Flow Control

Lecture 14

http://www.cs.rutgers.edu/~sn624/352-S22

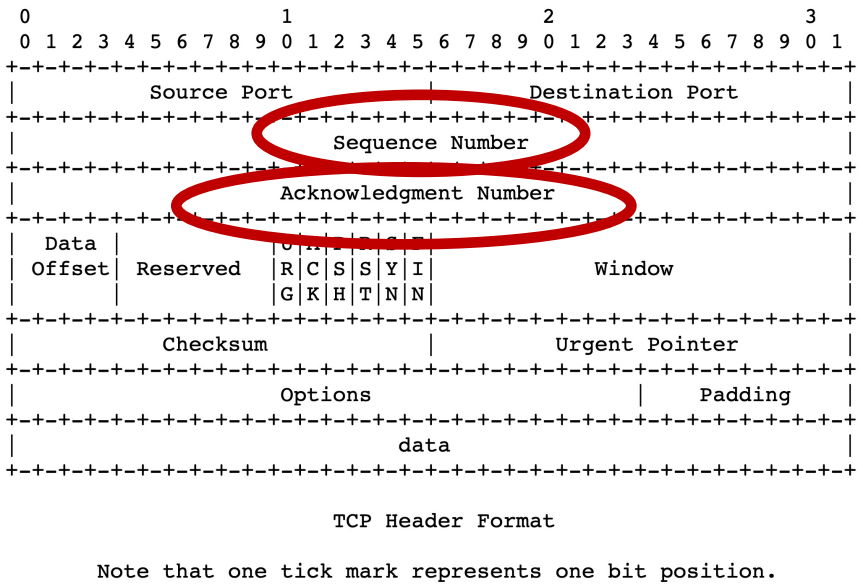
Srinivas Narayana

RUTGERS
UNIVERSITY | NEW BRUNSWICK

Quick recap of concepts

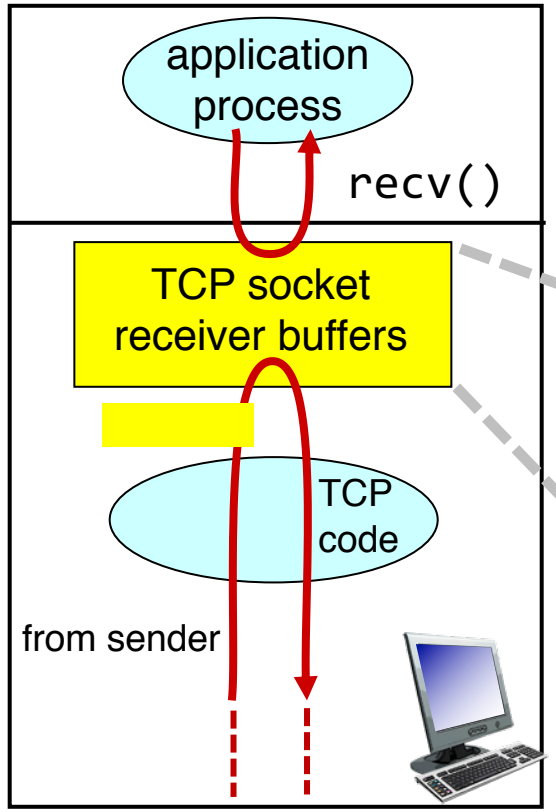
TCP:

Connection-oriented



TCP Header Format

Note that one tick mark represents one bit position.



receiver TCP interaction

Selective repeat

Cumulative ACK

`ACK x`

Selective ACK

```
ACK x
SACK x1-x2,x3-x4,x5-x6
```

Cumulative ACK

Precision of info:

<

Retransmissions

<

Complexity, bugs, …

>

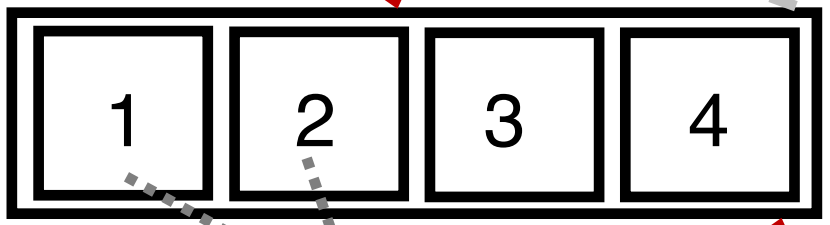
Selective ACK

Ordering

Reassembly

App

`recv()`



Data with sequence #

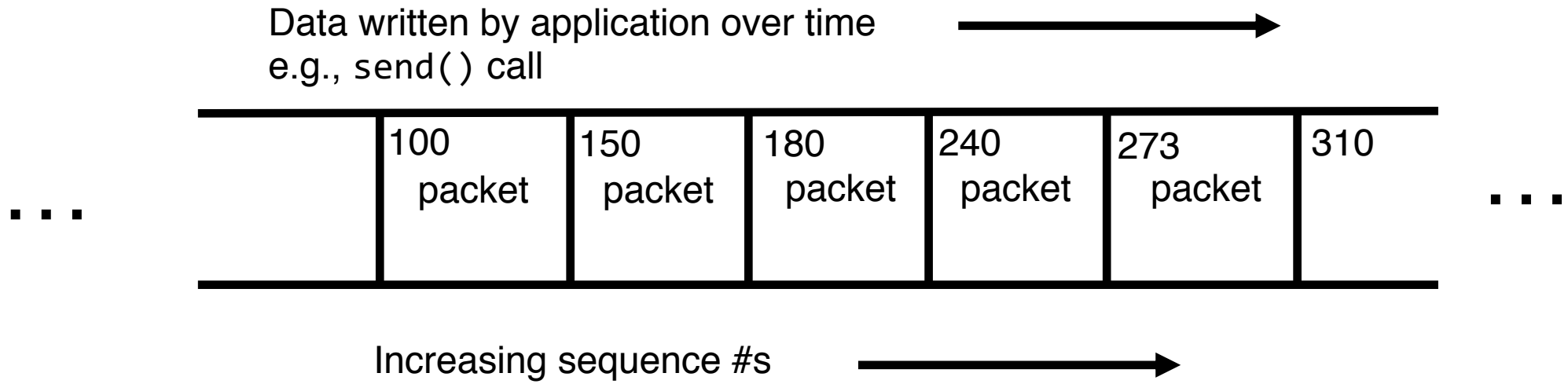
Implications of ordered delivery

- Packets cannot be delivered to the application if there is an in-order packet missing from the receiver's buffer
  - The receiver can only buffer so much out-of-order data
  - Subsequent out-of-order packets dropped
  - It won't matter that those packets successfully arrive at the receiver from the sender over the network

- TCP application-level throughput will suffer if there is too much packet reordering in the network
  - Data may have reached the receiver but won't be delivered to apps upon a `recv()` (...or may not even be buffered!)

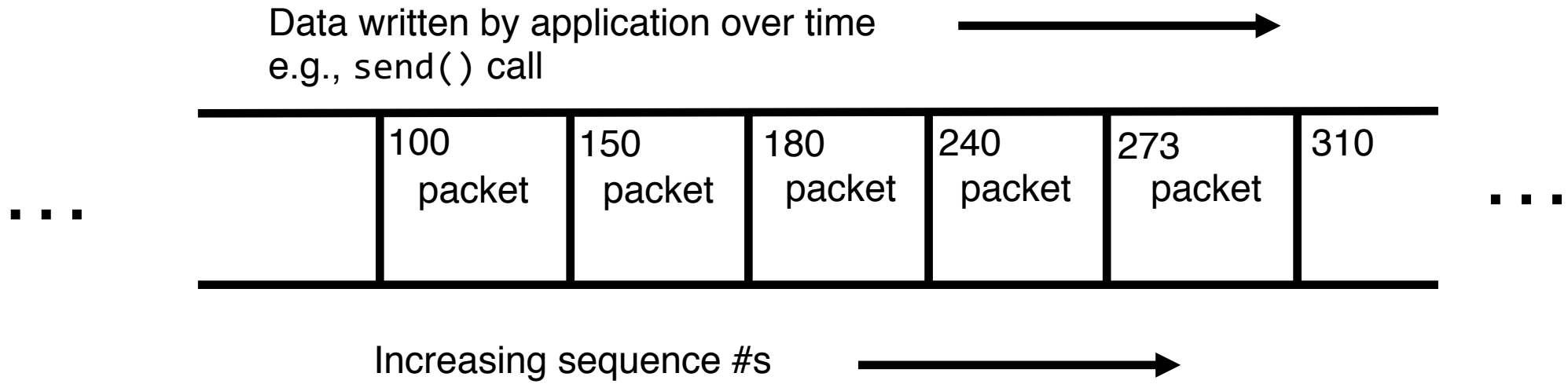
Stream-Oriented Data Transfer

Sequence numbers in the app's stream



TCP uses byte sequence numbers

Sequence numbers in the app's stream

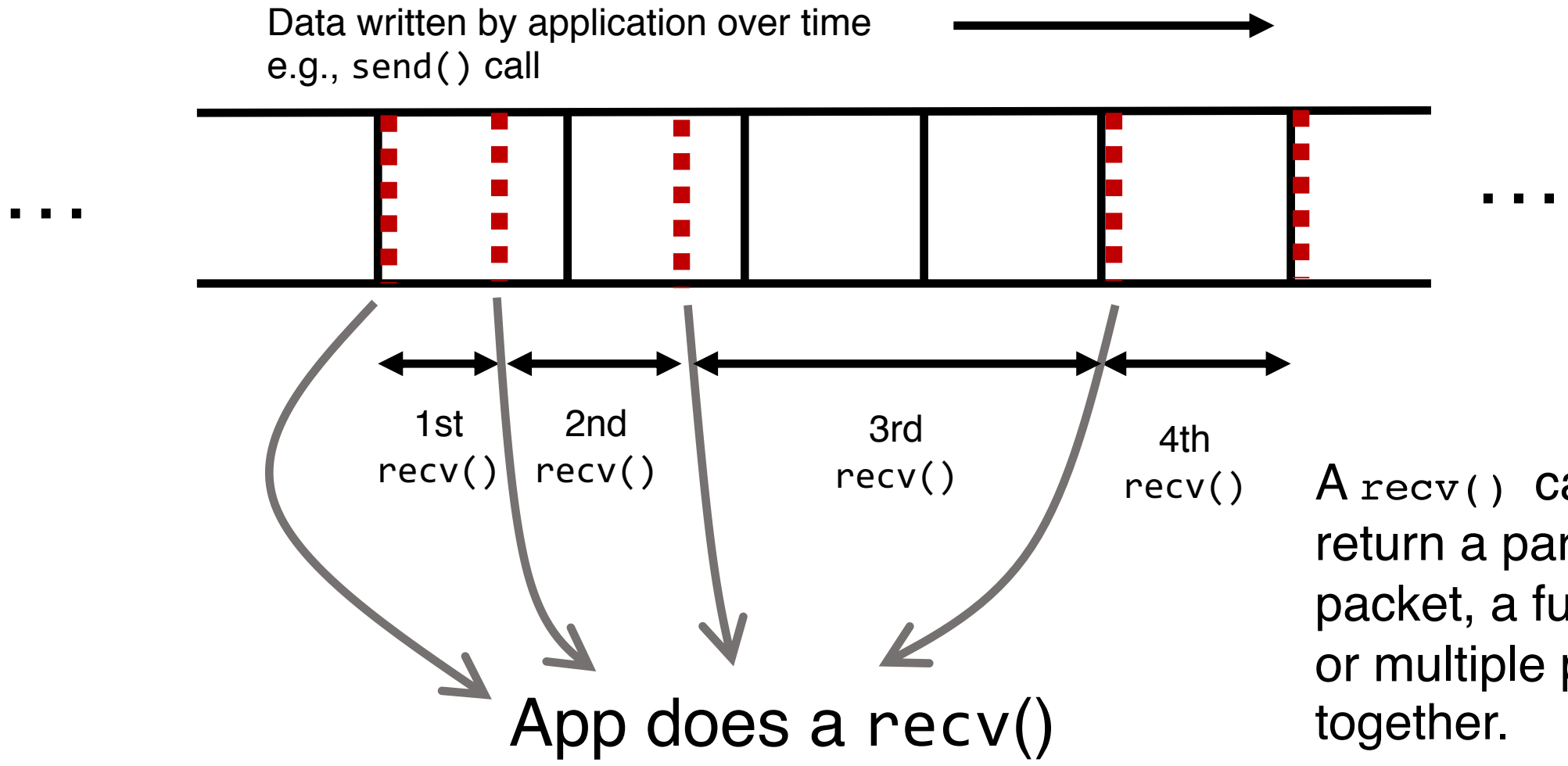


Packet boundaries aren't important for TCP software

TCP is a stream-oriented protocol

(We use `SOCK_STREAM` when creating sockets)

Sequence numbers in the app's stream



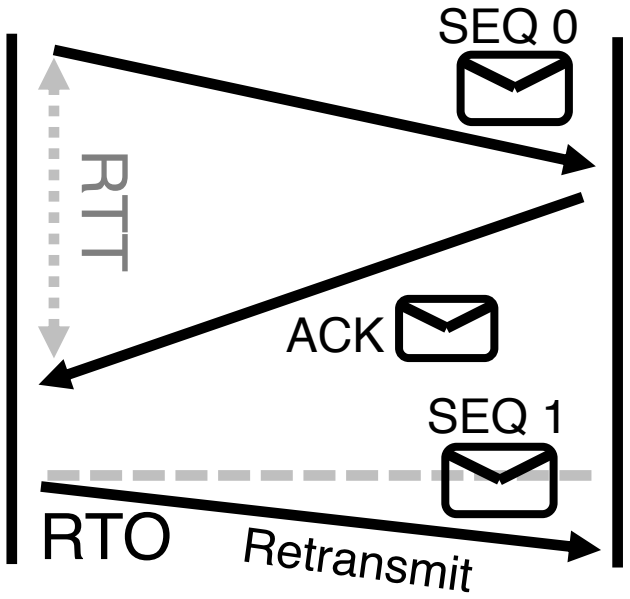
A `recv()` call may return a part of a packet, a full packet, or multiple packets together.

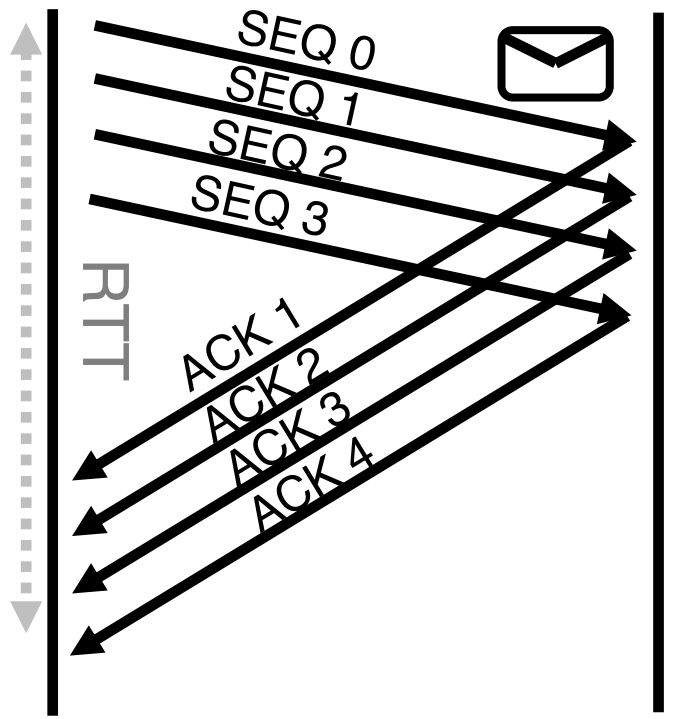
= window size

Proportional to throughput

How much data to keep in flight?

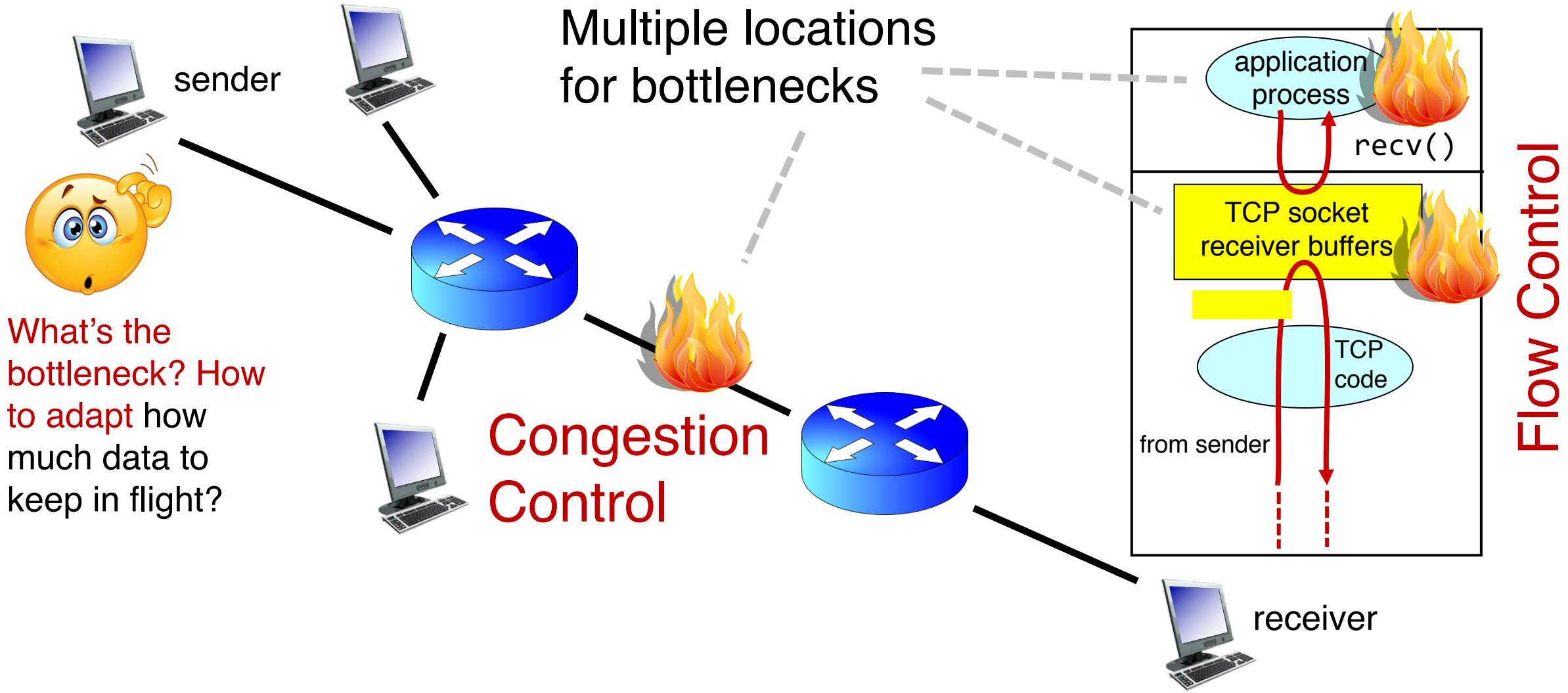
Stop and Wait





Pipelined Reliability

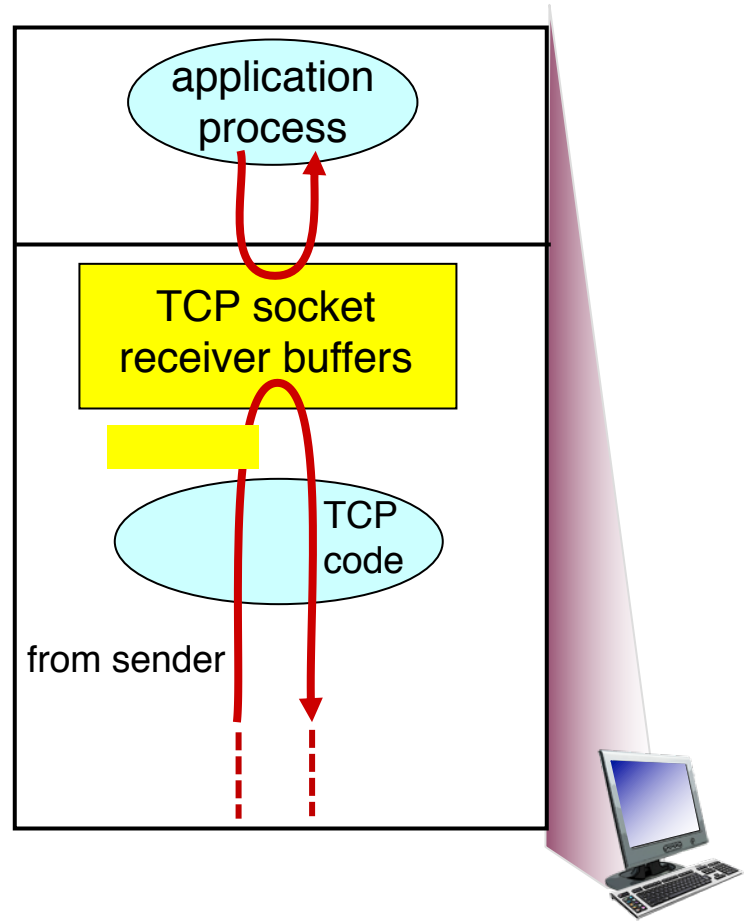
We want to increase throughput, but …



Flow Control

Socket buffers can become full

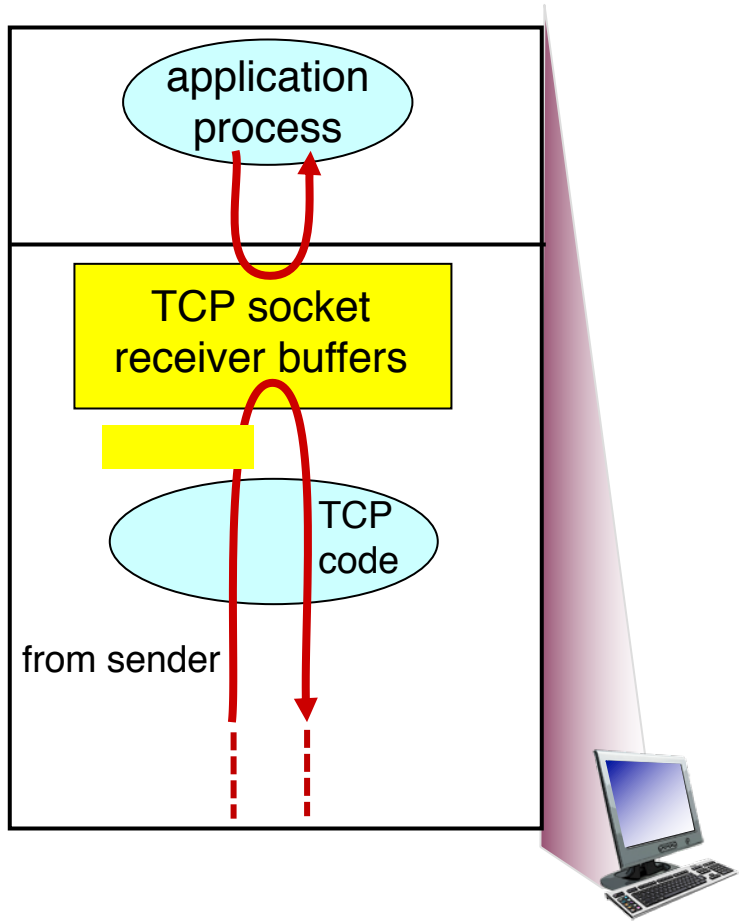
- Applications may read data slower than the sender is pushing data in
  - Example: what if an app infrequently or never calls `recv()`?
- There may be too much reordering or packet loss in the network
  - What if the first few bytes of a window are lost or delayed?
- Receivers can only buffer so much before dropping subsequent data



TCP receiver

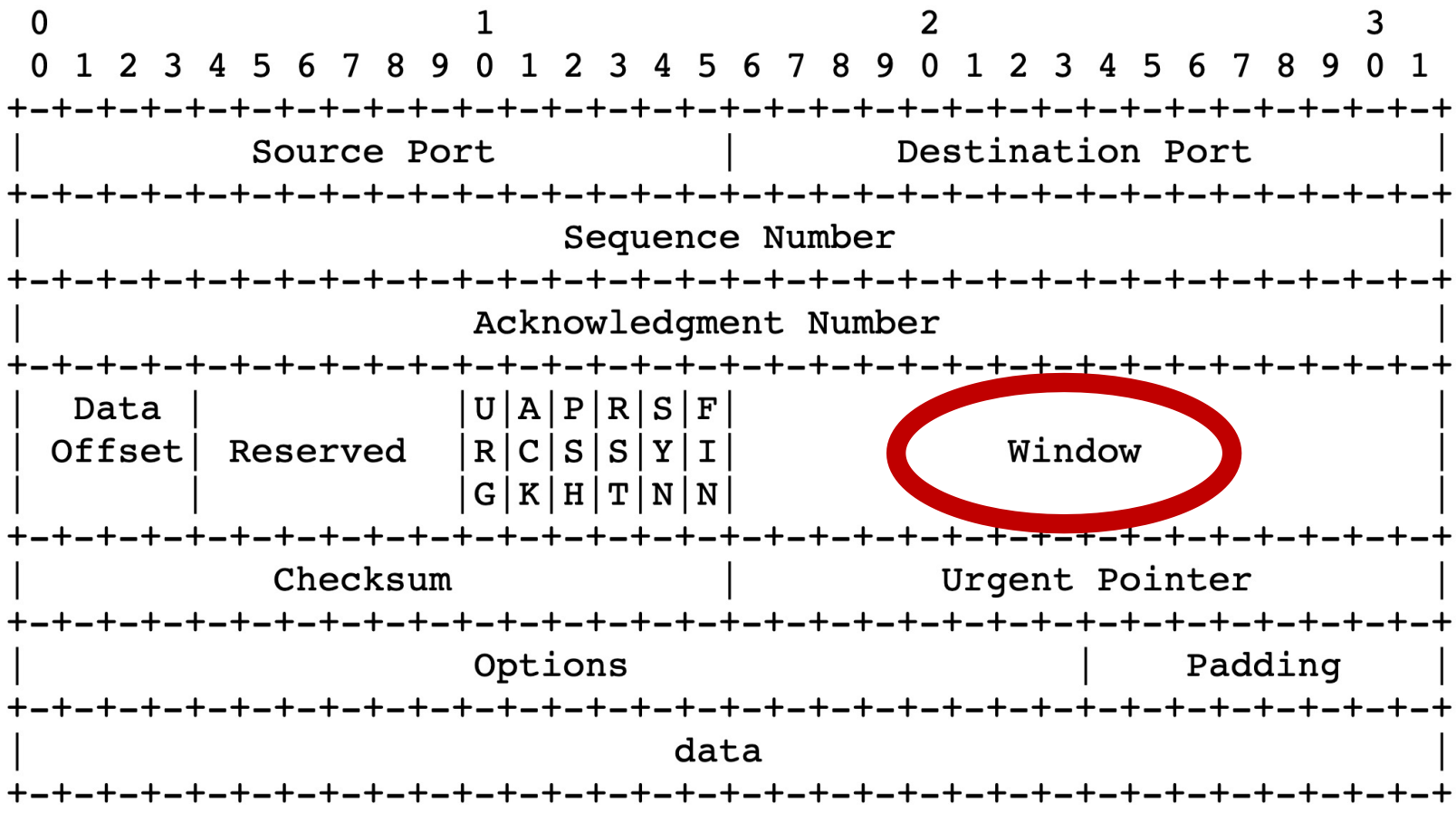
Goal: avoid drops due to buffer fill

- Have a TCP sender only send as much as the free buffer space available at the receiver.
- *Amount of free buffer varies over time!*
- TCP implements flow control
- Receiver's ACK contains the amount of data the sender can transmit without running out the receiver's socket buffer
- This number is called the advertised window size



receiver protocol stack

Flow control in TCP headers

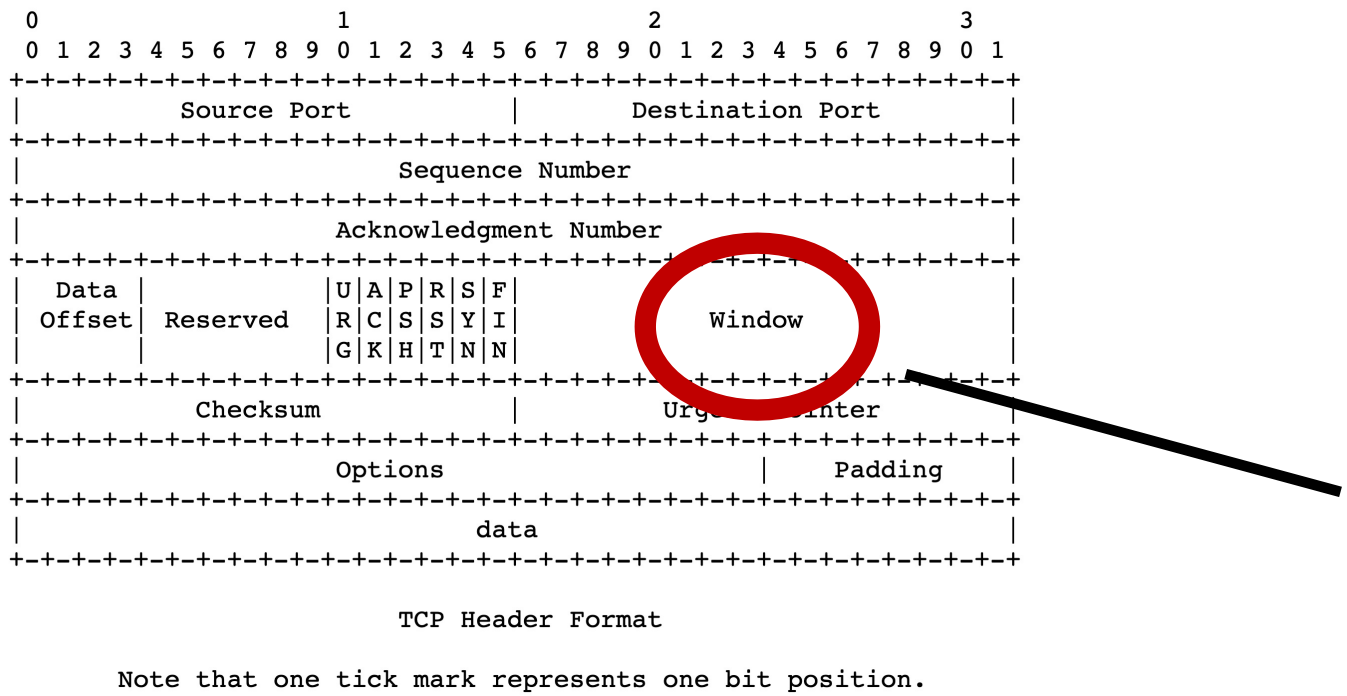


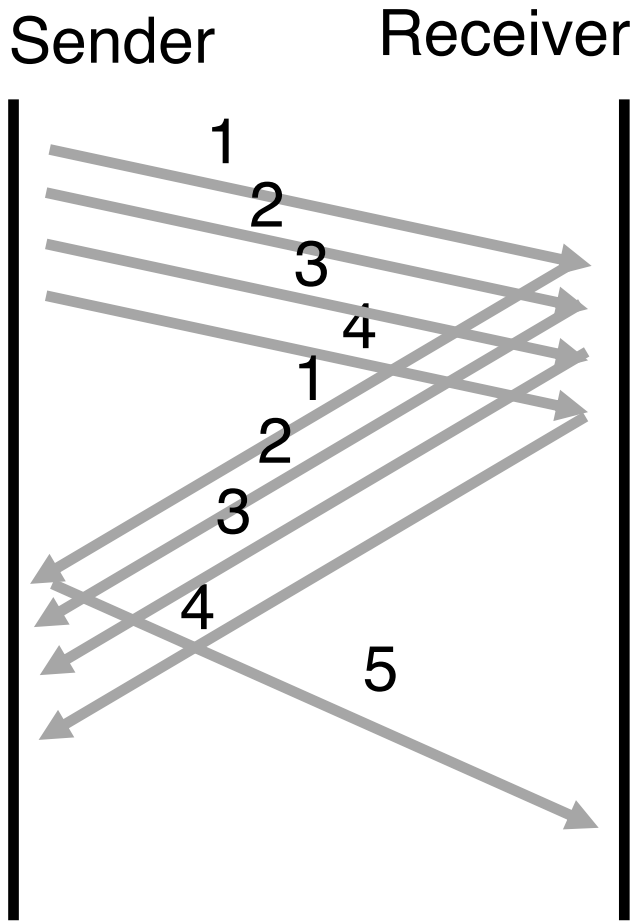
TCP Header Format

Note that one tick mark represents one bit position.

TCP flow control

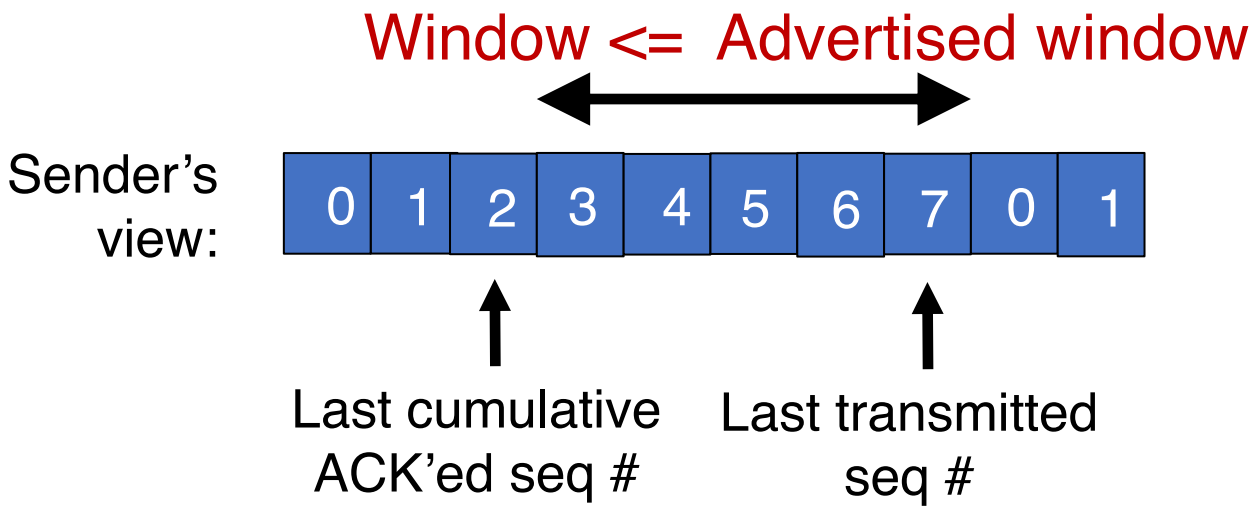
- Receiver advertises to sender (in the ACK) how much free buffer is available

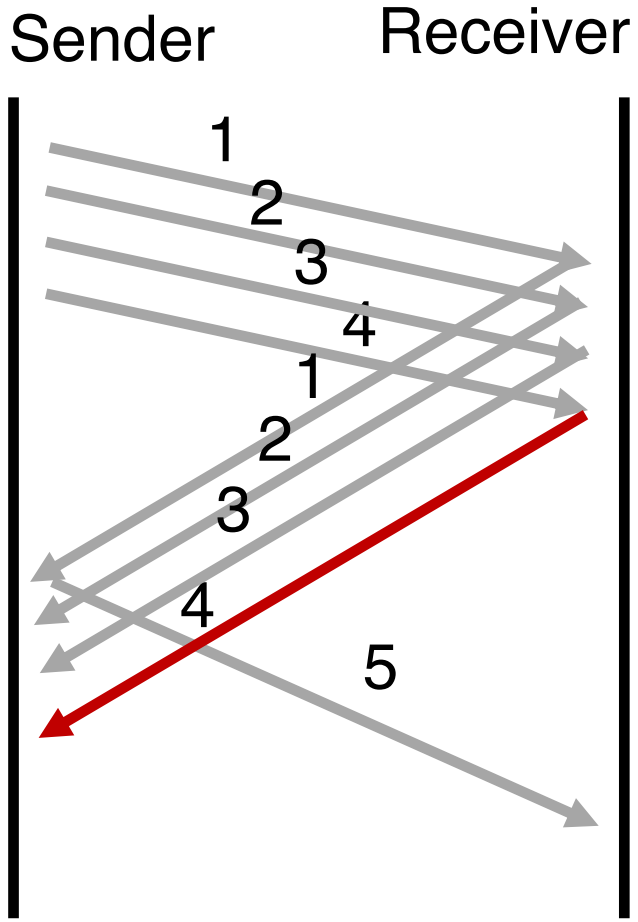




TCP flow control

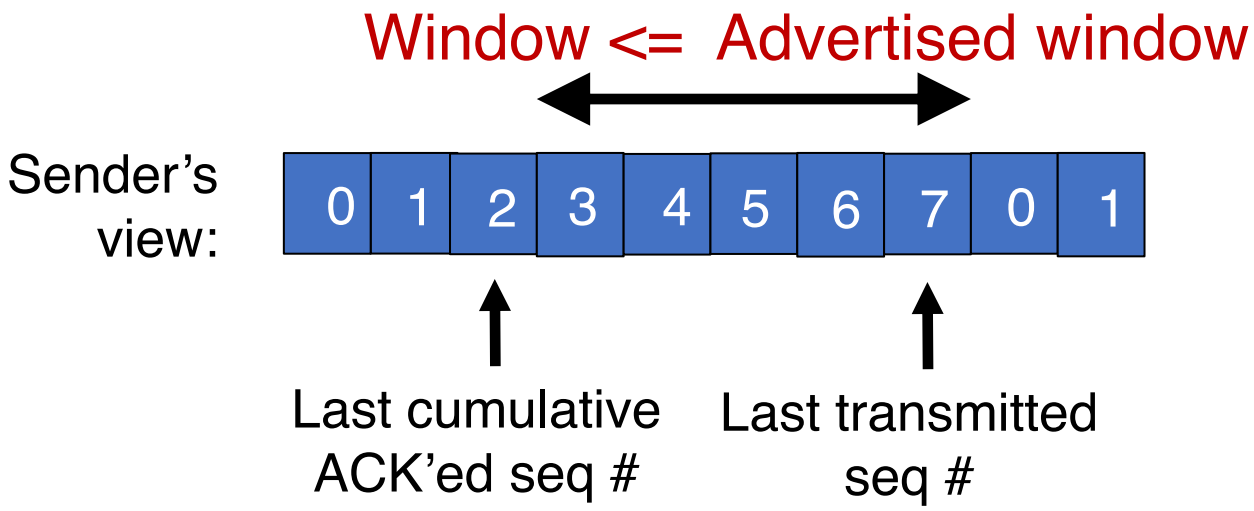
- Subsequently, the sender's sliding window cannot be larger than this value
- Restriction on new sequence numbers that can be transmitted
- == restriction on sending rate!

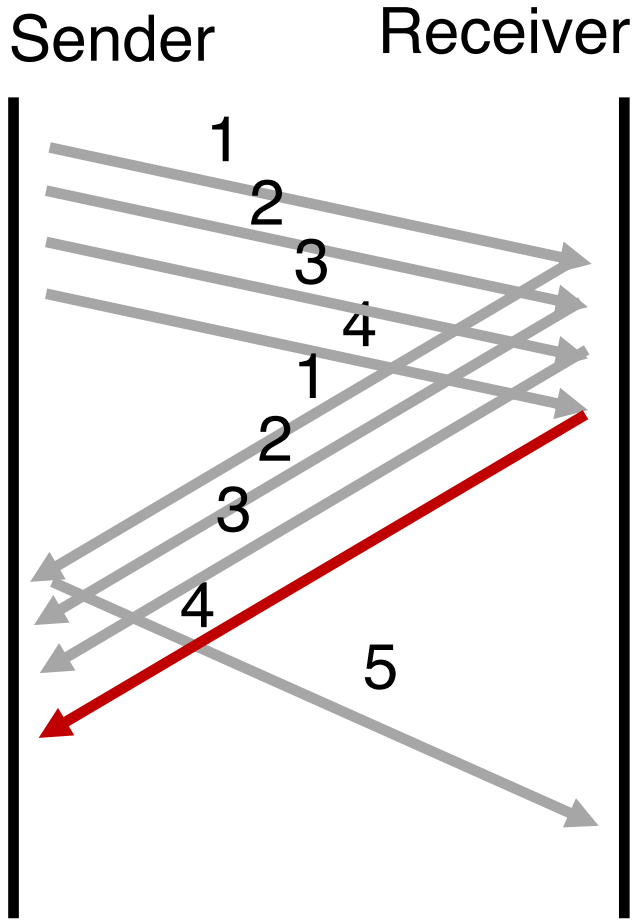




TCP flow control

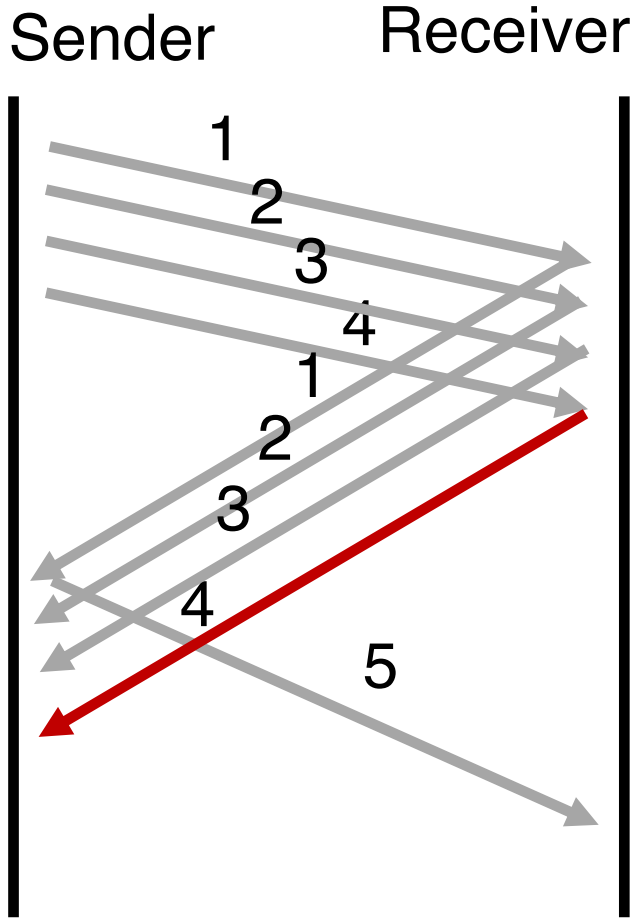
- If receiver app is too slow reading data:
  - receiver socket buffer fills up
  - So, advertised window shrinks
  - So, sender's window shrinks
  - So, sender's sending rate reduces

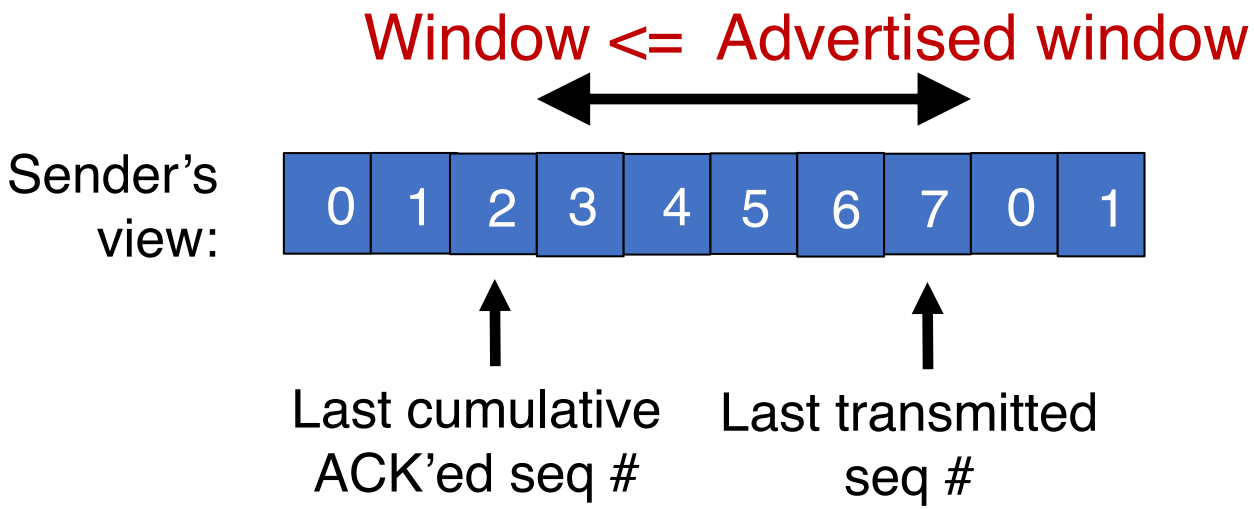




TCP flow control

Flow control matches the sender's write speed to the receiver's read speed.





Sizing the receiver's socket buffer

- Operating systems have a default receiver socket buffer size
  - Listed among `sysctl -a | grep net.inet.tcp` on MAC
  - Listed among `sysctl -a | grep net.ipv4.tcp` on Linux

- If socket buffer is too small, sender can't keep too many packets in flight ➔ lower throughput

- If socket buffer is too large, too much memory consumed per socket

- How big should the receiver socket buffer be?

Sizing the receiver's socket buffer

- Case 1: Suppose the receiving app is reading data too slowly:
  - no amount of receiver buffer can prevent low sender throughput if the connection is long-lived!

Sizing the receiver's socket buffer

- Case 2: Suppose the receiving app reads sufficiently fast *on average* to match the sender's writing speed.
  - Assume the sender has a window of size W.
  - The receiver must use a buffer of size at least W. Why?

- Captures two cases:
- (1) When the first sequence #s in the window are dropped
  - *Selective repeat*: data in window buffered until the ACKs of delivered data (within window) reach sender. Adv. win reduces sender's window
- (2) When the sender sends a burst of data of size W
  - Receiver may not match the *instantaneous* rate of the sender

Summary of flow control

- Keep memory buffers available at the receiver whenever the sender transmits data
- Inform the sender on an on-going basis (each ACK)
- Function #1: match sender speed to receiver speed
- Function #2: reassemble data in order and hold for selective repeat

- Correct socket buffer sizing is important for TCP throughput

Info on (tuning) TCP stack parameters

- https://www.ibm.com/support/knowledgecenter/linuxonibm/liaag/wkvm/wkvm_c_tune_tcpip.htm

- https://cloud.google.com/solutions/tcp-optimization-for-network-performance-in-gcp-and-hybrid