Lecture 8 (Transport Layer pt 3)

1 TCP

1. Point-to-point
2. Fully duplex
3. Notion of Maximum Segment Size (MSS) exists - maximum message size (doesn't count header)
4. Cumulative ACKs
5. Pipelining exists
6. Connection oriented - handshaking happens
7. Flow control

2 TCP Segment Structure

Header contains the following:

1. Source port number (16 bits)
2. Destination port number (16 bits)
3. Sequence number (32 bits) - stores the byte number instead of segment number
4. Acknowledgement number (32 bits) - next byte number which is needed
5. Header length (atmost 16 bits)
6. C, E (1 bit each) - congestion notification
7. U[rgent] flag (1 bit)
8. Acknowledgement flag (1 bit)
9. PSH [push] flag (1 bit)
10. RST, SYN, FIN (1 bit each)
11. Receive window (16 bits) - number of bytes receiver willing to accept
12. Checksum (16 bits)
13. Urgent data pointer (16 bits)
14. TCP options (variable length)

Without TCP options, header is of 20 bytes

3 TCP ACK

Out of order segment are handled by the implementor and no formal specification is present

4 TCP Timeout

TCP estimates RTT by considering a SampleRTT value from the time taken for ACK to reach, this values varies a lot and hence an estimatedRTT is needed:

$$estimatedRTT = (1 - \alpha)estimatedRTT + \alpha SampleRTT$$

This evaluation gives EWMA (exponential weighted moving average) and α is usually taken as 0.125. Timeout interval is given as:

$$TimeoutInterval = estimatedRTT + 4 \cdot deviationRTT$$

$deviationRTT$ is calculated using EWMA of deviation of $SampleRTT$ from $estimatedRTT$

5 TCP Sender (Simplified)

1. On data from application:
   - Create segment with sequence number
   - Start timer for oldest unACKed packet
2. When timeout:
   - Retransmit segment which caused timeout
   - Restart timeout
3. On ACK:
   - Update list of ACKed packets
   - Restart timer for unACKed packets

6 TCP Receiver

1. Receiving in-order segment
   - delayed ACK
   - wait for 500ms for more segments
2. 2nd segment arrived with another segment's ACK pending
   - send cumulative ACK for both segments
3. Out of order segment
   - send duplicate ACK for the missing segment
4. Segment that shifts the missing gap rightwards
   - send ACK

7 TCP Fast Retransmit

If sender receives 3 additional ACKs for same data, resend unACKed segment with smallest sequence number (since that segment has mostly been lost)

8 TCP Flow Control

`rwnd` field in the header contains the available empty buffer size, unACKed data is limited to `rwnd`

9 TCP Connection Management

9.1 3 Way Handshake

1. Inital sequence number is set randomly (x), SYN bit is set in the header
2. Server (receiver) of the connection sends a packet with a random sequence number (y), SYN and ACK bit set (ACK number = x + 1)
3. On receiving SYNACK, send ACK and request for y+1

9.2 Closing Connection

FIN bit is set to 1 and ACK for this packet is sent