Lecture 13: Transport Layer Flow and Congestion Control

COMP 332, Spring 2018

Victoria Manfredi

WESLEYAN UNIVERSITY

Acknowledgements: materials adapted from Computer Networking: A Top Down Approach 7th edition: ©1996-2016, J.F Kurose and K.W. Ross, All Rights Reserved as well as from slides by Abraham Matta at Boston University, and some material from Computer Networks by Tannenbaum and Wetherall.

Today

1. Announcements
   - exam wed!
   - graded homework update
2. TCP flow control
3. Causes and costs of congestion
4. TCP congestion control
5. Midterm overview
   - exam format

TCP
FLOW CONTROL

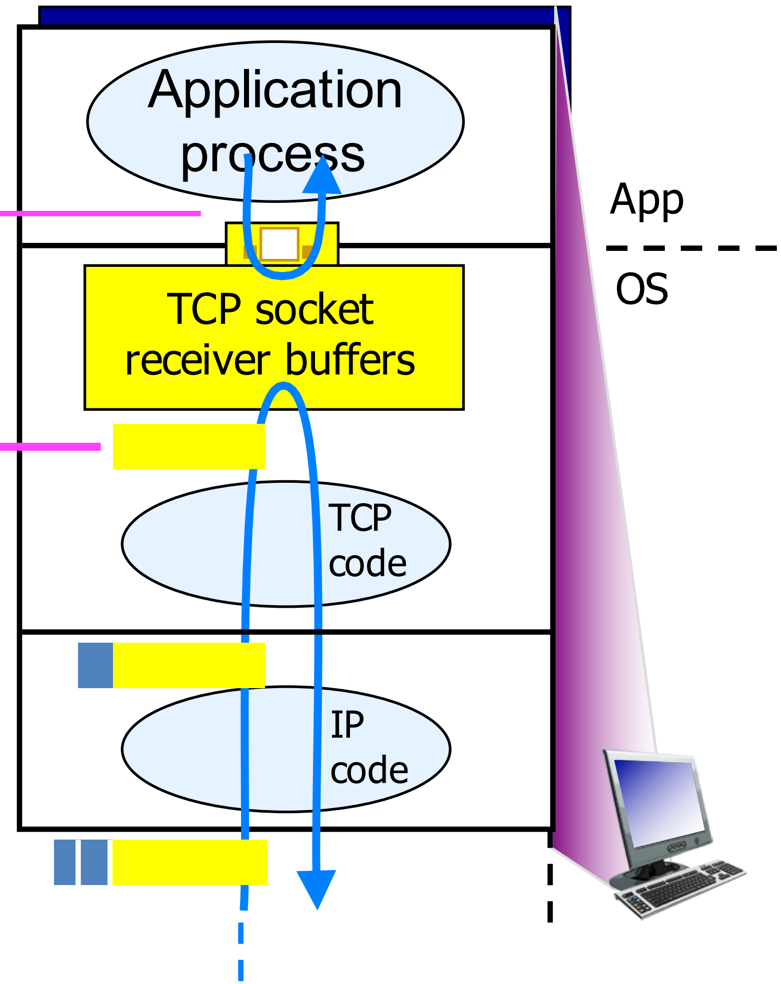
What if sender overwhelms receiver?

Problem

Application may remove data from TCP socket buffers ….

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

Receiver protocol stack



from sender

TCP flow control

Receiver provides feedback to sender

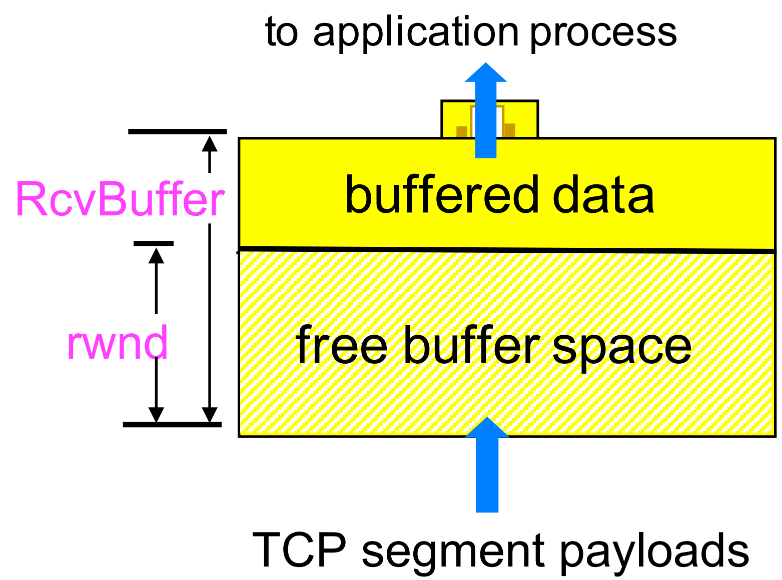
- so sender doesn't overflow receiver's buffer
- sender and receiver each maintain window

Receiver

- rwnd: free space in RcvBuffer
- puts rwnd in TCP header of receiver-to-sender segments

Sender

- limits unacked data to rwnd
- ensures RcvBuffer will not overflow



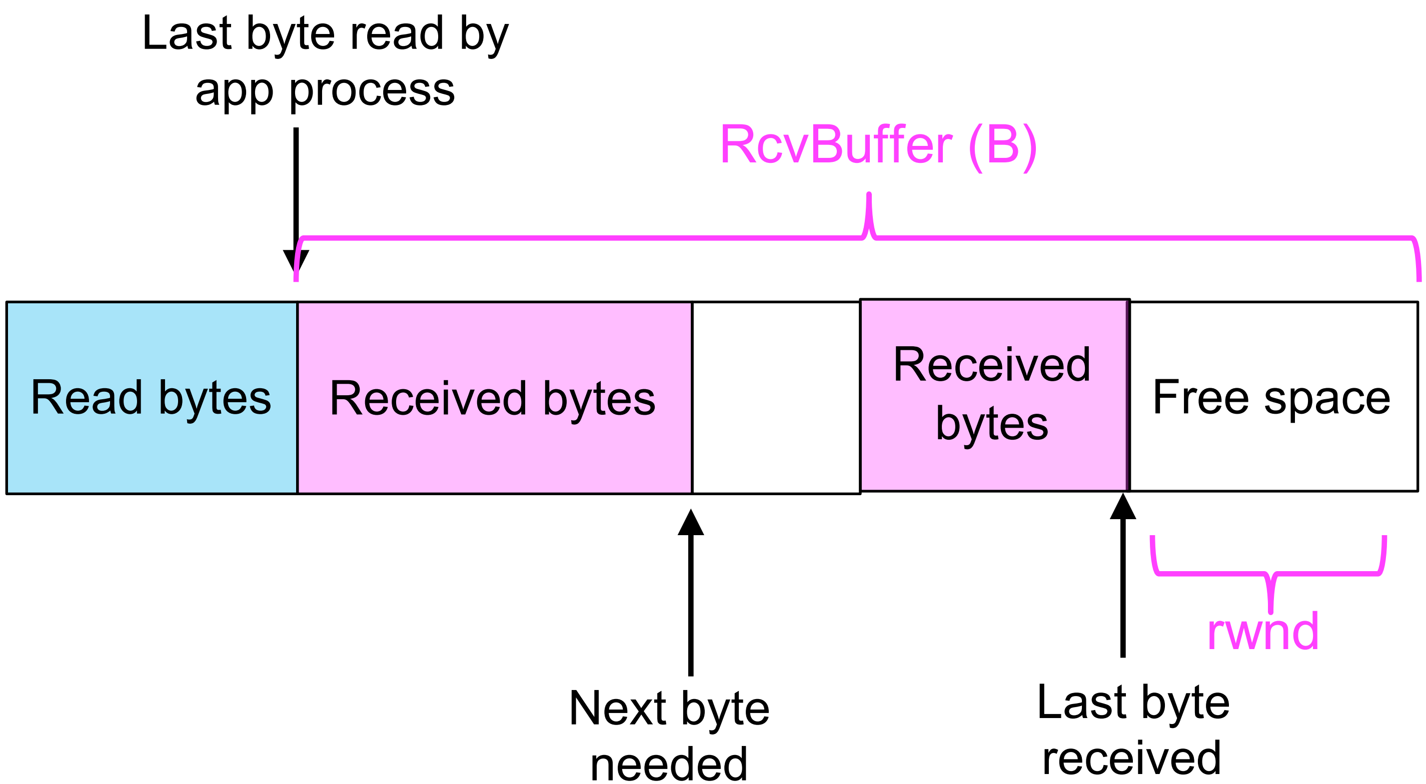
Receiver-side buffering

rwnd

```
▼ Transmission Control Protocol, Src Port: 443 (443), Dst Port: 52232 (52232), Seq: 0, Ack: 1,
    Source Port: 443
    Destination Port: 52232
    [Stream index: 0]
    [TCP Segment Len: 0]
    Sequence number: 0    (relative sequence number)
    Acknowledgment number: 1    (relative ack number)
    Header Length: 32 bytes
  ▼ Flags: 0x012 (SYN, ACK)
      000. .... .... = Reserved: Not set
      ...0 .... .... = Nonce: Not set
      .... 0... .... = Congestion Window Reduced (CWR): Not set
      .... .0.. .... = ECN-Echo: Not set
      .... ..0. .... = Urgent: Not set
      .... ...1 .... = Acknowledgment: Set
      .... .... 0... = Push: Not set
      .... .... .0.. = Reset: Not set
    ▶ .... .... ..1. = Syn: Set
      .... .... ...0 = Fin: Not set
      [TCP Flags: *******A**S*]
    Window size value: 8190
    [Calculated window size: 8190]
```

Receiver use of rwnd

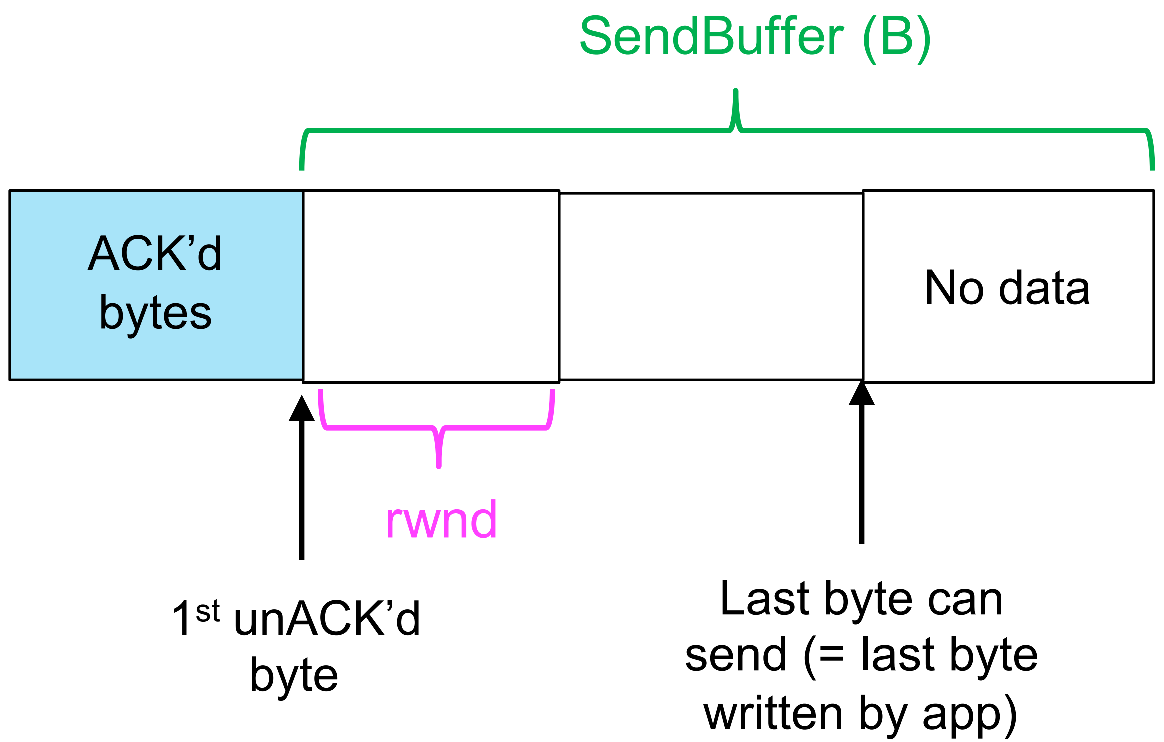
Keeps track of space in its RcvBuffer



rwnd = B – (last byte received – last byte read)

Sender use of rwnd

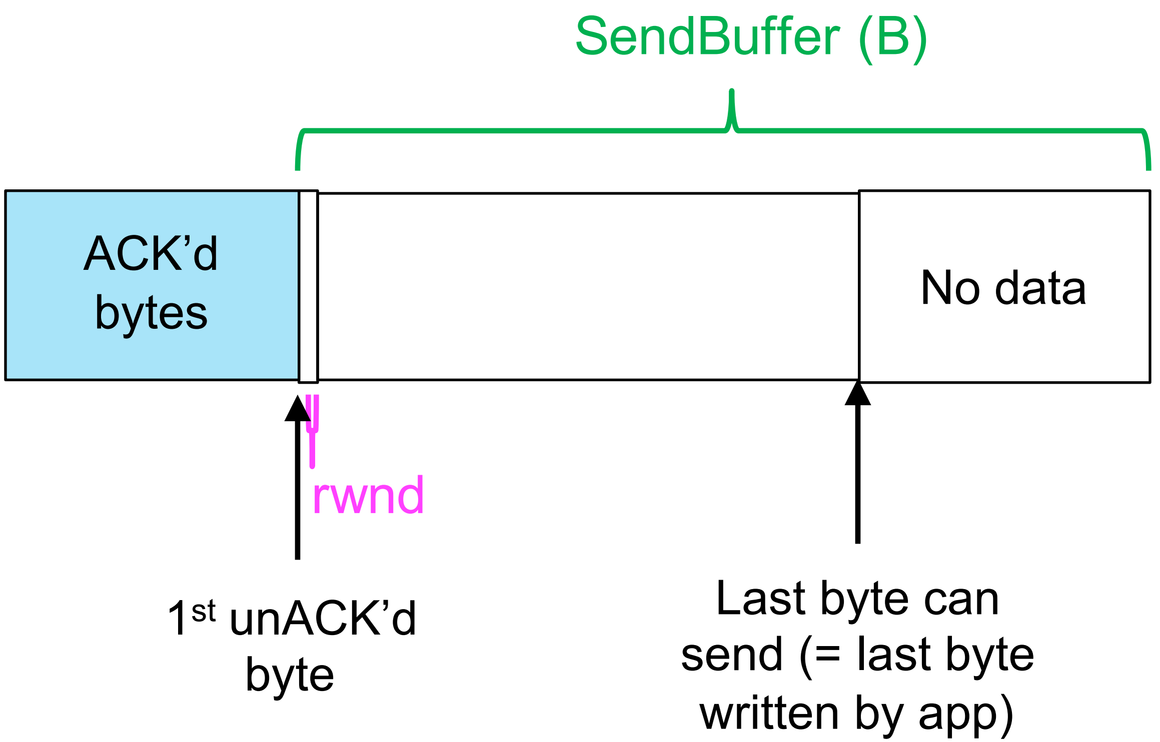
Limits # of in-flight segments of sender



Sending rate limited to: rwnd bytes/RTT seconds

Sender use of rwnd

Problem: if rwnd = 0, what happens?



No ACKs sent

- but then receiver has no way to let sender know rwnd increased

Solution

- send segments with 1 byte of data, which receiver ACKs

Congestion

CAUSES AND COSTS

What if sender overwhelms network?

Receive buffer is not only resource limitation

- every pkt has to travel through path of routers
- routers may be congested, have long queues …

Causes of network congestion

- many senders competing for network resources
- senders lacking knowledge
  - amount of resources available (bandwidth)
  - # of other senders competing

Costs of network congestion

As queues in bottleneck link fill up

- large packet delays
- dropped packets

As timeouts expire at sender due to delays/drops

- packets retransmitted

Bad feedback loop!

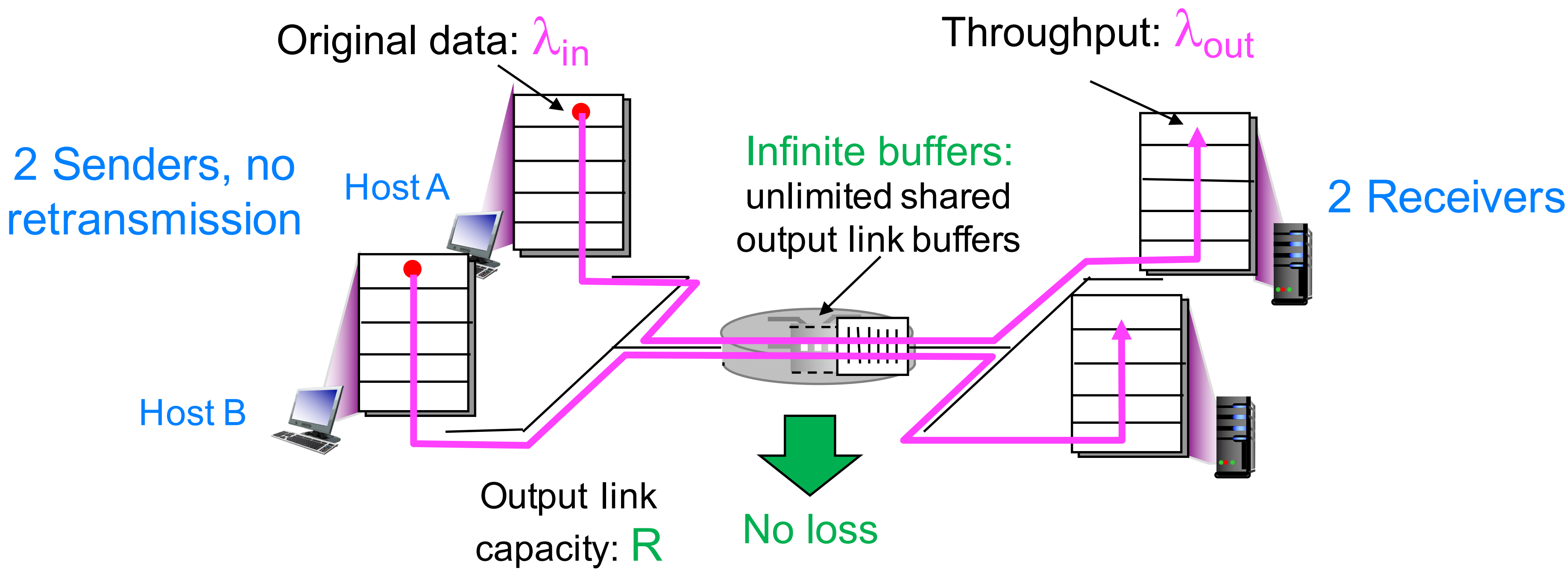
Problem

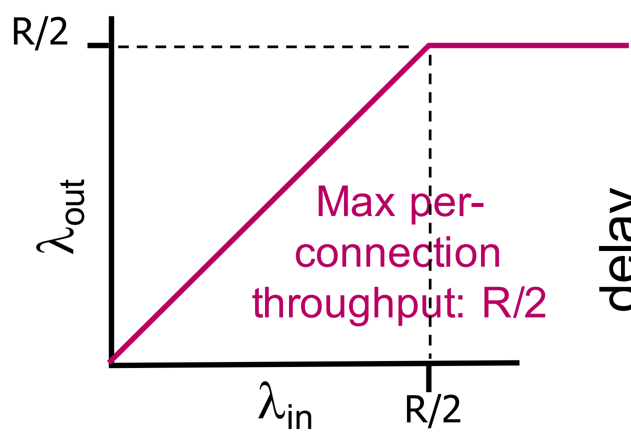
- retransmission treats symptoms but not underlying problem

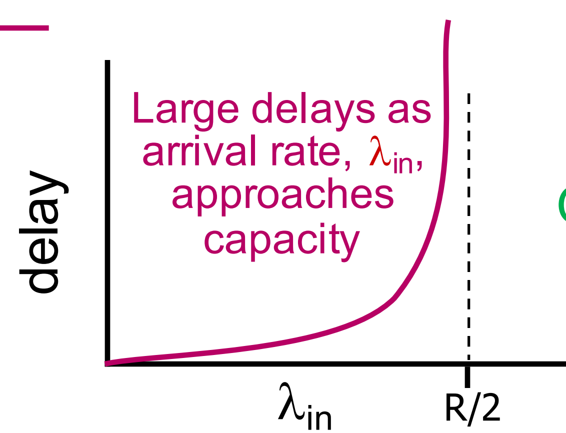
Q: How to solve underlying problem of congestion?

- reduce sending rate
- what should sending rate be?
  - depends on available bandwidth
  - sender increases/decreases sending rate based on congestion level

Scenario 1: no retransmission

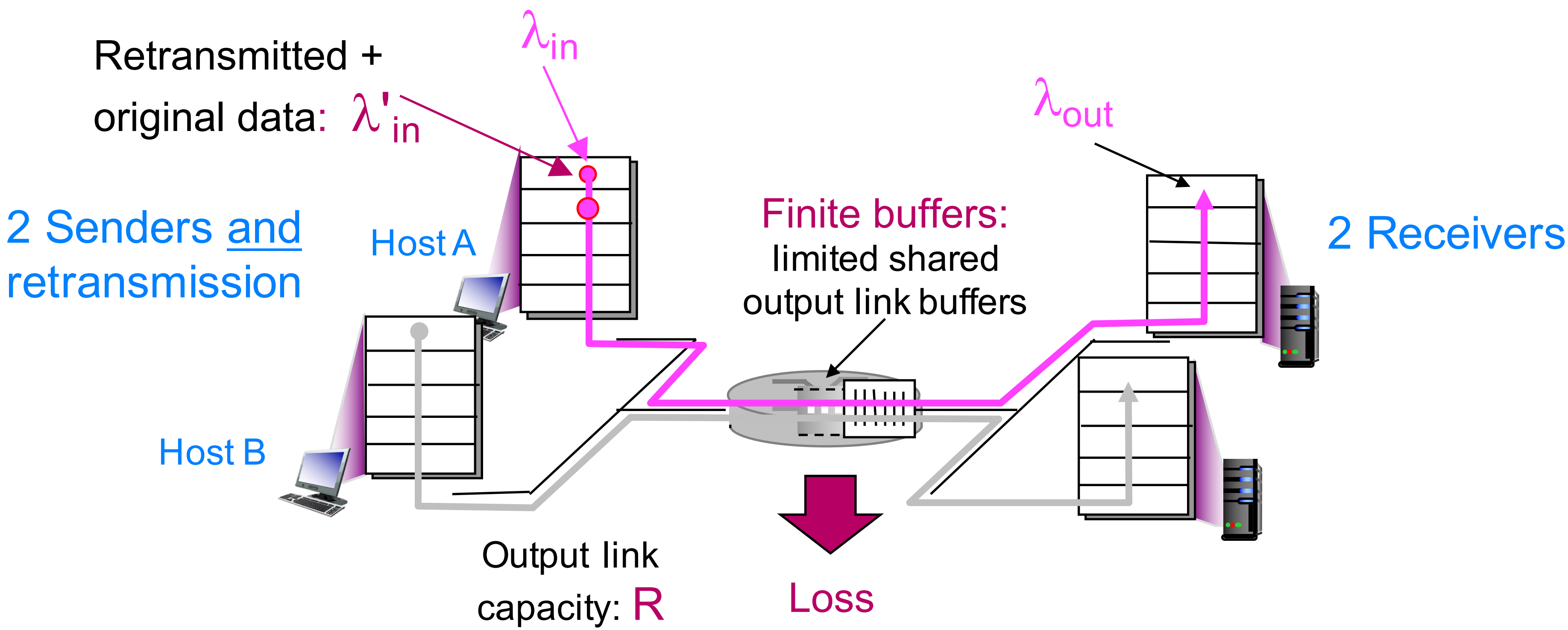






Q: Why R/2?

Scenario 2: retransmission



Sender retransmits timed-out packet

- $\lambda_{in} = \lambda_{out}$: app-layer input equals app-layer output
- $\lambda'_{in} \geq \lambda_{in}$: transport-layer input includes retransmissions

Performance now depends on how retransmission performed

Midterm
OVERVIEW

Midterm overview

In class on Wednesday Mar. 28

- closed book, closed notes
- covers material in lectures 1 to 12

Will not ask questions on

- probability
- distributed hash tables

5 questions

- app layer short questions
- transport layer short questions
- sequence number ranges
- caching and delays
- reliable data transport protocols

Problems 1 and 2

App layer and transport layer short questions

- 6 in total
- similar to review questions in book
- should only need to write a few sentences to answer

Problem 3

Do some reasoning about sequence #s

- for a given receiver window range of sequence #s, what range of sequence #s can sender have?

Problems 4

Given a network that can use caching for DNS and HTTP

- give all messages that must be sent when a user enters a URL
- sum up the delays incurred
- (ignoring TCP handshaking)

Problem 5

Design a reliable data transfer protocol

- given channel characteristics design most efficient protocol
- be able to design reliable data transfer protocol like Stop-and-wait, know your timeline diagrams