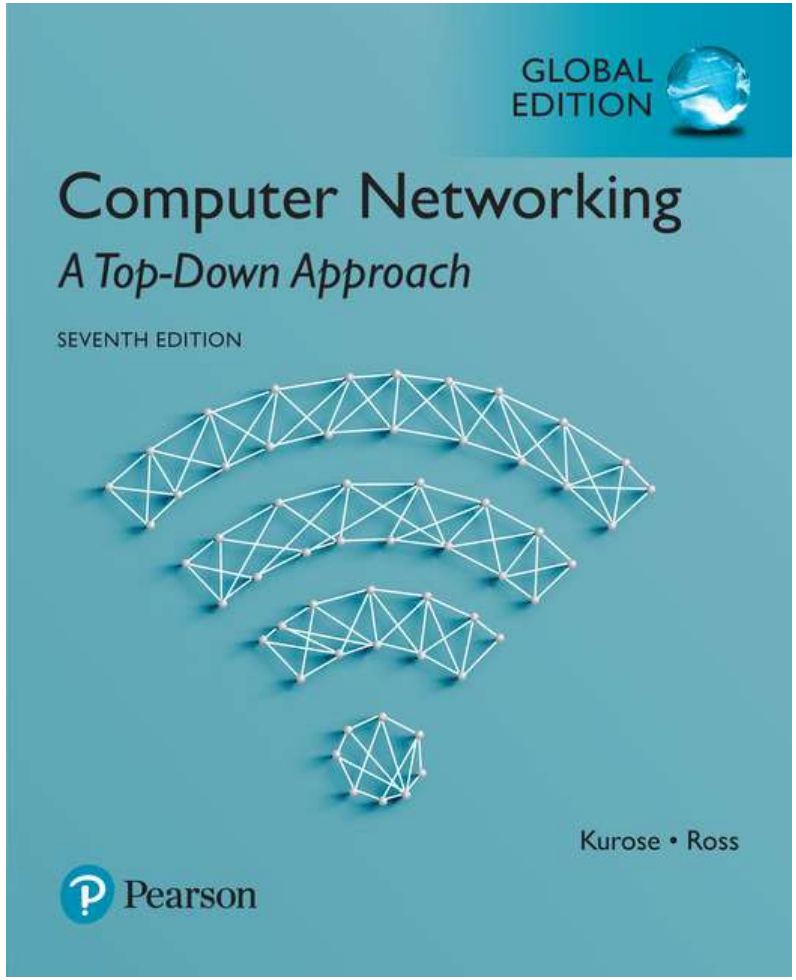
제15강
TCP 신뢰 전송과 흐름 제어

Computer Networking: A Top Down Approach

컴퓨터 네트워크
(2019년 1학기)

박승철교수

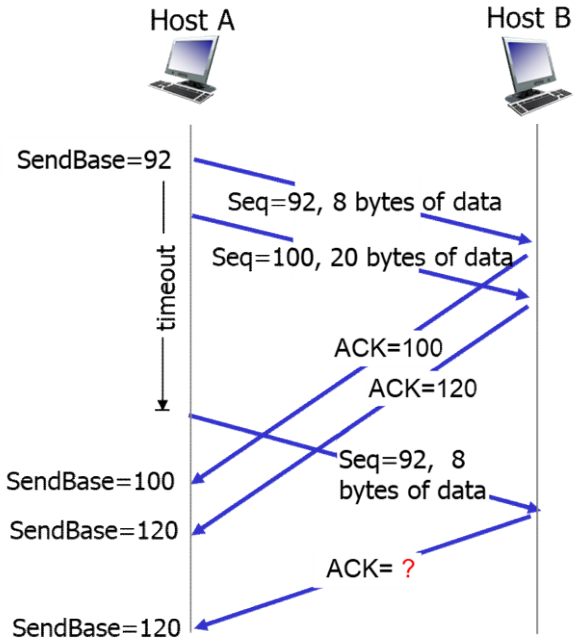
한국기술교육대학교
컴퓨터공학부



Pre-study Test :

1) TCP의 신뢰 전송을 잘못 설명한 것은?

① 바이트 스트림 전송을 지원한다.
② 누적 수신확인(cumulative ack)을 지원한다.
③ 가장 오래된 세그먼트에 대한 1개의 타이머를 유지한다.
④ 수신자는 버퍼를 유지하지 않는다.



2) 오른쪽 그림의 마지막 ACK 번호는 무엇인가?

① 93
② 100
③ 101
④ 120

3) TCP는 누적 수신확인을 지원한다. 동일한 확인번호를 가진 ACK이 중복되어 도착하는 경우는 어떤 경우인가?

① 라우터 혼잡 상황이 심각하여 지속적으로 패킷 손실 발생
② 라우터 혼잡 상황이 경미하여 순간적으로 패킷 손실 발생
③ 라우터 혼잡 상황이 심각하여 순간적으로 패킷 손실 발생
④ 라우터 혼잡 상황이 경미하여 지속적으로 패킷 손실 발생

4) 아직 타임아웃이 발생하지 않았지만 동일한 ACK가 중복되어 여러 번 수신되었다. 어떻게 하는 것이 바람직한가?

① 타임아웃이 될 때까지 기다린다.
② 패킷이 손실되었음으로 마지막으로 전송한 세그먼트를 재전송한다.
③ 패킷이 손실되었음으로 처음 전송한 세그먼트를 재전송한다.
④ 패킷이 손실되었음으로 ACK 수신번호에 해당하는 세그먼트를 재전송한다.

5) 버퍼의 크기가 B이고 응용 프로세스가 마지막으로 읽은 바이트 번호가 R이다. TCP에 의해 수신된 마지막 바이트 번호가 A일 때 수신윈도우(rwnd)의 크기는 얼마인가?

① B
② B - R
③ B – R + A
④ B – R + A + 1

6) 수신윈도우의 크기가 rwnd이고, 최종 송신 바이트 번호는 S이며, 최종 수신확인(ack) 번호는 A이다. rwnd, S, A간에 유지되어야 하는 관계는?

① S – A ≤ rwnd
② S – A ≥ rwnd
③ S – A + 1 ≤ rwnd
④ S – A + 1≥ rwnd

Chapter 3 outline

3.1 transport-layer services

3.2 multiplexing and demultiplexing

3.3 connectionless transport: UDP

3.4 principles of reliable data transfer

3.5 connection-oriented transport: TCP

- segment structure
- reliable data transfer
- flow control
- connection management

3.6 principles of congestion control

3.7 TCP congestion control

TCP reliable data transfer

- TCP creates rdt service on top of IP's unreliable service
  - pipelined segments
  - cumulative acks
  - single retransmission timer
- retransmissions triggered by:
  - timeout events
  - duplicate acks

let's initially consider simplified TCP sender:

- ignore duplicate acks
- ignore flow control, congestion control

TCP sender events:

data rcvd from app:

- create segment with seq #
- seq # is byte-stream number of first data byte in segment
- start timer if not already running
  - think of timer as for oldest unacked segment
  - expiration interval: `TimeOutInterval`

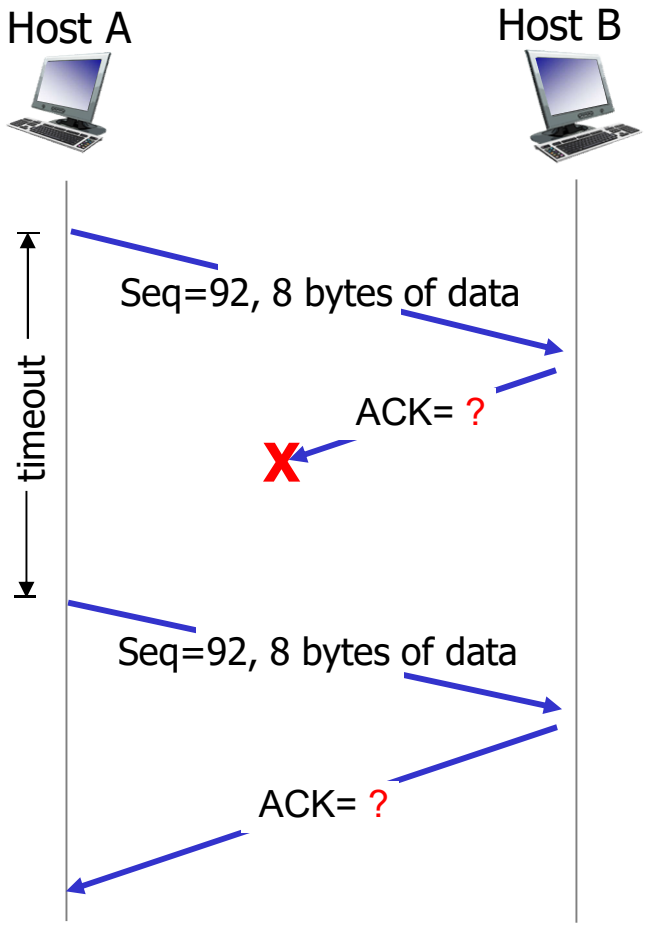
timeout:

- retransmit segment that caused timeout
- restart timer(타이머 대상 패킷 이동)

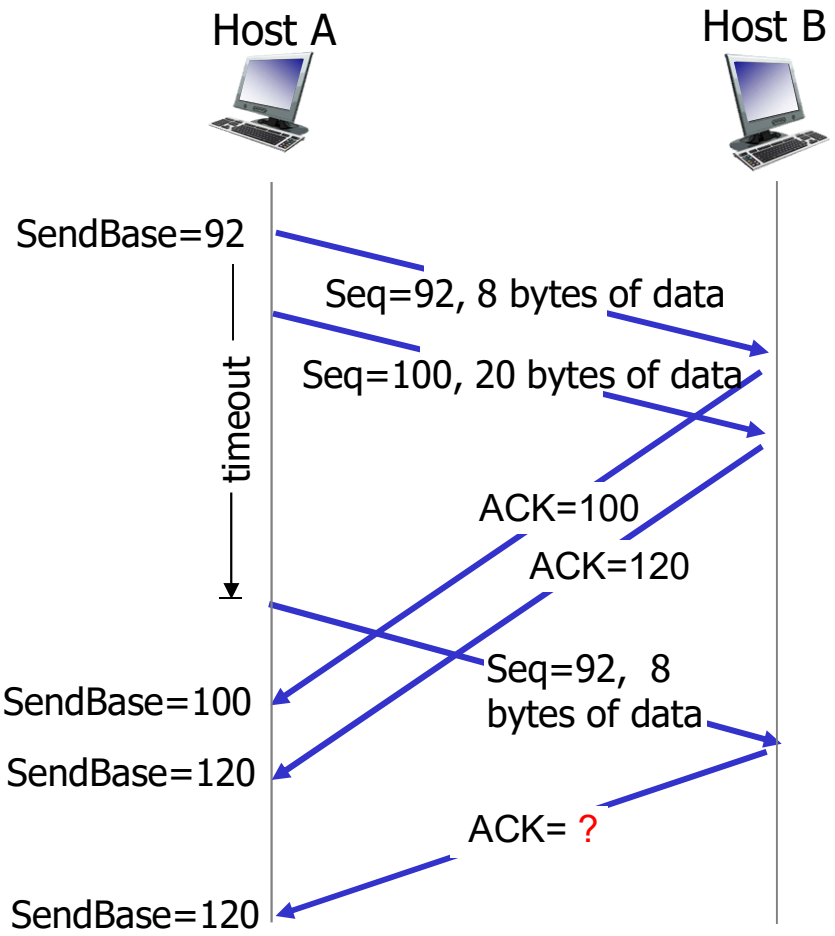
ack rcvd:

- if ack acknowledges previously unacked segments
  - update what is known to be ACKed
  - start timer if there are still unacked segments

TCP: retransmission scenarios

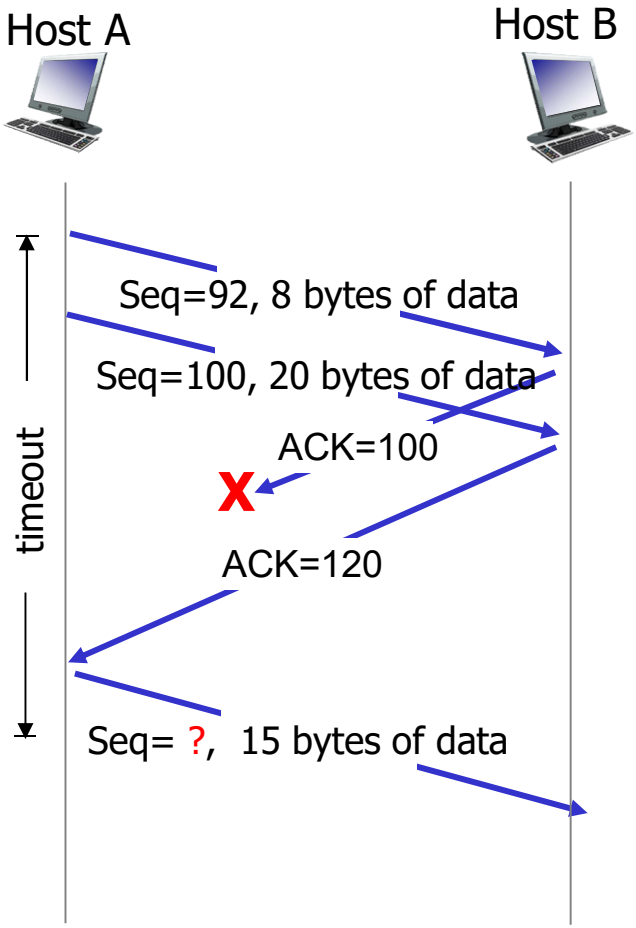


lost ACK scenario



premature timeout

TCP: retransmission scenarios



cumulative ACK

TCP fast retransmit

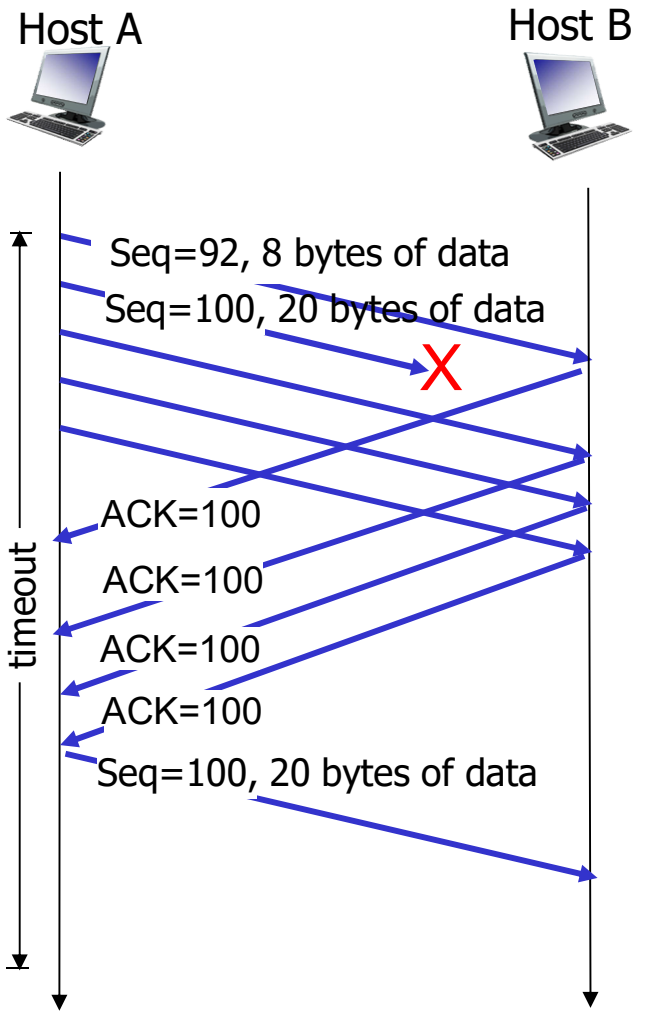
- time-out period often relatively long:
  - long delay before resending lost packet
- detect lost segments via duplicate ACKs.
  - sender often sends many segments back-to-back
  - if segment is lost, there will likely be many duplicate ACKs.

TCP fast retransmit

if sender receives 3 ACKs for same data ("triple duplicate ACKs"), resend unacked segment with smallest seq #

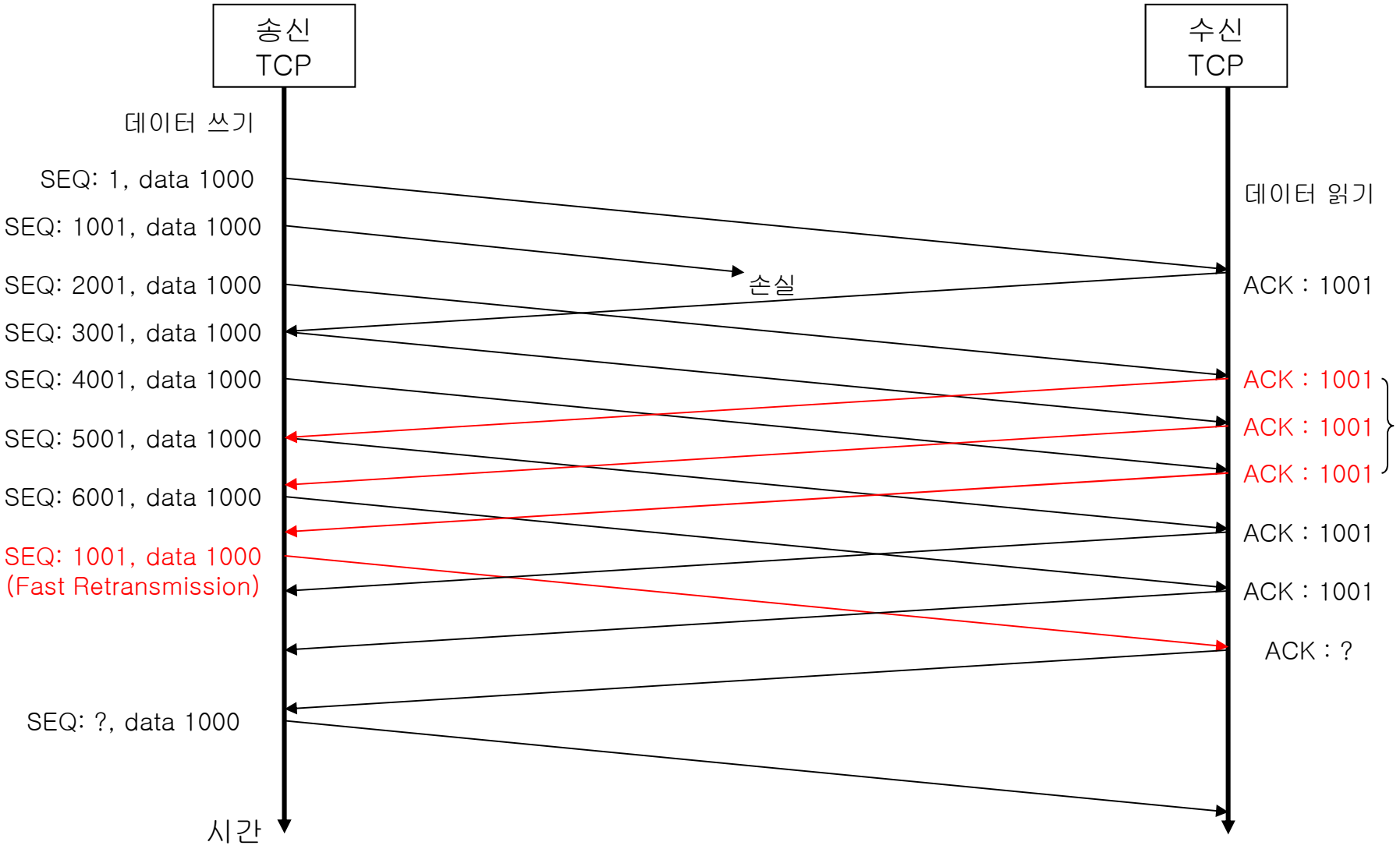
- likely that unacked segment lost, so don't wait for timeout

TCP fast retransmit



fast retransmit after sender receipt of triple duplicate ACK

TCP fast retransmit



Chapter 3 outline

3.1 transport-layer services

3.2 multiplexing and demultiplexing

3.3 connectionless transport: UDP

3.4 principles of reliable data transfer

3.5 connection-oriented transport: TCP

- segment structure
- reliable data transfer
- flow control
- connection management

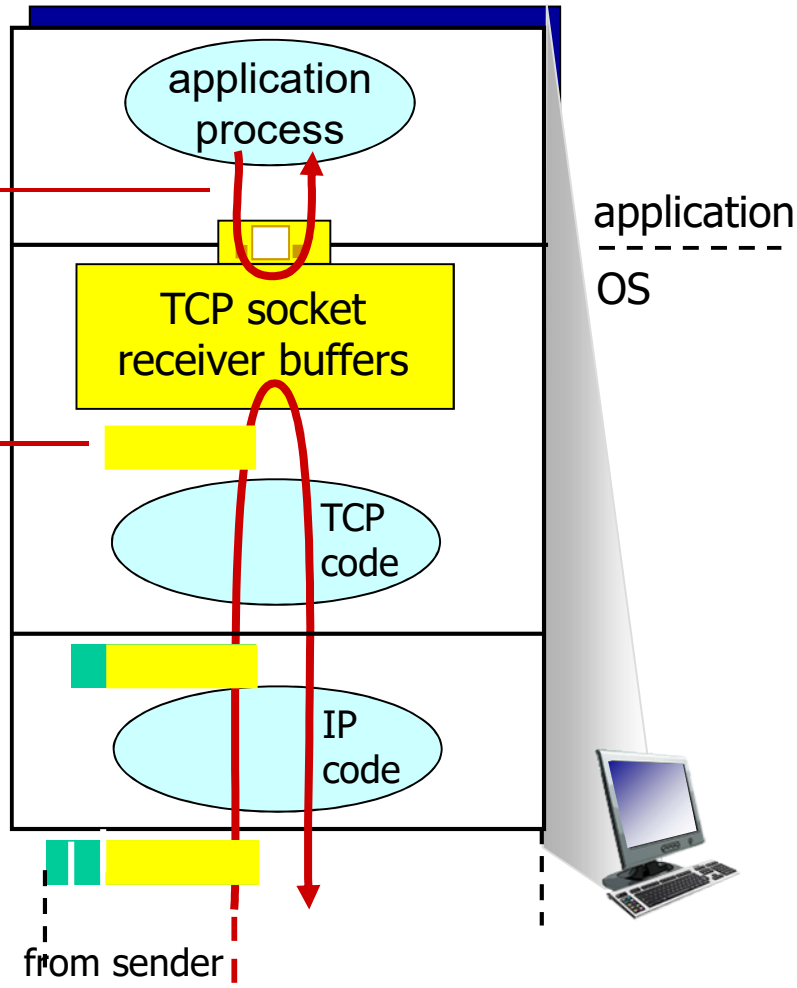
3.6 principles of congestion control

3.7 TCP congestion control

TCP flow control

application may remove data from TCP socket buffers ….

… slower than TCP receiver is delivering (sender is sending)



receiver protocol stack

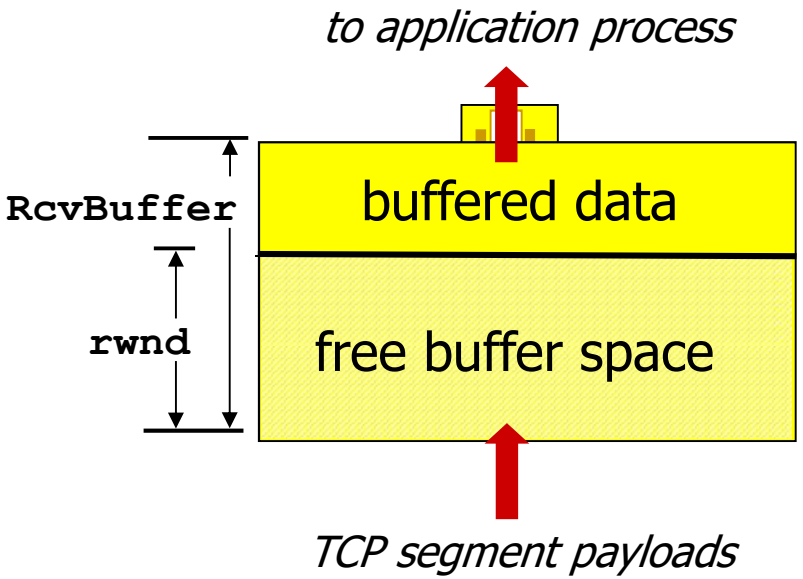
flow control

receiver controls sender, so sender won't overflow receiver's buffer by transmitting too much, too fast

TCP flow control

- receiver "advertises" free buffer space by including **`rwnd`** value in TCP header of receiver-to-sender segments
  - **`RcvBuffer`** size set via socket options (typical default is 4096 bytes)
  - many operating systems autoadjust **`RcvBuffer`**
- sender limits amount of unacked ("in-flight") data to receiver's **`rwnd`** value
- guarantees receive buffer will not overflow

AP의 읽기 : rwnd 증가



TCP의 수신: rwnd 감소

TCP flow control

- 버퍼 크기 : B
- AP가 마지막 읽은 바이트 번호 : R
- TCP에 마지막 도착한 바이트 번호 : A

➢ 현재의 수신윈도우 크기(rwnd)는?

TCP flow control

- 수신윈도우 크기 : rwnd
- 최종 송신 바이트 번호 : S
- 최종 확인(Ack) 번호 : A

➢ rwnd, S, A에 유지되어야 하는 관계식은?

TCP flow control

- 수신윈도우 크기(rwnd)는 누가 언제 어떻게 알려주어야 하는가?

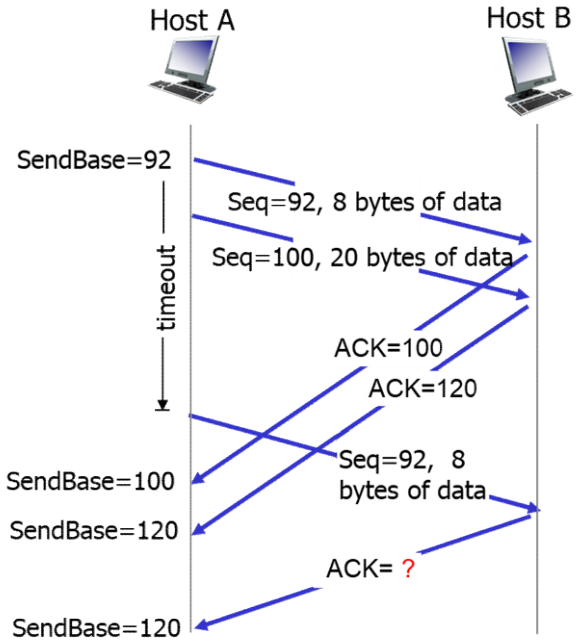
TCP flow control

- 수신윈도우 크기(rwnd)가 0이 되는 경우는 어떤 경우인가?
- 수신윈도우 크기(rwnd)가 0이 되면 어떤 문제가 발생할 수 있는가?

After-study Test :

1) TCP의 신뢰 전송을 잘못 설명한 것은?

① 바이트 스트림 전송을 지원한다.
② 누적 수신확인(cumulative ack)을 지원한다.
③ 가장 오래된 세그먼트에 대한 1개의 타이머를 유지한다.
④ 수신자는 버퍼를 유지하지 않는다.



2) 오른쪽 그림의 마지막 ACK 번호는 무엇인가?

① 93
② 100
③ 101
④ 120

3) TCP는 누적 수신확인을 지원한다. 동일한 확인번호를 가진 ACK이 중복되어 도착하는 경우는 어떤 경우인가?

① 라우터 혼잡 상황이 심각하여 지속적으로 패킷 손실 발생
② 라우터 혼잡 상황이 경미하여 순간적으로 패킷 손실 발생
③ 라우터 혼잡 상황이 심각하여 순간적으로 패킷 손실 발생
④ 라우터 혼잡 상황이 경미하여 지속적으로 패킷 손실 발생

4) 아직 타임아웃이 발생하지 않았지만 동일한 ACK가 중복되어 여러 번 수신되었다. 어떻게 하는 것이 바람직한가?

① 타임아웃이 될 때까지 기다린다.
② 패킷이 손실되었음으로 마지막으로 전송한 세그먼트를 재전송한다.
③ 패킷이 손실되었음으로 처음 전송한 세그먼트를 재전송한다.
④ 패킷이 손실되었음으로 ACK 수신번호에 해당하는 세그먼트를 재전송한다.

5) 버퍼의 크기가 B이고 응용 프로세스가 마지막으로 읽은 바이트 번호가 R이다. TCP에 의해 수신된 마지막 바이트 번호가 A일 때 수신윈도우(rwnd)의 크기는 얼마인가?

① B
② B - R
③ B – R + A
④ B – R + A + 1

6) 수신윈도우의 크기가 rwnd이고, 최종 송신 바이트 번호는 S이며, 최종 수신확인(ack) 번호는 A이다. rwnd, S, A간에 유지되어야 하는 관계는?

① S – A ≤ rwnd
② S – A ≥ rwnd
③ S – A + 1 ≤ rwnd
④ S – A + 1≥ rwnd