TCP: Reliable Data Transfer

Course “Computer Networks”

Master of Science in “Computer Science”
AUEB

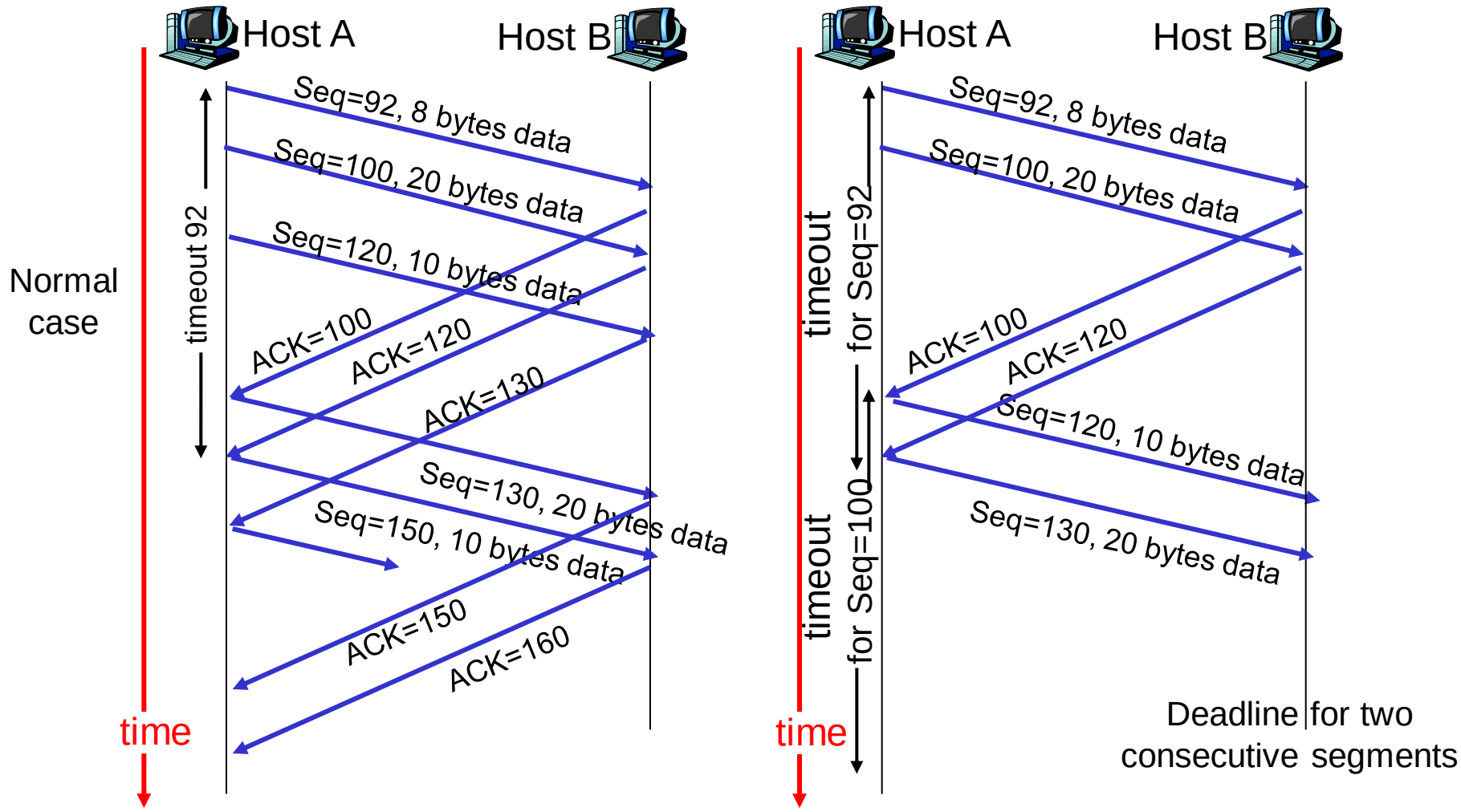
George D. Stamoulis

Fall semester 2016-17

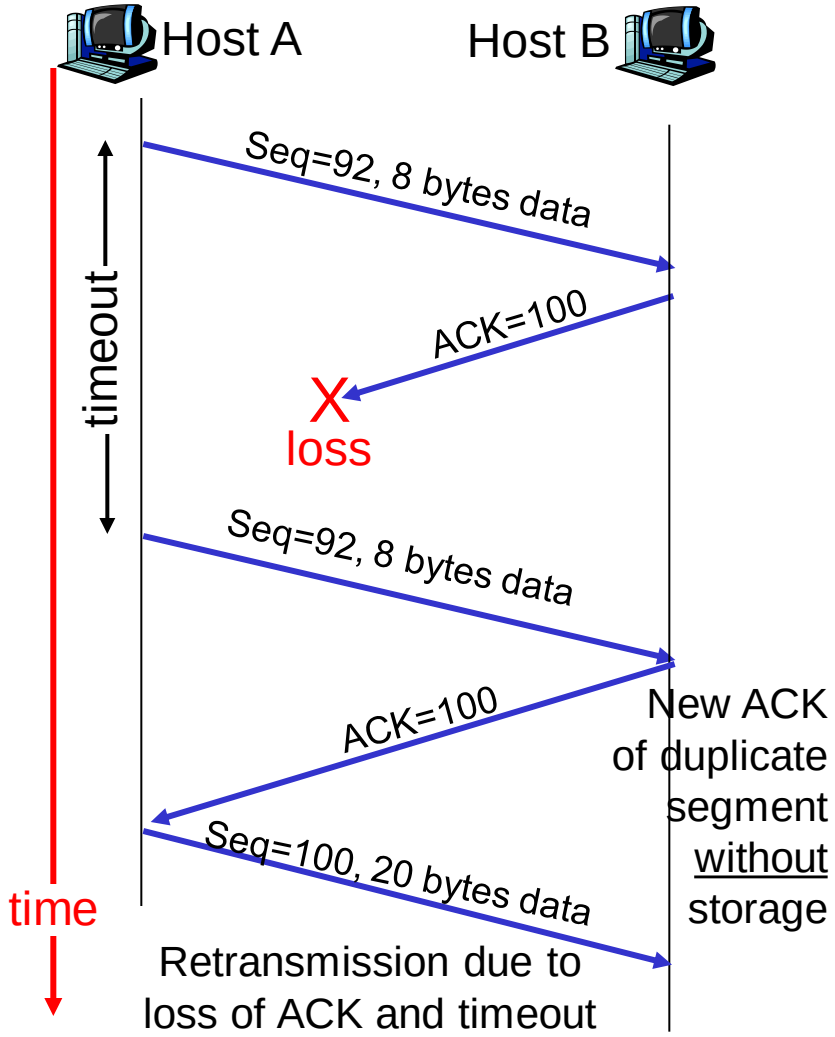
Sliding window protocol employed in TCP

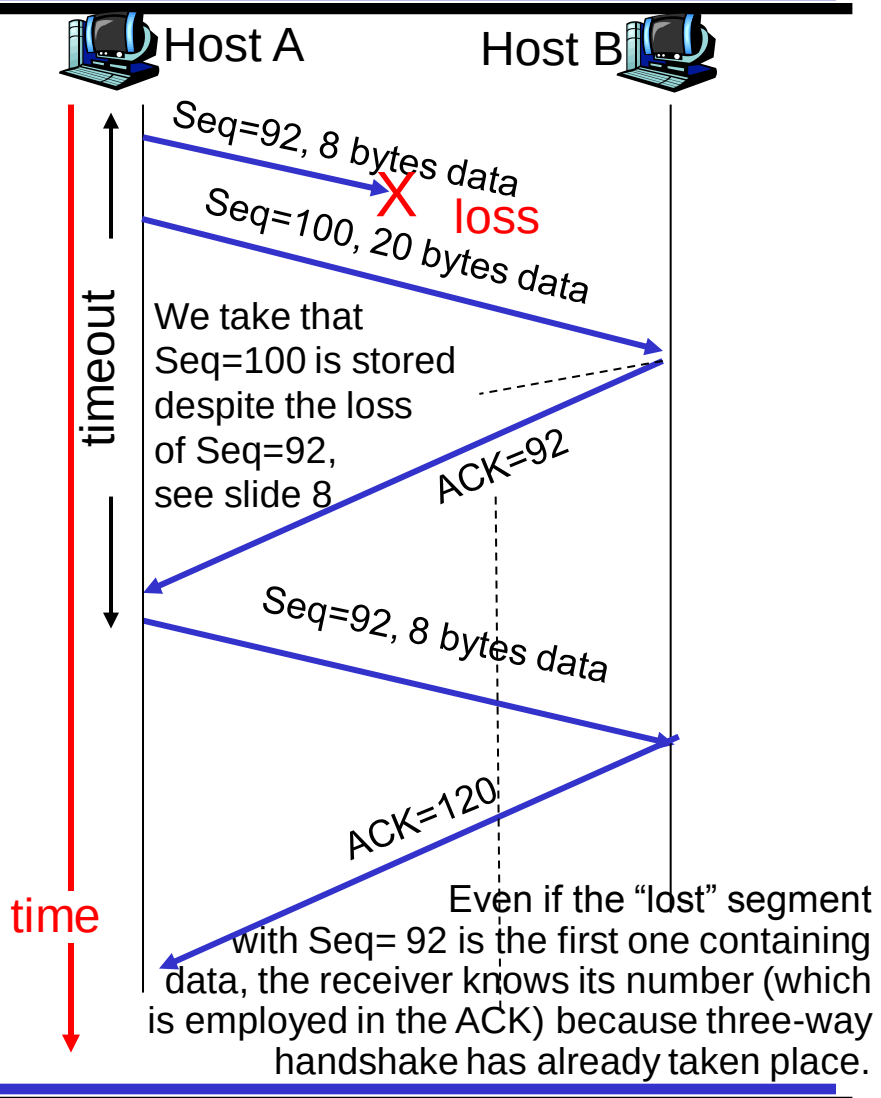
- Segments are numbered according to data bytes, rather than as 1,2,3,…
  - Example: 1st segment: SEQ = x - contains 512 data bytes
    2nd segment: SEQ = x+512 - contains 1024 data bytes, 3rd segment: SEQ = x+1536
- Segments are ACKed cumulatively
  - ACK number = sequence number of the next data byte awaited
    - Implies that all previous bytes have been received
- Only one timer is employed for retransmissions
  - Upon timeout only the oldest not ACKed segment is retransmitted
    - The timeout parameter is then doubled
- Is TCP "Go Back N" or "Selective Repeat" ? TCP protocol for handling retransmissions can correspond to :
  - Variation of Go Back N, if out-of-order segments are not stored when they are received:
    - Only one (in general) retransmission upon timeout,
  - Selective Repeat, if out-of-order segments are not stored
  - or to an "intermediate" protocol
  - The exact protocol depends on the buffering (storage) policy concerning out-of-order segments, without the need for an "agreement" of the two sides.

Examples (I)

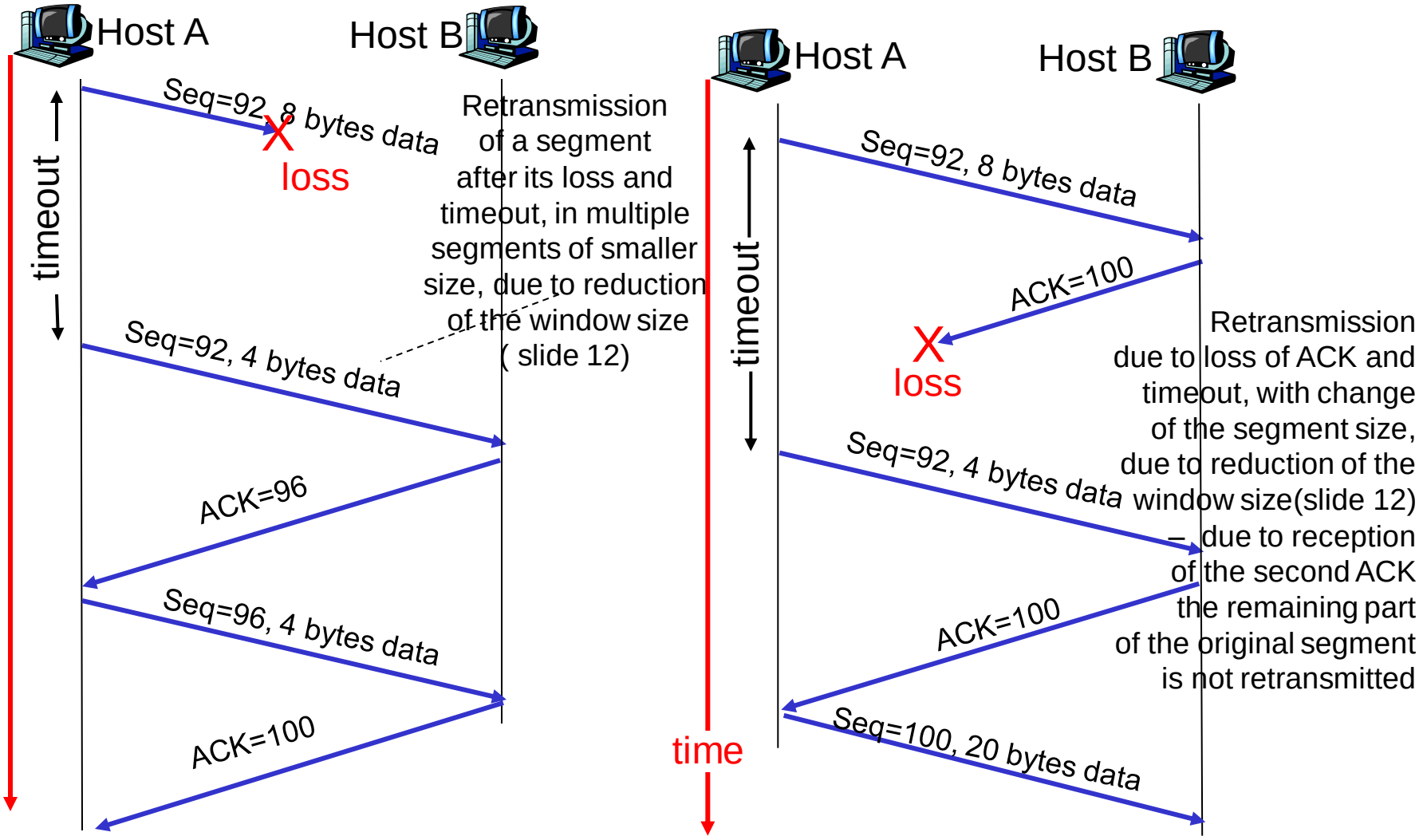

Host A
Host B
Normal case
timeout 92
Seq=92, 8 bytes data
Seq=100, 20 bytes data
Seq=120, 10 bytes data
ACK=100
ACK=120
ACK=130
Seq=130, 20 bytes data
Seq=150, 10 bytes data
ACK=150
ACK=160
time
Host A
Host B
timeout for Seq=92
timeout for Seq=100
Seq=92, 8 bytes data
Seq=100, 20 bytes data
ACK=100
ACK=120
Seq=120, 10 bytes data
Seq=130, 20 bytes data
time
Deadline for two consecutive segments

Examples (II)

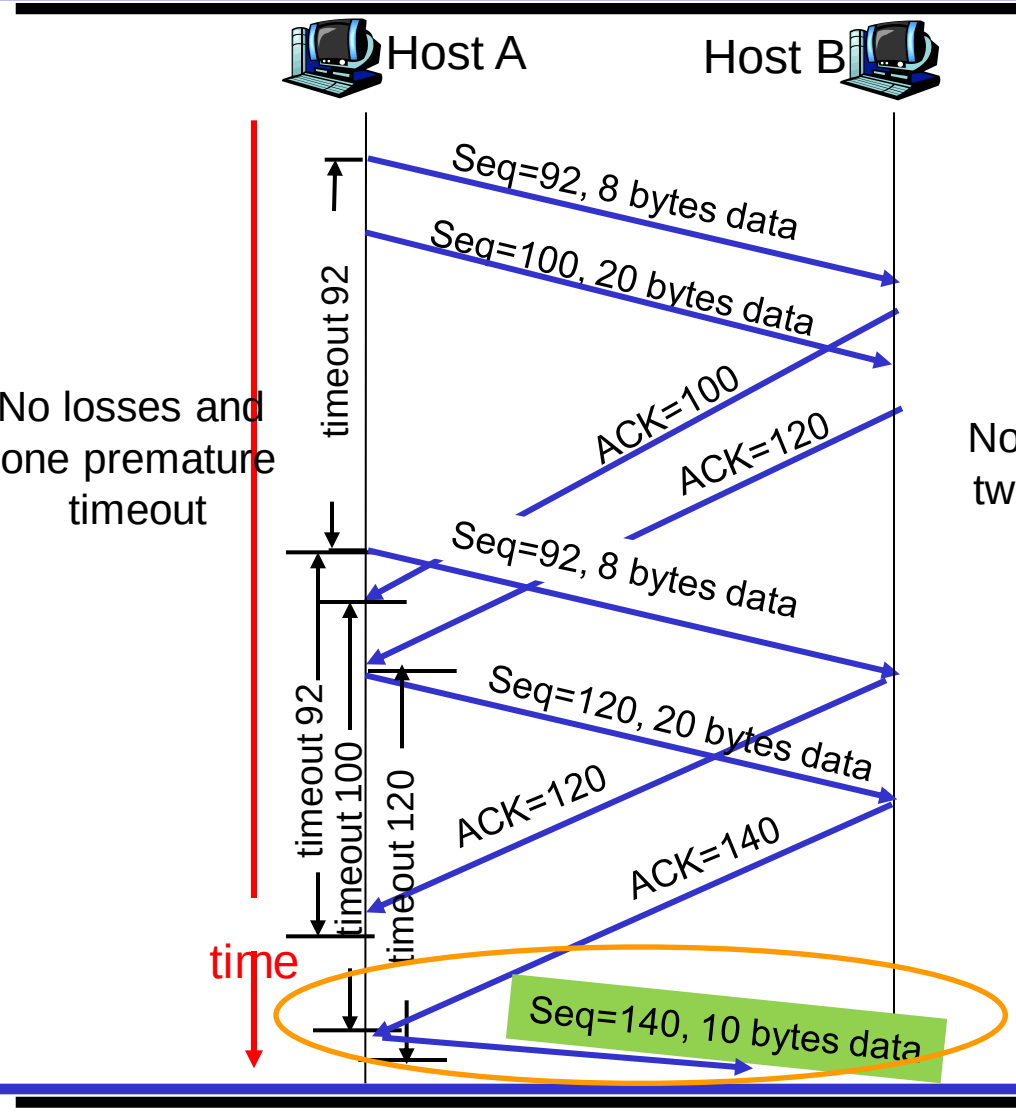

Host A
Host B
timeout
Seq=92, 8 bytes data
ACK=100
X
loss
Seq=92, 8 bytes data
ACK=100
New ACK
of duplicate
segment
without
storage
Seq=100, 20 bytes data
time
Retransmission due to
loss of ACK and timeout

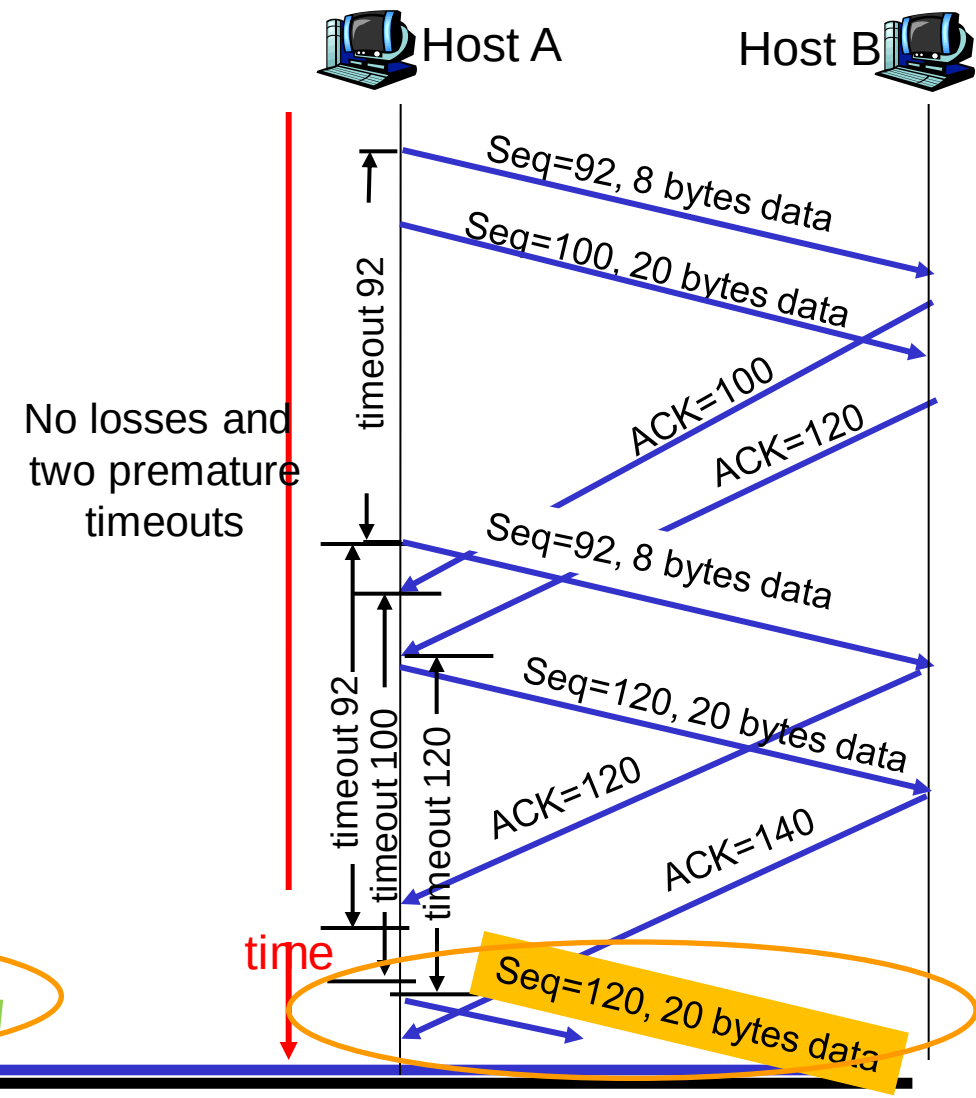

Host A
Host B
timeout
Seq=92, 8 bytes data
X
loss
Seq=100, 20 bytes data
We take that
Seq=100 is stored
despite the loss
of Seq=92,
see slide 8
ACK=92
Seq=92, 8 bytes data
ACK=120
time
Even if the “lost” segment
with Seq= 92 is the first one containing
data, the receiver knows its number (which
is employed in the ACK) because three-way
handshake has already taken place.

Examples(III)

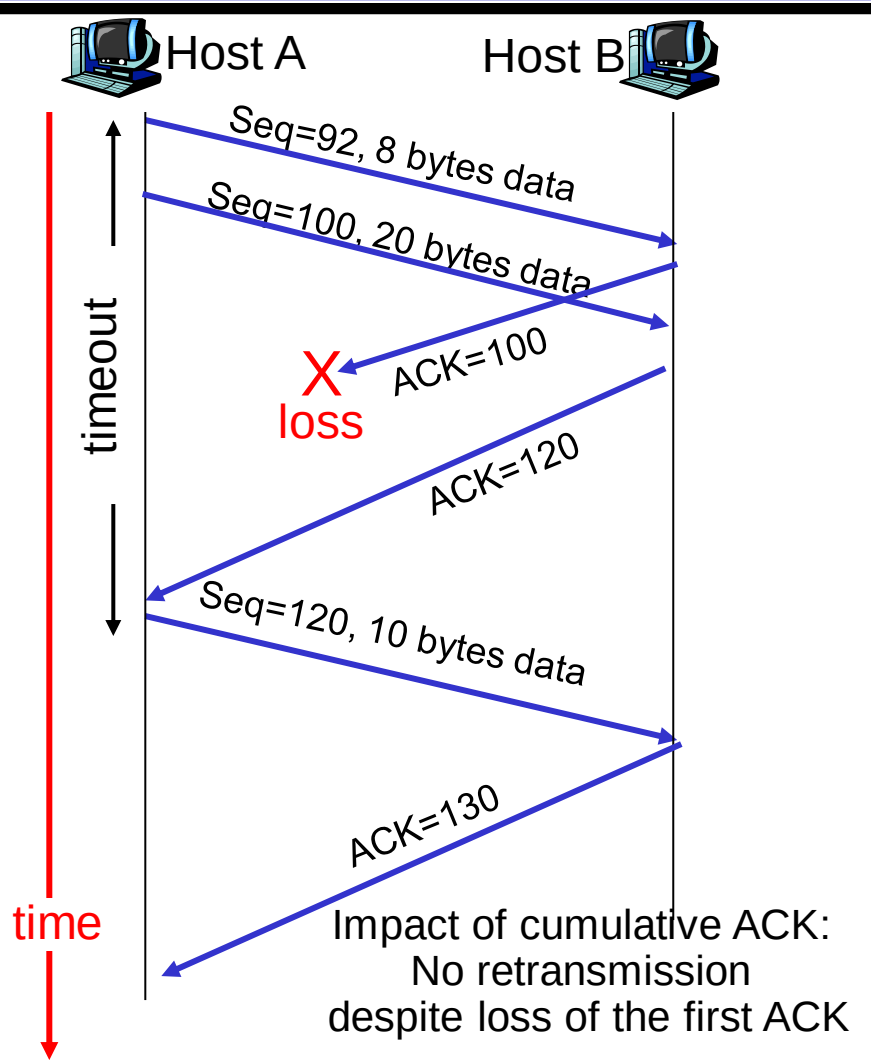


Examples(IV)

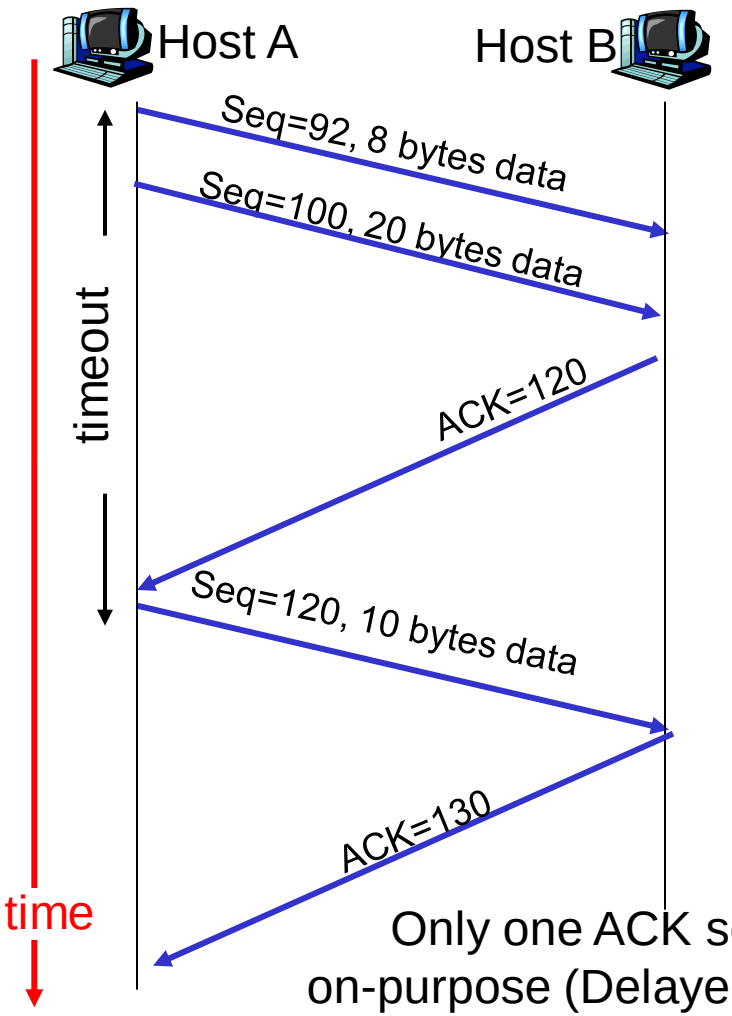

Host A
Host B
No losses and one premature timeout
Seq=92, 8 bytes data
Seq=100, 20 bytes data
timeout 92
ACK=100
ACK=120
Seq=92, 8 bytes data
timeout 92
timeout 100
Seq=120, 20 bytes data
timeout 120
ACK=120
ACK=140
time
Seq=140, 10 bytes data


Host A
Host B
No losses and two premature timeouts
Seq=92, 8 bytes data
Seq=100, 20 bytes data
timeout 92
ACK=100
ACK=120
Seq=92, 8 bytes data
timeout 92
timeout 100
Seq=120, 20 bytes data
timeout 120
ACK=120
ACK=140
time
Seq=120, 20 bytes data

Examples(V)

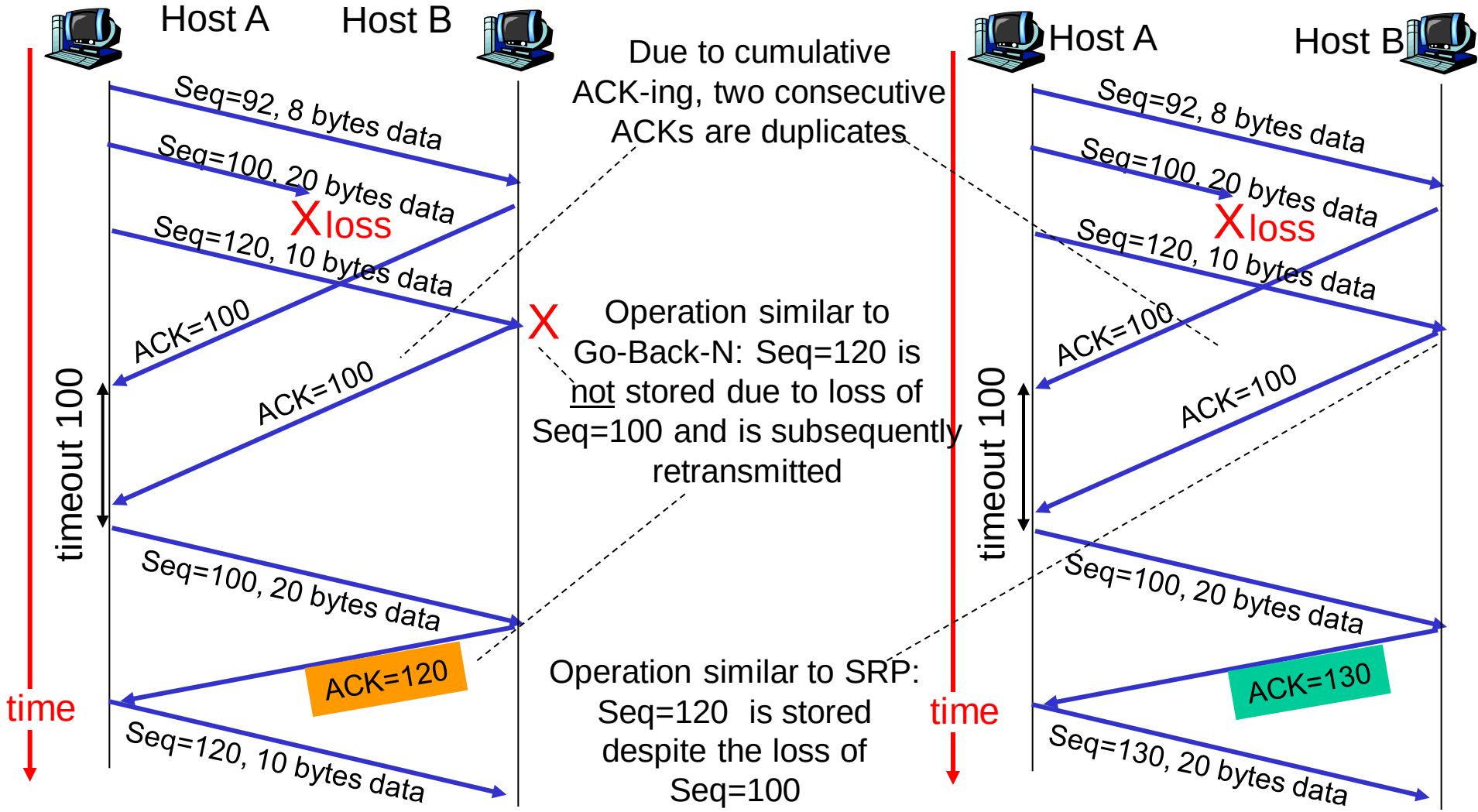


Impact of cumulative ACK:
No retransmission
despite loss of the first ACK

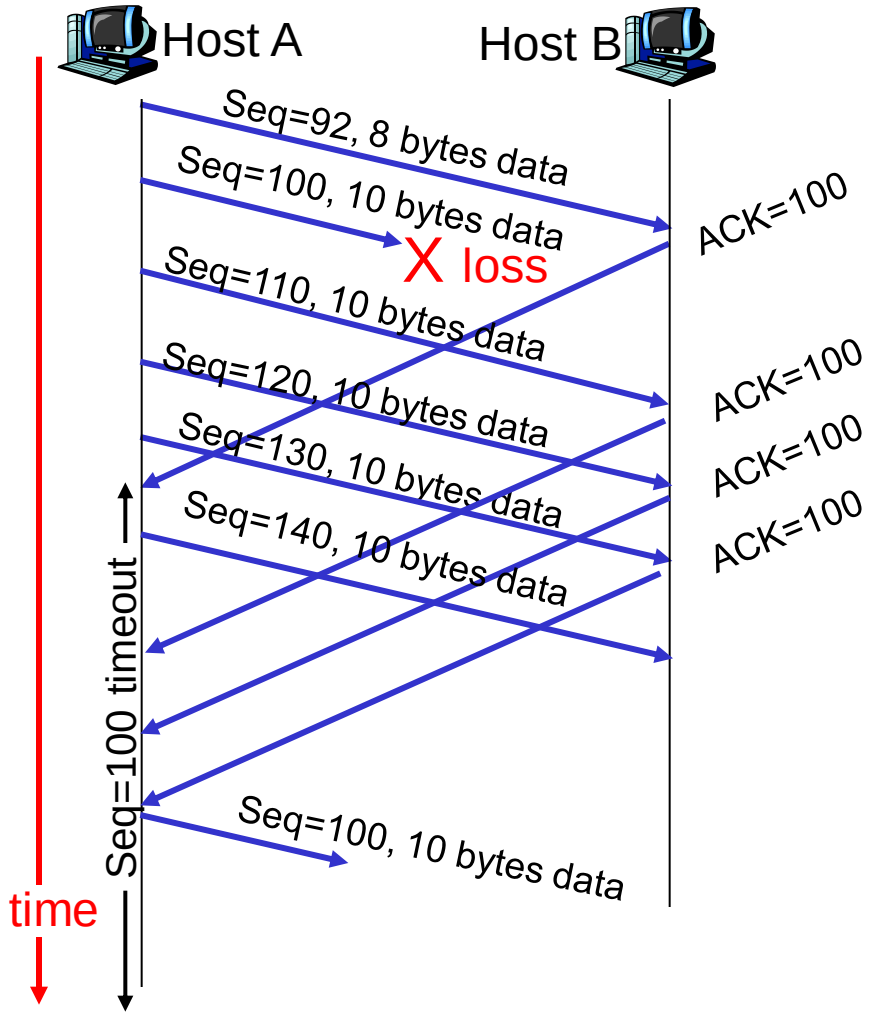


Only one ACK sent
on-purpose (Delayed ACK)

Examples(VI)


Host A
Host B
Seq=92, 8 bytes data
Seq=100, 20 bytes data
X loss
Seq=120, 10 bytes data
ACK=100
ACK=100
timeout 100
Seq=100, 20 bytes data
ACK=120
Seq=120, 10 bytes data
time
Due to cumulative ACK-ing, two consecutive ACKs are duplicates
X
Operation similar to Go-Back-N: Seq=120 is not stored due to loss of Seq=100 and is subsequently retransmitted
Operation similar to SRP: Seq=120 is stored despite the loss of Seq=100
Host A
Host B
Seq=92, 8 bytes data
Seq=100, 20 bytes data
X loss
Seq=120, 10 bytes data
ACK=100
ACK=100
timeout 100
Seq=100, 20 bytes data
ACK=130
Seq=130, 20 bytes data
time

Examples(VII)



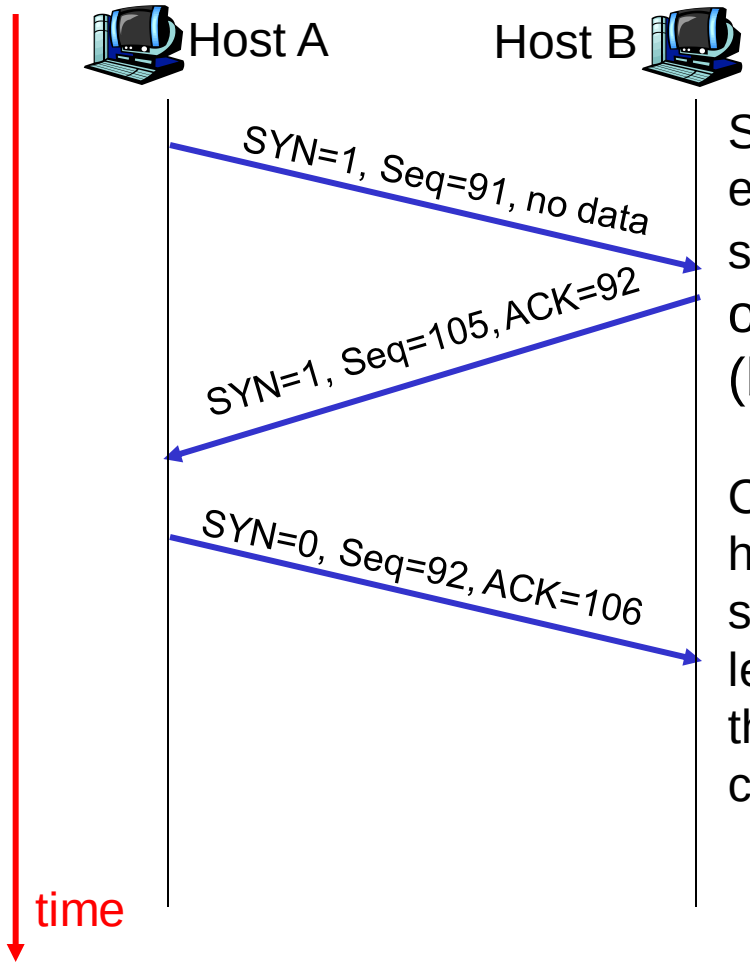
Fast retransmit due to reception of y=3 consecutive duplicates of the initial ACK=100

Since timeout for Seq=100 occurs later, the transmission of several-out-of-order segments is avoided due to this earlier reaction

Connection Establishment

- Main difference between layers 2 (data link) and 4 (transport):
  - 2nd layer: all frames belong to a single data link, which runs “for ever”; the sliding window protocol employs a fixed and sufficiently large value of window
    → When a frame is received that has the same number with another previously received (=old) frame, then it can be deduced with certainty (from the number of the frame) whether this is a new frame or a redundant copy of the old one.
  - 4th layer: the above distinction cannot be made with certainty when a segment is received that has the same sequence number (and the same source and destination port numbers) with a previously received segment
  - ... because this segment can also be a very delayed copy of a segment from an old connection between the same ports
    → In the 4th layer: numbering of segments should also provide distinction of segments across different connections (between the same ports)

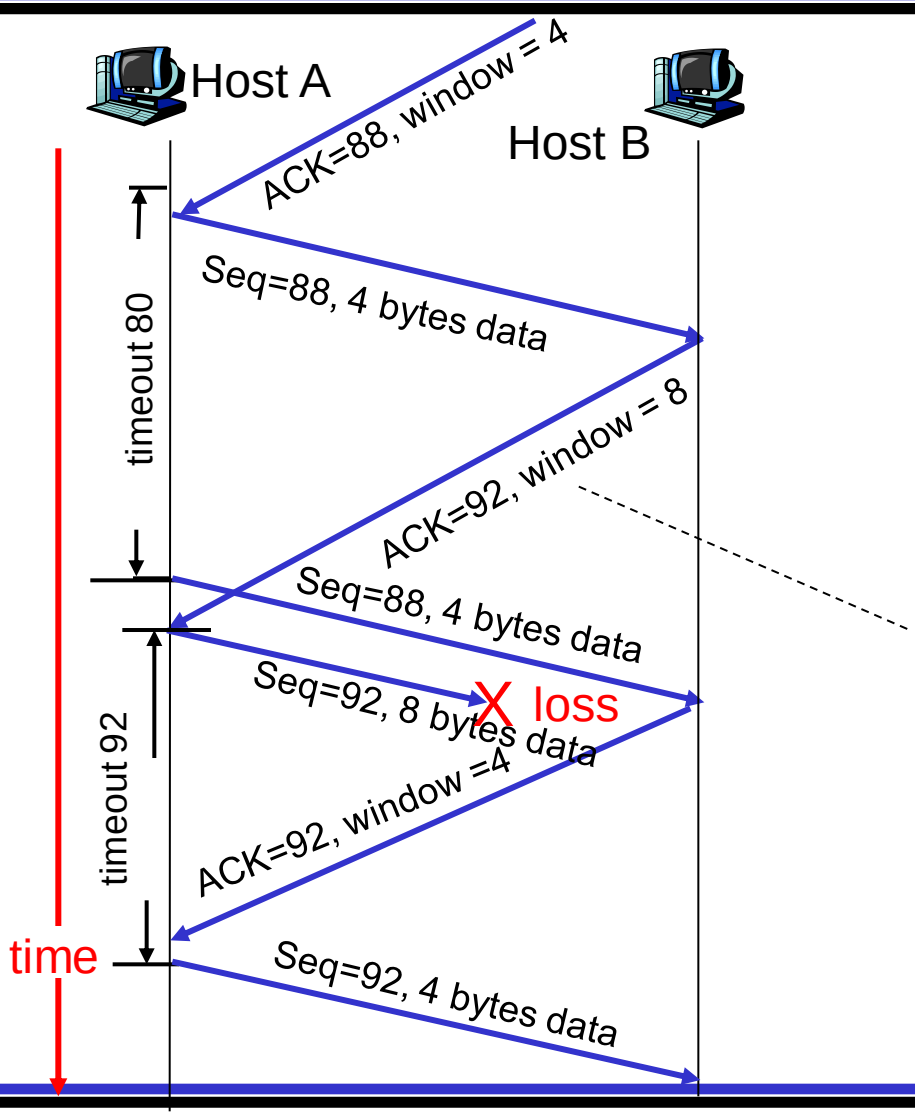
Three-way Handshake



Selection of a random initial sequence number in each end using an adequate number of the least significant bits of the clock, provide distinction of segments across different connections (between the same ports)

Counter-example: If all SYN segments of a client had Seq=0, then the retransmission of a SYN segments (e.g. due to loss of its ACK), could lead to confusion of the server regarding whether this is a new or a retransmitted request for a connection.

Window Reduction



In all examples of slides 3 – 9 it is implicitly assumed that the window is not increasing upon reception of ACKs (according to the congestion control algorithm), because it is limited by low values of the flow control window.

Also, the reduction of the window size assumed in slide 5 can happen as depicted in this figure