|  |  |
| --- | --- |
|  | **University of Dhaka**  **Department of Computer Science and Engineering**  **CSE 3111 – Computer Networking Laboratory Credits: 1.5 Batch: 26/3rd Year 1st Sem 2022**  Instructors: Prof. Dr. Md. Abdur Razzaque (AR), Mr. Md. Mahmudur Rahman (MRR), Mr. Md. Ashraful Islam (MAI) and Mr. Md. Fahim Arefin (FA) |

**Lab Assignment # 6**

|  |
| --- |
| **Name of the Experiment:** Implementation of TCP Reno congestion control algorithm.  **Objective:** To understand and implement the TCP Reno congestion control algorithm, and compare it with TCP Tahoe. |

**Theory :**

TCP Reno is a congestion control algorithm used in the Transmission Control Protocol (TCP) to manage network congestion. It was named after the city of Reno, Nevada, where the algorithm was first presented at a conference in 1990. TCP Reno is an extension of the earlier TCP Tahoe algorithm, and it introduces a new mechanism called "fast recovery" to improve network performance.

TCP Reno operates in four phases: slow start, congestion avoidance, fast retransmit, and fast recovery.

**1. Slow Start:** When a connection is established, the congestion window size is initially set to 1. The window size is increased by 1 for each ACK received, doubling the window size every round trip time (RTT) until the slow start threshold is reached.

**2. Congestion Avoidance:** Once the slow start threshold is reached, the congestion window size is increased linearly, by 1/cwnd for each ACK received, where cwnd is the current congestion window size.

**3. Fast Retransmit:** When three duplicate ACKs are received, TCP Reno assumes that a packet has been lost and immediately retransmits the lost packet.

**4. Fast Recovery:** After retransmitting the lost packet, TCP Reno enters the fast recovery phase. The congestion window size is halved and **3 is added to it**, and then kept constant until all lost packets are retransmitted. After that, the congestion avoidance phase is entered, and the congestion window size is increased linearly.

TCP Tahoe and TCP Reno are similar in many ways, but there are some key differences between them.

**1. Fast Recovery:**

The most significant difference between TCP Tahoe and TCP Reno is the way they handle packet loss. In TCP Tahoe, when a packet loss is detected, the congestion window is reduced to 1 and the slow start phase is entered. This means that the congestion window size is increased exponentially until the slow start threshold is reached, and then increased linearly.

In TCP Reno, a fast recovery phase is added. When a packet loss is detected, the congestion window is halved, and the fast recovery phase is entered. In this phase, the congestion window size is kept constant until all lost packets are retransmitted. After that, the congestion avoidance phase is entered, and the congestion window size is increased linearly.

**2. Congestion Window Size:**

In TCP Tahoe, the congestion window size is reduced to 1 when a packet loss is detected. This can lead to a significant decrease in throughput, especially in high-bandwidth networks.

In TCP Reno, the congestion window size is halved when a packet loss is detected. This means that the throughput is not reduced as much as in TCP Tahoe, and the network can recover more quickly from congestion.

In summary, the main difference between TCP Tahoe and TCP Reno is the way they handle packet loss. Figure 3.52 illustrates the evolution of TCP’s congestion window for both TCP Tahoe and Reno.

Chart, line chart

Description automatically generated

In this figure, the threshold is initially equal to 8 MSS. For the first eight transmission rounds, Tahoe and Reno take. identical actions. The congestion window climbs exponentially fast during slow start and hits the threshold at the fourth round of transmission. The congestion window then climbs linearly until a triple duplicate- ACK event occurs, just after transmission round 8. Note that the congestion window is 12 MSS when this loss event occurs.

The value of ssthresh is then set to 0.5 \*cwnd = 6 MSS. Under TCP Reno, the congestion window is set to cwnd = 9 MSS and then grows linearly.

Under TCP Tahoe, the congestion window is set to 1 MSS and grows exponentially until it reaches the value of ssthresh, at which point it grows linearly.

**Tasks :**

The implementation of TCP Reno should include the following :

- **Slow start:** Increase the congestion window size exponentially until the threshold is reached.

- **Congestion avoidance:** Increase the congestion window size linearly after the threshold is reached.

- **Fast retransmit:** Retransmit lost packets immediately after receiving three duplicate ACKs.

- **Fast recovery:** Keep the congestion window size constant after retransmitting lost packets.

Afterwards, compare the results with those obtained in the previous assignment on TCP Tahoe. Analyze the differences and similarities between the two algorithms.

**Deliverables:**

- Source code for the client-server application with TCP Reno

- Run simulations with TCP Reno and collect performance metrics such as throughput, packet loss rate, and round-trip time (RTT).

- A report that documents the design and implementation of the algorithms, as well as the performance comparison under different network conditions between TCP Tahoe and TCP Reno [ Do not include code in the report ]

**References:**

* <https://www.geeksforgeeks.org/tcp-tahoe-and-tcp-reno>
* <https://www.geeksforgeeks.org/tcp-reno-with-example/>
* <https://www.cs.cmu.edu/~prs/15-441-F14/lectures/rec05-congestion.pdf>
* <https://en.wikipedia.org/wiki/TCP_congestion_control#TCP_Tahoe_and_Reno>