Implementing TCP Reno

TCP Reno is a version of the Transmission Control Protocol (TCP) that introduces several mechanisms to improve network congestion control and data transmission efficiency. It builds on the earlier TCP Tahoe implementation and incorporates additional features, like fast recovery, to handle network congestion more effectively.

The first part of implementing TCP Reno is actually implementing TCP Tahoe. TCP Tahoe consists of three separate parts – Slow Start, Additive increase/multiplicative decrease (AIMD), and Fast Retransmit.

For slow start, every single time we received a unique ack, the value of our current window was increased by the size of the acked message. This continues until our current window reaches our max window. If we see any sort of packet loss, however, we halve the value of our current window.

AIMD was implemented by increasing/decreasing our current window respectively on packet loss/packet acknowledgement.

Fast retransmit was trivial to implement. When we detected three duplicate acks we immediately retransmitted from the retransmission queue. As opposed to TCP Tahoe where slow start would begin at this point, TCP Reno relies on fast recovery which is essentially halving the current window. This is for packet loss detected by duplicate acks. If we detect packet loss via the expiration of our rto timer, we reset our current window to the initial value. Fast recovery is intended to rapidly reduce network congestion.