CSC411: Advanced Networks

TCP Sequencing and Flow Control

Note: **This class lecture will be recorded!**
If you do not consent to this recording, please do not ask questions via your video, audio or public chat; send your question to the instructor using the private chat.

Dr. Lisa Frye, Instructor
frye@kutztown.edu
Kutztown University

Sequence Numbers

- Sequence Numbers
  - First byte numbered 0
  - File size 500,000 bytes
  - MSS 1,000 (500 segments)
  - Sequence #1=0, Sequence #2=1000, Sequence #3=2000, etc.
- Maximum Segment Size (MSS)
  - Application-layer data only
- Maximum Transfer Unit (MTU)
  - Application-layer data + transport-layer header + network-layer header

Acknowledgement Numbers

- Sequence number of next segment expected
  - Received bytes 0 through 535
  - Waiting for byte 536
  - Puts 536 in acknowledgement number field of segment
- Buffer out-of-order segments

TCP ACK Generation

Event	TCP Receiver Action
Arrival of in-order segment with expected sequence number. All data up to expected sequence number already acknowledged.	Delayed ACK. Wait up to 500 msec for arrival of another in-order segment. If next in-order segment does not arrive in this interval, send an ACK.
Arrival of in-order segment with expected sequence number. One other in-order segment waiting for ACK transmission.	Immediately send single cumulative ACK, ACKing both in-order segments.
Arrival of out-of-order segment with higher-than-expected sequence number. Gap detected.	Immediately send duplicate ACK, indicating sequence number of next expected byte (which is the lower end of the gap).
Arrival of segment that partially or completely fills in gap in received data.	Immediately send ACK, provided that segment starts at the lower end of gap.

Table 3.2 ♦ TCP ACK Generation Recommendation [RFC 1122, RFC 2581]

Lost Acknowledgement

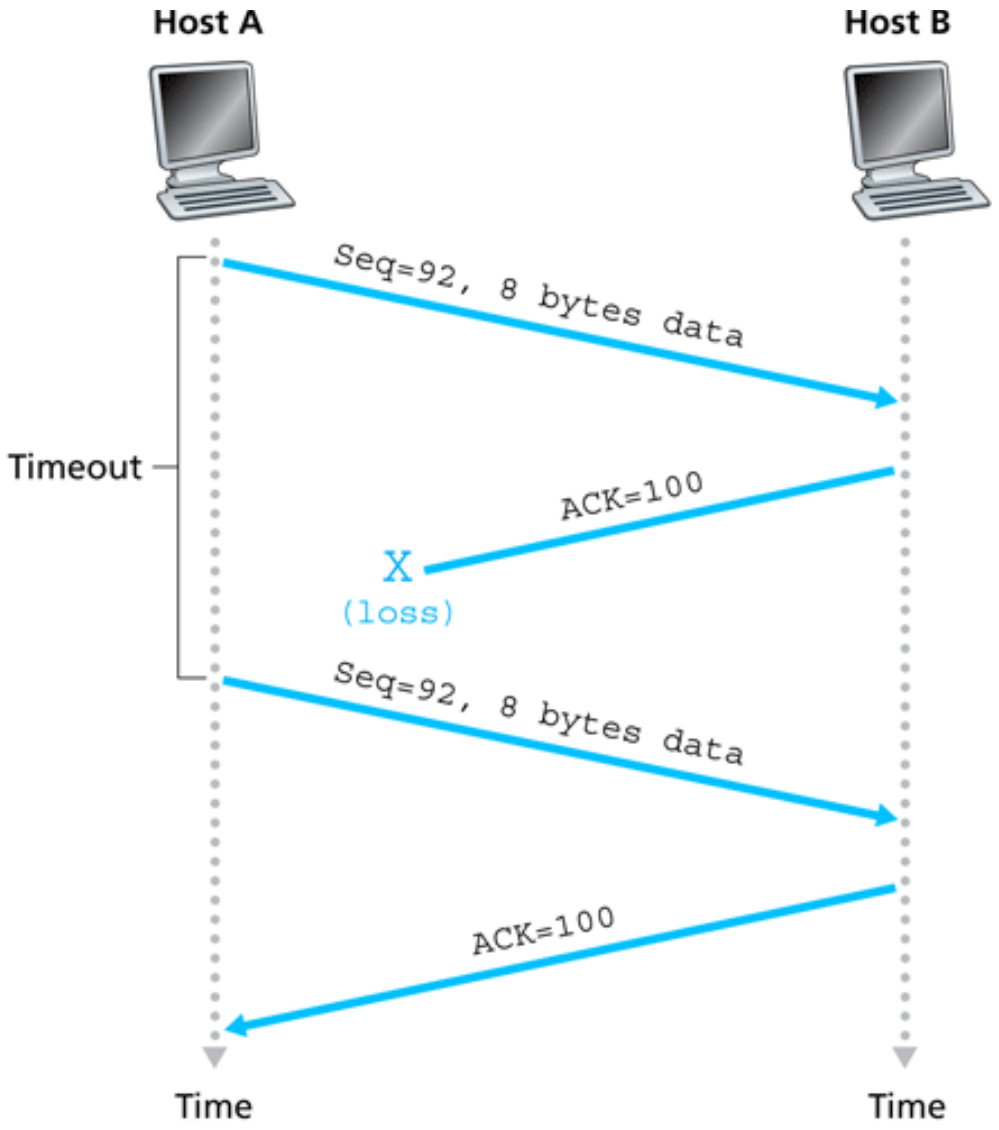


Figure 3.34 ♦ Retransmission due to a lost acknowledgment

Send Two Segments, Back to Back

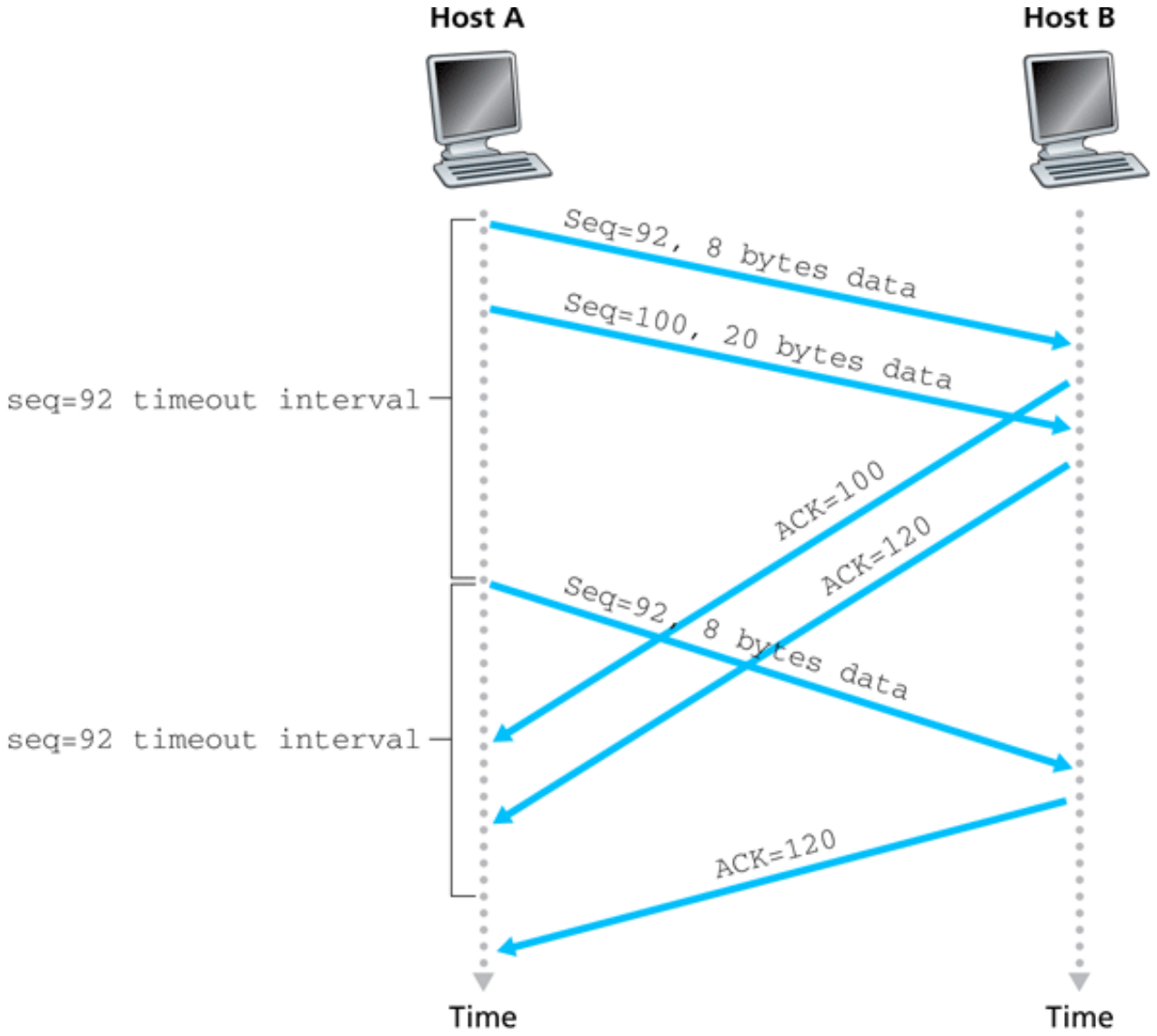


Figure 3.35 ♦ Segment 100 not retransmitted

Receive 2nd Acknowledgement

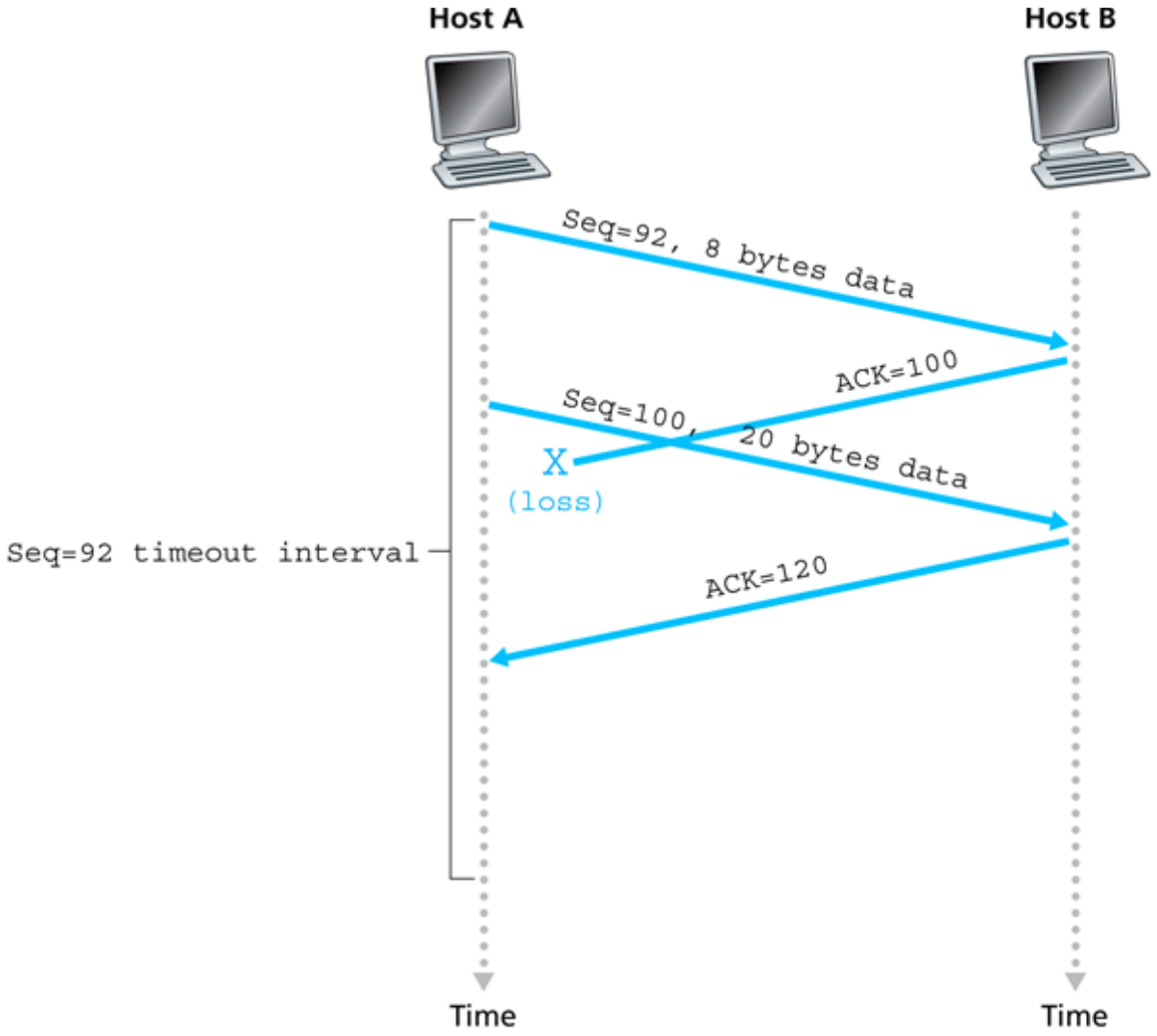


Figure 3.36 ♦ A cumulative acknowledgment avoids retransmission of the first segment.

Simulation – Try These!

- http://www.cs.stir.ac.uk/~kjt/software/comms/jasper/TCPcs.html
- http://www.cs.stir.ac.uk/~kjt/software/comms/jasper/SWP3.html

Question

- Suppose Host A sends two TCP segments back to back to Host B over a TCP connection. The first segment has sequence number 90; the second has sequence number 110. How much data is in the first segment?

Question

- Suppose Host A sends two TCP segments back to back to Host B over a TCP connection. The first segment has sequence number 90; the second has sequence number 110. Suppose that the first segment is lost but the second segment arrives at B. In the ACK sent from B to A, what will be the acknowledgement number?

Questions – Information

- Host A and Host B are communicating over a TCP connection, and Host B has already received from A all bytes up through byte 144. Suppose that Host A then sends two segments to Host B back-to-back. The first and second segments contain 20 and 40 bytes of data, respectively. In the first segment, the sequence number is 145, source port number is 303, and the destination port number is 80. Host B sends an acknowledgement whenever it receives a segment from host A.

Question #1

- In the second segment sent from Host A to B, what are the sequence number, source port number and destination port number?

Question #2

- If the first segment arrives before the second segment, in the acknowledgement field of the first arriving segment, what is the acknowledgement number, the source port number, and the destination port number?

Question #3

- If the second segment arrives before the first segment, in the acknowledgement field of the first arriving segment, what is the acknowledgement number?

Question #4

- Assume now the normal operation of TCP when handling acknowledgements. If the first segment arrives before the second segment, in the acknowledgement field of the first arriving segment, what is the acknowledgement number, the source port number, and the destination port number?

Flow Control

- Buffer overflow at receiver
- Receive window

Receiver Variables

- RcvBuffer
- LastByteRead
- LastByteRcvd
- RcvWindow

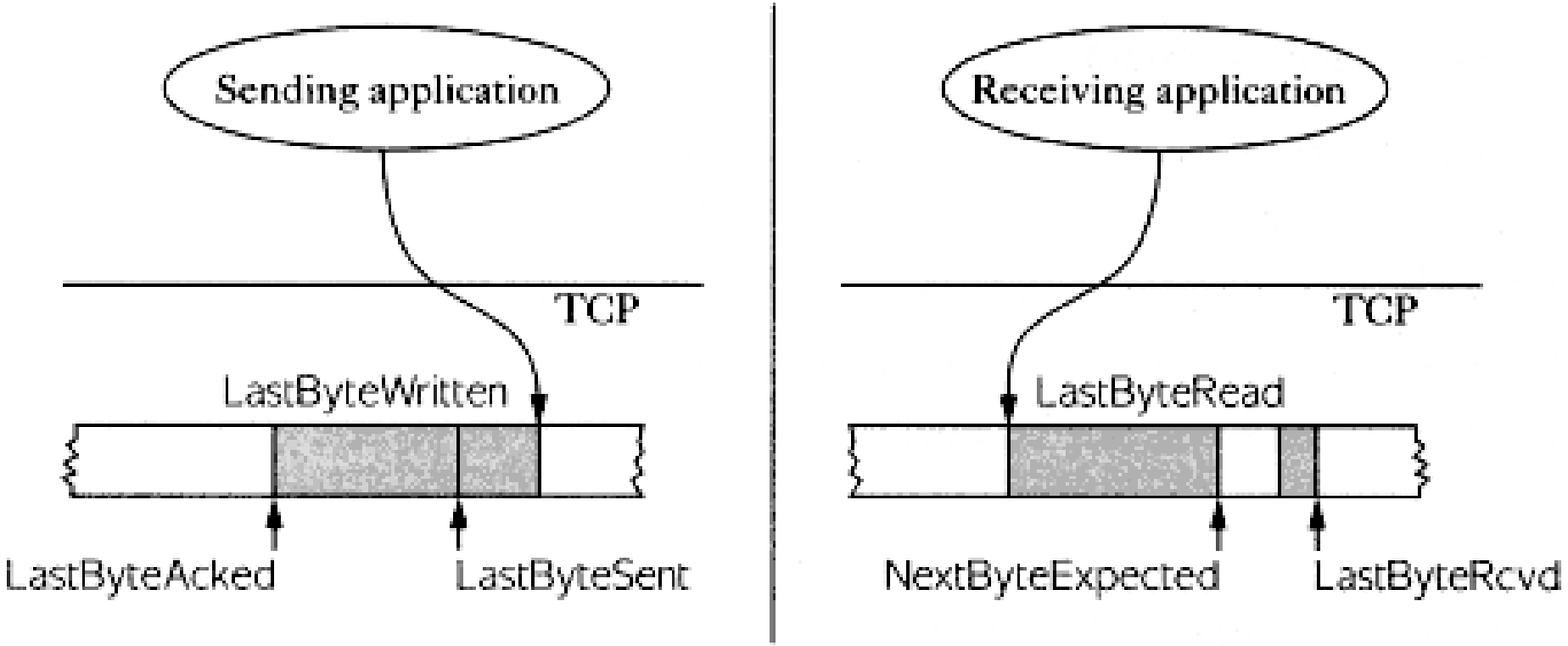
- LastByteRcvd – LastByteRead <= RcvBuffer
- RcvWindow = RcvBuffer – (LastByteRcvd – LastByteRead)

Sender Variables

- LastByteSent
- LastByteAcked

- LastByteSent – LastByteAcked = amount of unacknowledged data sender sent to the receiver.

Sliding Window



From http://www.ssfnet.org/Exchange/tcp/tcpTutorialNotes.html

Demo – Try This!

- http://www2.rad.com/networks/2004/sliding_window/

Silly Window Syndrome

- What if RcvWindow = 0?
- Receiver reads one byte from buffer
  - Receiver sends ACK
  - Sender sends 1-byte segments
- What is wrong with this?
- No data from receiver to sender
  - Sender continues to send 1-byte segments

Silly Window Syndrome Avoidance

- Sender – shortly
- Receiver
  - Keep track of current window size
  - Only increase to sender if “significant”

- What is wrong with this?

Solutions

- Send ACK, don't increase window size

- Delay the ACK
  - Advantage – decrease traffic
    - How is the traffic decreased?
  - Disadvantages
    - Delayed too long – what happens?
    - Confuse averageRTT

Sender Silly Window Syndrome Avoidance – Nagle's Algorithm

- Clumping
  - TCP waits to create segment
- How long should the wait be?
- Self-clocking
  - No computation
  - Arrival of ACK triggers transmission