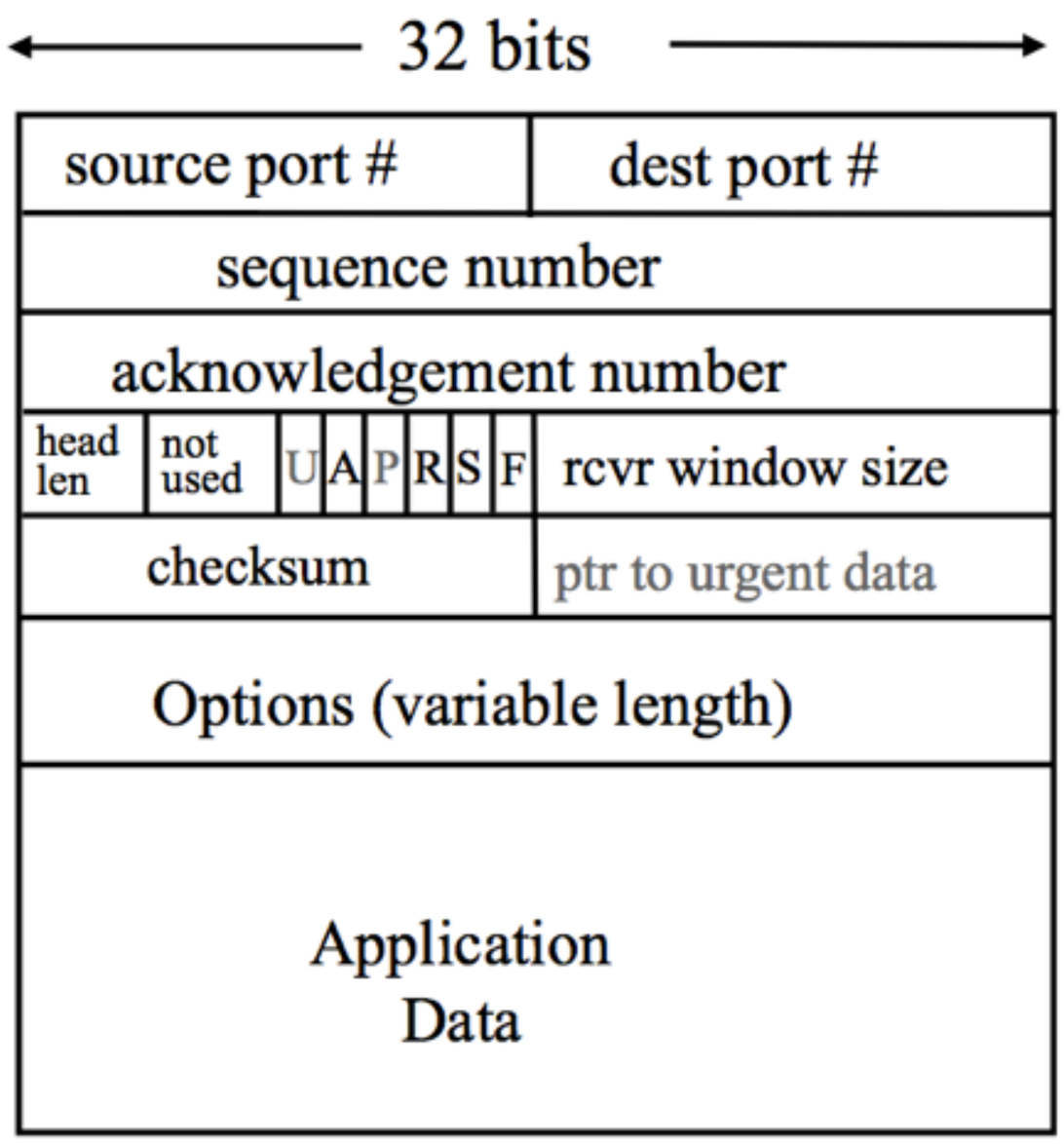
CS118 Discussion Week 4

Taqi

TCP

- Point-to-point, byte-stream reliable transport protocol
  - **Multiplexing/de-multiplexing**: Source/Dest port
  - **Reliable data transfer**: sequence number, ack, checksum, RTT estimation
  - **Connection setup**: sequence number, SYN, receive window
  - **Connection teardown**: sequence number, FIN



TCP Connection Management

- Connection setup: three-way handshaking
  - 1st round: SYN+initial sequence number
  - 2nd round: SYN+SYNACK+server's initial sequence number
  - 3rd round: SYNACK+ACK+(optional)data
- Connection Teardown
  - "Half-closed" connection
  - Why timed wait?

TCP Congestion Control

- End-to-end congestion control
- Congestion is indicated by packet loss
  - Timeout or duplicated loss
  - congestion control is coupled with reliable transfer
- AIMD-based congestion window adaptation
  - Fairness V.S. efficiency
- Slow start for fast convergence
- Fast retransmission/recovery based on duplicated ACK
  - cwnd ← cwnd/2 + 3MSS

> The lost segment starting at SND.UNA MUST be retransmitted and cwnd set to ssthresh plus 3*SMSS. This artificially "inflates" the congestion window by the number of segments (three) that have left the network and which the receiver has buffered.
>
> —RFC 5681

Comparison of Reliable Transport Protocol

Protocol	Buffer at sender	Buffer at receiver	ACK	Timeout/ Retransmission
Stop & Wait	No	No	No out-of-order	Retransmit timeout packet
Go-Back-N	Yes	No	Accumulative Seq#	Retransmit all packets in window
Selective Repeat	Yes	Yes	Received Seq#	Retransmit timeout packet
TCP	Yes	Yes	Next expected Seq#	Retransmit timeout packet

Let’s try it!

Consider the evolution of a TCP connection with the following characteristics. Assume that all the following algorithms are implemented in TCP congestion control: slow start, congestions avoidance, fast retransmit and fast recovery, and retransmission upon timeout. **If ssthresh equals to cwnd, use the slow start algorithm in your calculation.**

- The receiver acknowledges every segment, and the sender always has data available for transmission.
- Initially ssthresh at the sender is set to 6. Assume cwnd and ssthresh are measured in segments, and the transmission time for each segment is negligible. Retransmission timeout (RTO) is initially set to 500ms at the sender and is **unchanged** during the connection lifetime. The RTT is 100ms for all transmissions.
- The connection starts to transmit data at time t = 0, and the initial sequence number starts from 1. **Segment with sequence number 4 is lost once**. No other segments are lost.

How long does it take, in milliseconds, for the sender to receive the ACK for the segment with the sequence number 12? show your intermediate steps or your diagram.