ in TCP SACK scenario. Figure 15 shows a very opposite result to the TCP Reno scenario.

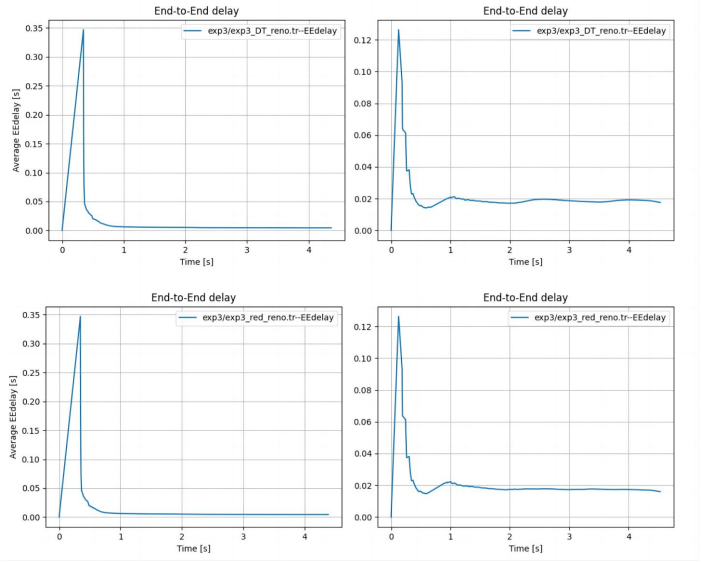


Figure 14 *Left-right: CBR-TCP, Top-down: Droptail-RED*

## 4.2 DropTail/Red comparison under TCP SACK

*4.1.4 RED and SACK*

Figure 15 shows that when taking RED algorithm and using TCP SACK, the creation of CBR can greatly affect the TCP. The bandwidth usage is not fair in this scenario. Thus, if we consider the fairness of the network, RED can be a bad idea while dealing with SACK. However, as RED is designed to reduce packet loss, and using RED with SACK can help to improve network performance and reduce packet loss and congestion. In this case, it could be a good idea.

## 4.3 Conclusion

In conclusion, the performance of TCP variants in the presence of a CBR flow depends on various factors such as average throughput, drops, and latency. It is important to carefully evaluate the performance of different TCP variants and queuing algorithms in specific network scenarios to make informed decisions.

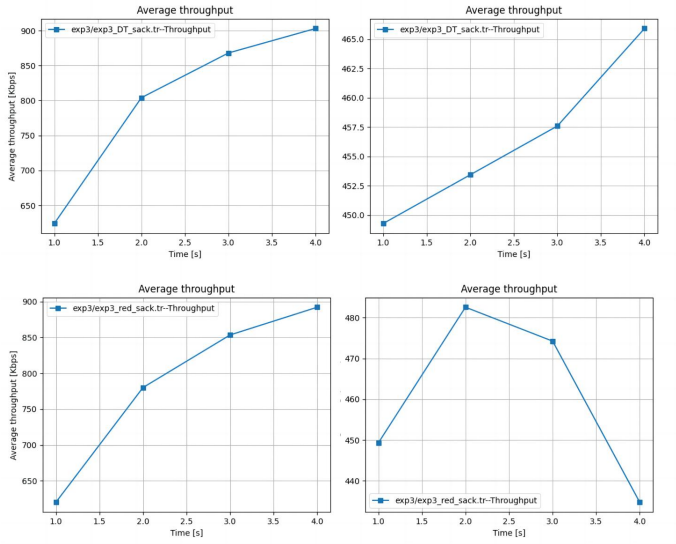


**Figure 15** *Left-right: CBR-TCP, Top-down: Droptail-RED*

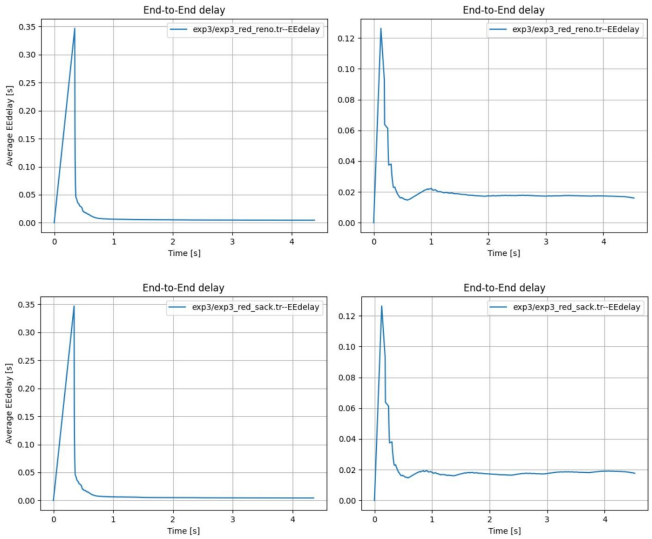


**Figure 16** *Left-right: CBR-TCP, Top-down: Droptail-RED*