CPSC 317 COMPUTER NETWORKING

2023W2: Transport – Day 6 – Flow + Congestion Control and TCP

1

READING

- Reading: 3.5.2, 3.5.4, 3.5.5

ADMINISTRATION

- PA3 starts today
- Fixing bug with calculation of extension hours used

LEARNING GOALS

- Explain the concept of flow control, and how a sliding window protocol can implement it
- Explain the concept of congestion control, and how a sliding window protocol can implement it

LEARNING GOALS

TCP

- Describe and analyze strategies for controlling the sending window size with respect to congestion
- Describe and analyze strategies for how the receiver can control how much data the sender can send before pausing and waiting for buffers to empty
- Trace the sending of packets/ACKS and how sequence numbers are used in TCP
- Trace how TCP sets up a connection
- Trace how TCP terminates a connection

FLOW CONTROL

- Should we always send the full window size?
- What if the receiving application is slow in accepting new data?
  - Packets will accumulate in the receiver's buffer
  - Eventually buffer will be full, packets will be dropped
  - Immediately re-sending this data does not resolve the problem
- Receiver will notify the sender how much data it can handle
  - This information is usually included in the ACK
  - Sender adjusts its window size based on this information
  - Why can't the receiver just not ACK the extra data?

CONGESTION CONTROL

- What if the network can't handle a full window's worth of data?
  - Packets and ACKs will be dropped by the routers
- Missing ACKs are a sign that there is congestion somewhere (in either direction)
- Sender can reduce sender window once congestion is detected
  - Example: if some number of ACKs are missing in a period of time
- If all packets are ACKed, we can increase the window again

EFFECT ON THROUGHPUT

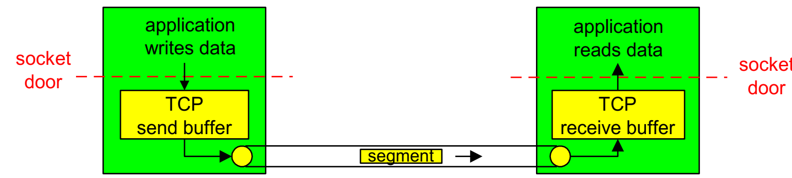
- Throughput is affected by
  - Bandwidth of sender's direct network connection
  - Receiver's specified window size (flow control)
  - Sender's adjusted window size (congestion control)
- If sender's direct connection is not the bottleneck:
  - Another router will experience congestion elsewhere
  - Congestion control will reduce transmission speed

RECAP

- We've built reliable delivery on top of an unreliable network (lower-level protocol)
- We realized we need more packets in flight, and introduced send windows
  - GBN vs SR
- We've talked about flow control (a slow receiver)
- We've talked about congestion control (a busy network)
- Now we are going to see what a real protocol looks like

TCP: OVERVIEW

RFCs: 793, 1122, 1323, 2018, 2581



- point-to-point:
  - exactly two participants
- connection-oriented:
  - handshaking (exchange of control msgs) initializes sender, receiver state before data exchange
- full duplex data:
  - bi-directional data flow in same connection
  - (sometimes we may talk about data flow in only one direction)
- *send & receive buffers*
- reliable, in-order *byte stream:*
  - no "message boundaries"
  - MSS: maximum segment size
- pipelined:
  - TCP congestion and flow control set window size
- flow controlled:
  - sender will not overwhelm receiver

TCP SEGMENT FORMAT

- SYN – conn. start, synchronize seq#
- FIN – conn. close, no new data from sender
- RST – reset conn. (rejected connection)
- ACK – indicate that ACK # is set
- PSH – sender pushed the accumulated send buffer
- URG – segment contains “urgent” data; combined with urgent data pointer, points to the data to be delivered before other data.
- ECE – ECN-Echo, indicate ECN capability
- CWR – congestion window reduced

32 bits

<table>
<tr><td colspan="10">Source port #</td><td>Destination port #</td></tr>
<tr><td colspan="11">Sequence Number</td></tr>
<tr><td colspan="11">Acknowledgement Number</td></tr>
<tr><td>Hdr Len</td><td>Not used</td><td>CWR</td><td>ECE</td><td>URG</td><td>ACK</td><td>PSH</td><td>RST</td><td>SYN</td><td>FIN</td><td>Receiver Window</td></tr>
<tr><td colspan="10">Checksum</td><td>Urgent Data Pointer</td></tr>
<tr><td colspan="11">Options (optional, variable length)</td></tr>
<tr><td colspan="11">Application payload (optional, variable length)</td></tr>
</table>

For information only, you will not be tested on the format of the header

TCP SEQUENCE NUMBERS AND ACKS

- Each byte of a TCP segment has a sequence number
- The sequence number of a segment is the sequence number of its first byte
- ACKs correspond to the first sequence number not yet received (similar to Go-Back-N, but byte-based and +1)
- Receiver stores packets in its own window (like Selective-Repeat)
- ACKs are cumulative (like Go-Back-N)

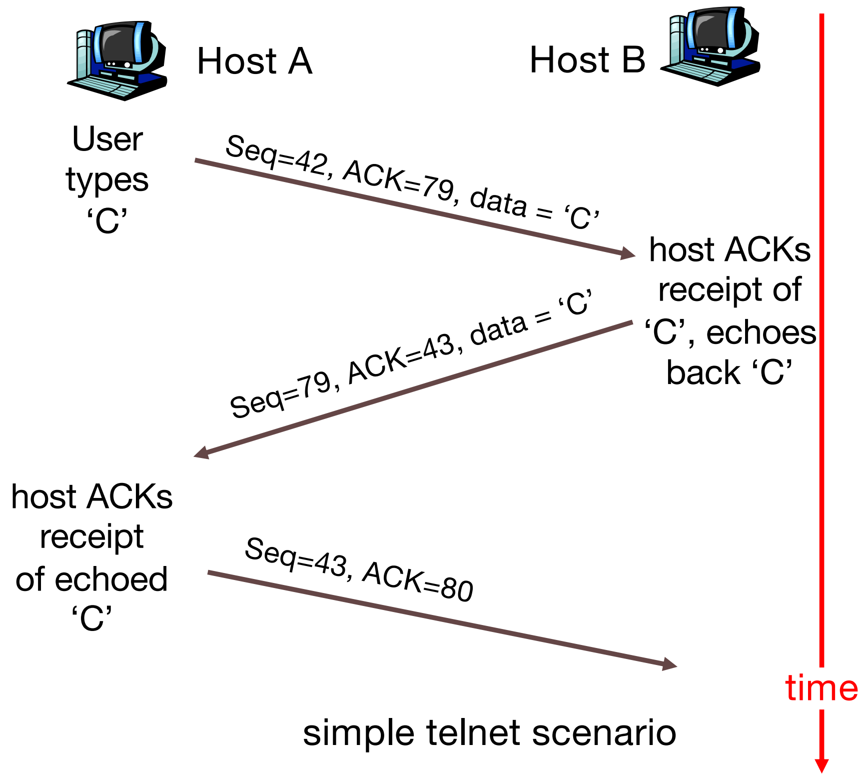
TCP SEQ. #'S AND ACKS

Seq. #'s:

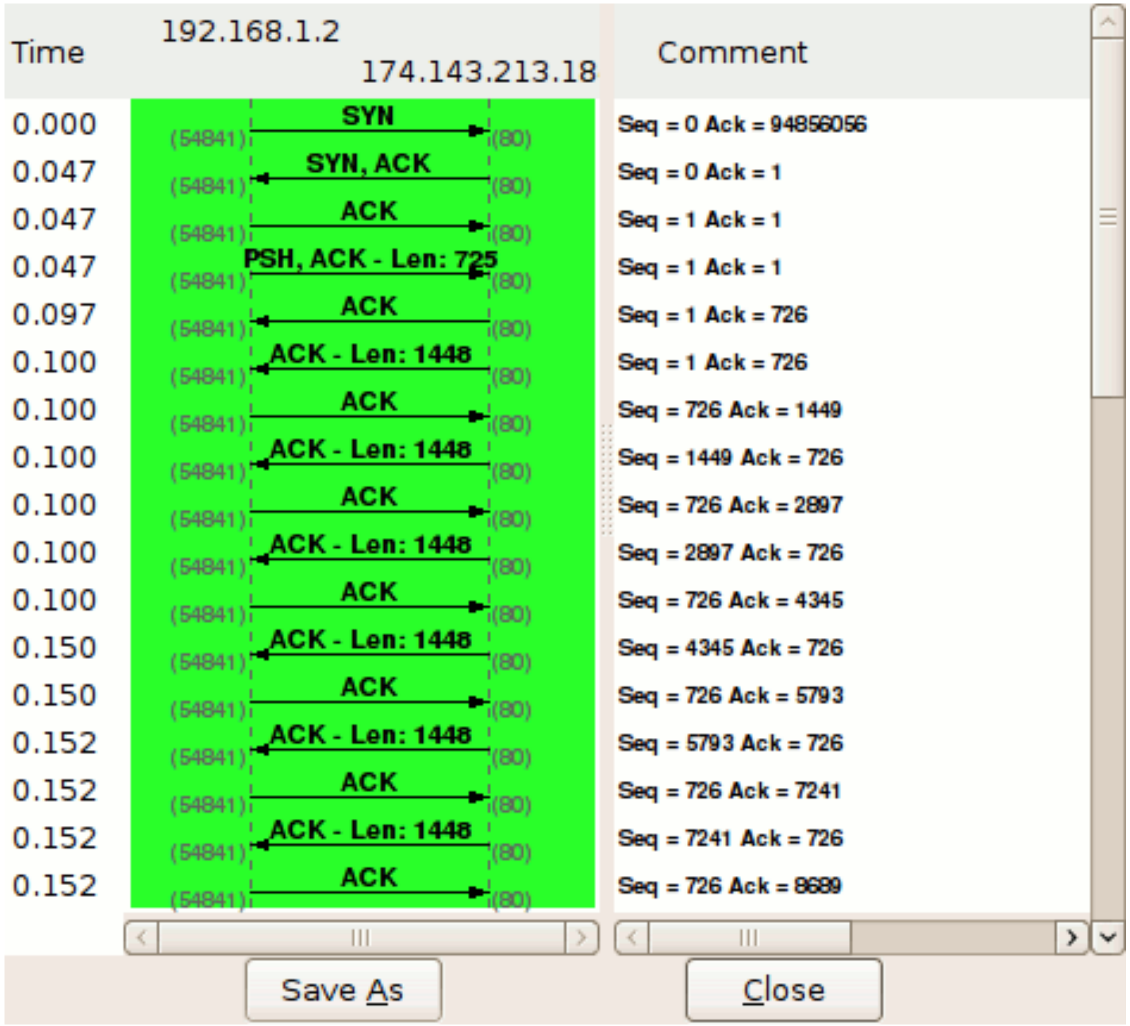
- byte stream "number" of first byte in segment's data

ACKs:

- seq # of next byte expected from other side
- cumulative ACK



simple telnet scenario


Time
192.168.1.2
174.143.213.18
Comment
0.000 SYN (54841) (80) Seq = 0 Ack = 94856056
0.047 SYN, ACK (54841) (80) Seq = 0 Ack = 1
0.047 ACK (54841) (80) Seq = 1 Ack = 1
0.047 PSH, ACK - Len: 725 (54841) (80) Seq = 1 Ack = 1
0.097 ACK (54841) (80) Seq = 1 Ack = 726
0.100 ACK - Len: 1448 (54841) (80) Seq = 1 Ack = 726
0.100 ACK (54841) (80) Seq = 726 Ack = 1449
0.100 ACK - Len: 1448 (54841) (80) Seq = 1449 Ack = 726
0.100 ACK (54841) (80) Seq = 726 Ack = 2897
0.100 ACK - Len: 1448 (54841) (80) Seq = 2897 Ack = 726
0.100 ACK (54841) (80) Seq = 726 Ack = 4345
0.150 ACK - Len: 1448 (54841) (80) Seq = 4345 Ack = 726
0.150 ACK (54841) (80) Seq = 726 Ack = 5793
0.152 ACK - Len: 1448 (54841) (80) Seq = 5793 Ack = 726
0.152 ACK (54841) (80) Seq = 726 Ack = 7241
0.152 ACK - Len: 1448 (54841) (80) Seq = 7241 Ack = 726
0.152 ACK (54841) (80) Seq = 726 Ack = 8689
Save As
Close

TCP WINDOW MANAGEMENT

- The receiver has a flow control window
  - Size is measured in bytes
  - Size is selected by the application (if not the default)

TCP WINDOW MANAGEMENT (II)

- The sender has a congestion window
  - Size is measured in segments
  - Size is determined by the presence or absence of congestion
- Actual send window is the minimum of the flow control window (advertised by the receiver) and the (scaled) congestion window (computed by the sender)

IN-CLASS ACTIVITY

- ICA46
- TCP sequence and ack numbers