CS 456/656 - Computer Networks **Fall 2018**

Lecture 13 - 18: October 29 - November 13, 2018

Lecturer: Kamal Zille Huma *Notes By: Harsh Mistry*

13.1 Transport Layer Continued

- Ack indicates the expected packet from the other side
- Timeout Interval = EstimatedRTT + 4 * DEVRTT

13.1.1 TCP

- Retransmissions are triggered by Timeout and Duplicate Acks
- TCP added pipelined segments, cumulative acks, and single transmission timer
- Sequence number is the byte stream number of first data byte in segment
- Fast Retransmit : If sender receives 3 ACKs for same data, resend unacked segment with smallest sequence number

13.1.1.1 TCP Flow Control

- Receiver controls sender, so sender won't overflow receiver's buffer
- Receiver "advertises" free buffer space by including rwnd value in TCP header of receiver
- rwnd is typically total buffer size - RcvBuffer
- RcvBuffer is typically 4096, but some systems may dynamically change that

13.1.1.2 Connection Management

- Before exchanging data, sender/receeiver need to agree to establish connection and agree on connection parameters
- Two-way handshake can not work sure to variable delays , retransmitted messages, and message re-ordering
- Three-way handshaking is introduced to resolve theses issues
  1. Client sends SYN Request (Synbit = 1, Seq bit = x)
  2. Server chooses seq starting point and sends SYN ACK (SynBit = 1, seq = y, AckBit =1, AckNum = x+1)
  3. Client sends Ack (ACKBit = 1, ACKnum =y+1)

- Closing a connection
  1. Client sends packet with FinBit (FinBit = 1, Seq = x)
  2. Server Acks packet
  3. Server Keeps sending data till its ready to close
  4. Server sends FinBit and can no longer send data (FinBit = 1, Seq = y)
  5. Client Acks Fin Bit Packet
  6. Connection is closed

13.1.1.3 Congestion Control Principles

- informally: "too many sources sending too much data too fast for network to handle"
- Different from flow control
- Can lead to lost packets and long delays

13.1.1.4 TCP Congestion Control

- sender increases transmission rate (window size), probing for usable bandwidth, until loss occurs
- **Additive Increase** : increase cwnd by 1 MSS every RTT until loss detected
- **Multiplicative Decrease** : cut cwnd in half after loss
- Sender Limits transmission

$$\text{LastByteSent - LastByteAcked} \leq \text{cwnd}$$

$$\text{send rate} \approx \frac{cwnd}{RTT}\text{Bytes/sec}$$

- TCP slow start is when cwnd is doubled every RTT. The initial rate is slow, but ramps up exponetially fast
- Loss is indicated by 3 duplicate acks
- TCP RENO cuts window in half and the grows linearly
- TCP Tahoe resets cwnd to 1

13.1.1.5 TCP Throught

- TCP Throughput = $\frac{3}{4}\frac{W}{RTT}$ bytes/sec
- TCP Throughput = $\frac{1.22 \times MSS}{RTT\sqrt{L}}$

13.1.1.6 TCP Fairness

- if K TCP sessions share same bottleneck link of bandwidth R, each should have average rate of R/K

13.1.1.7 Explicit Congestion Notification

- two bits in IP header (ToS field) marked by network router to indicate congestion
- receiver (seeing congestion indication in IP datagram)) sets ECE bit on receiver-to-sender ACK segment to notify sender of congestion