CSCI 466: Networks

Lecture 7: Transport Layer Wrap-up

Reese Pearsall

Fall 2022

Announcements

PA2 Due Wednesday October 19th

- Files must be pushed to a PA2 folder on your GitHub Repo
- **(NEW) BOTH MEMBERS MUST SUBMIT A REPO LINK TO D2L**
- Video demo with commentary is required
- GLHF

October is very busy for Reese

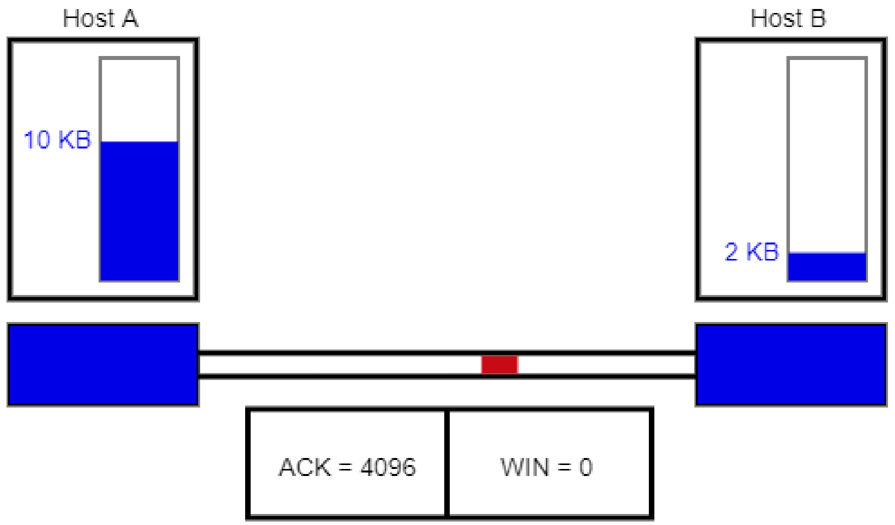
Reese is out of the country 10/13 – 10/18

- No class next Friday (10/14)→ Work day for PA2
- No class next next Monday (10/17)
- I will still be reachable by email and discord

- Virtual class on Monday 10/23

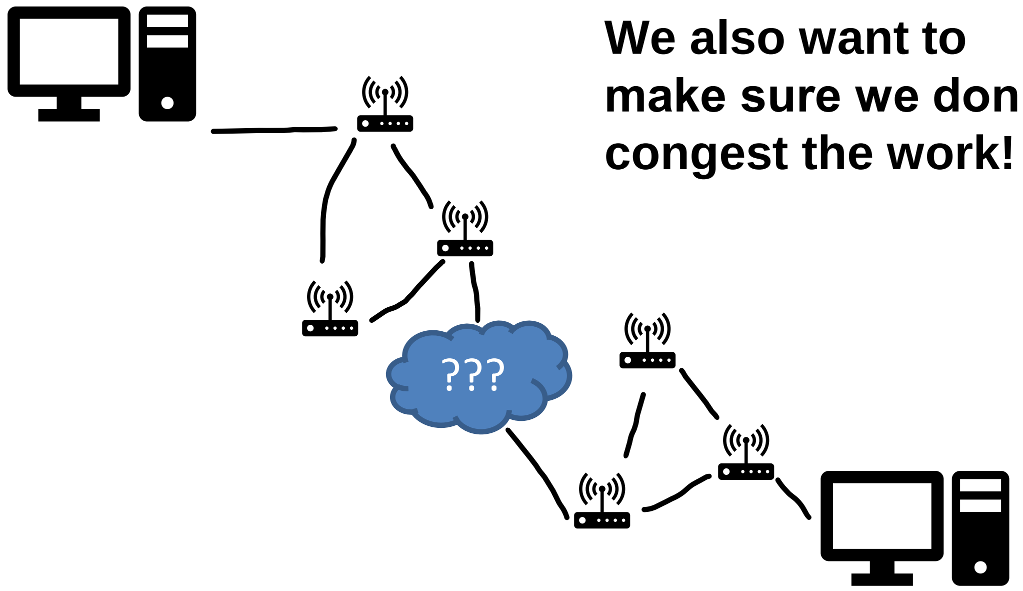
Extra Credit survey still available !!

TCP Congestion Control



TCP sends back amount of available buffer space in the receiver

This helps make sure we don't overwhelm the receiver

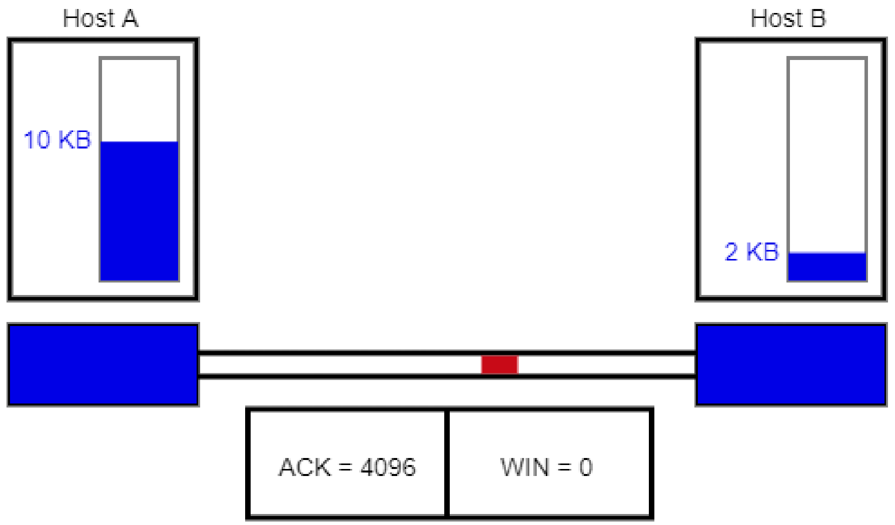


We also want to make sure we don't congest the work!

Issues:

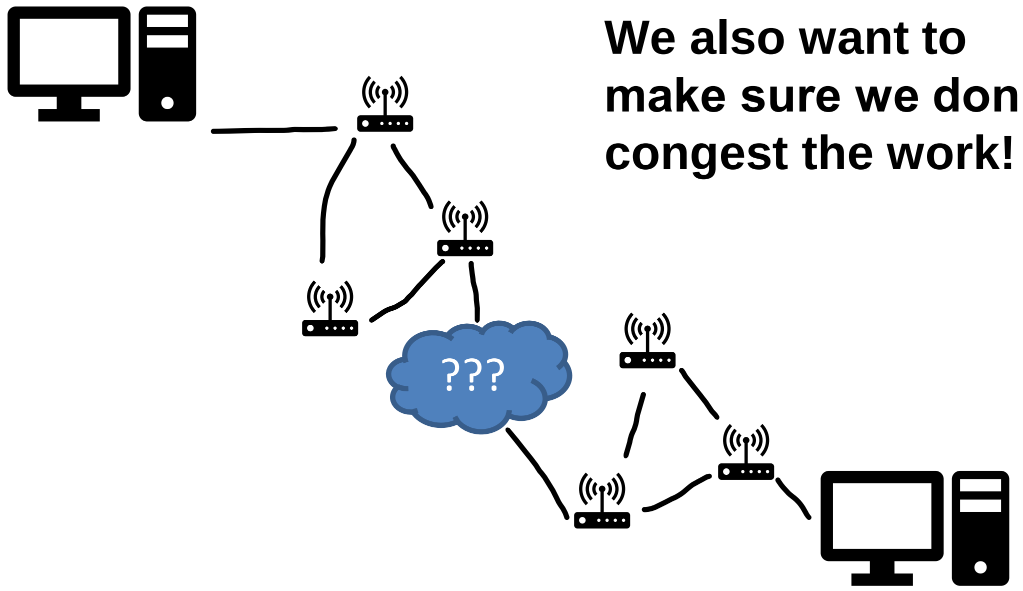
- If the network is congested, we want to slow down our sending rate
- If the network is not congested, we should try to send more stuff

TCP Congestion Control



TCP sends back amount of available buffer space in the receiver

This helps make sure we don't overwhelm the receiver



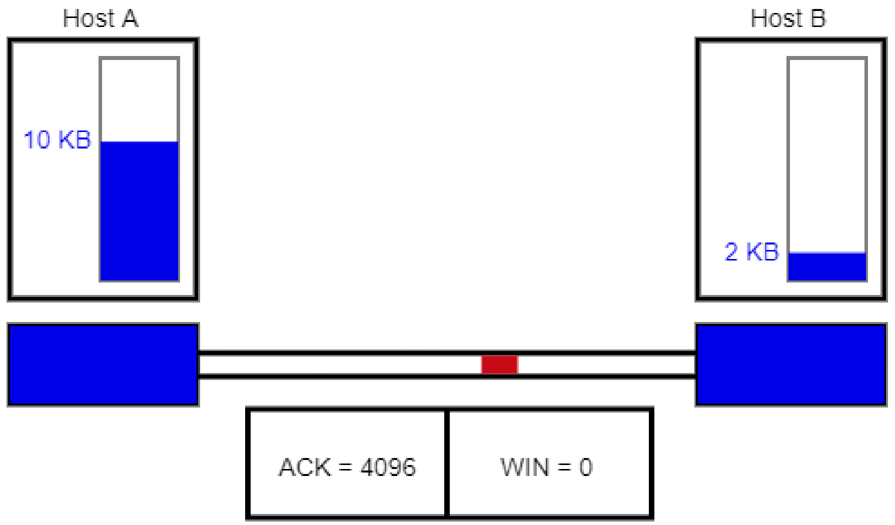
We also want to make sure we don't congest the work!

Issues:

- If the network is congested, we want to slow down our sending rate
- If the network is not congested, we should try to send more stuff

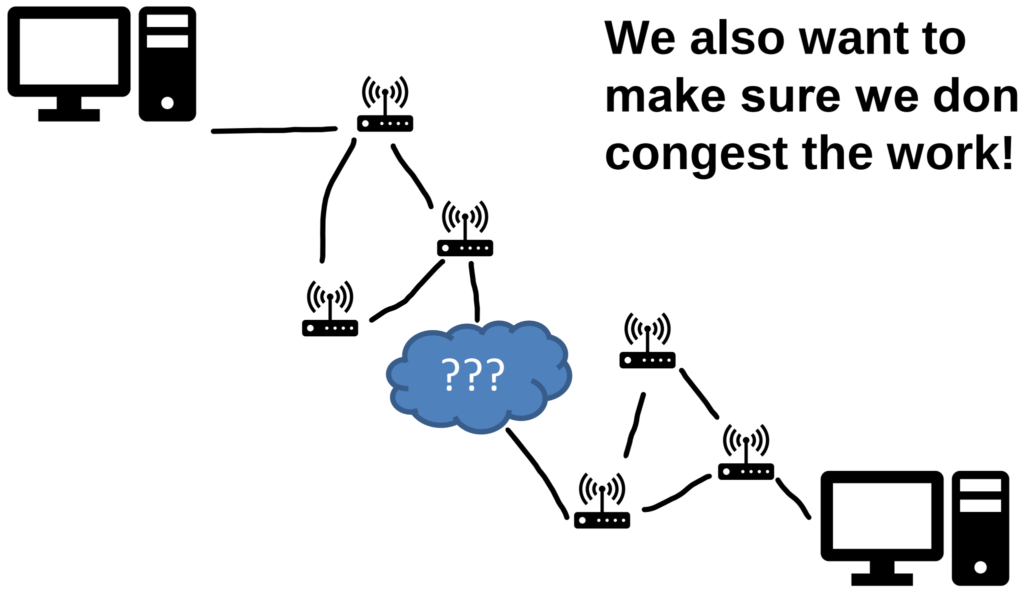
From the sender perspective, how could we measure how congested the network is?

TCP Congestion Control



TCP sends back amount of available buffer space in the receiver

This helps make sure we don't overwhelm the receiver



We also want to make sure we don't congest the work!

Issues:

- If the network is congested, we want to slow down our sending rate
- If the network is not congested, we should try to send more stuff

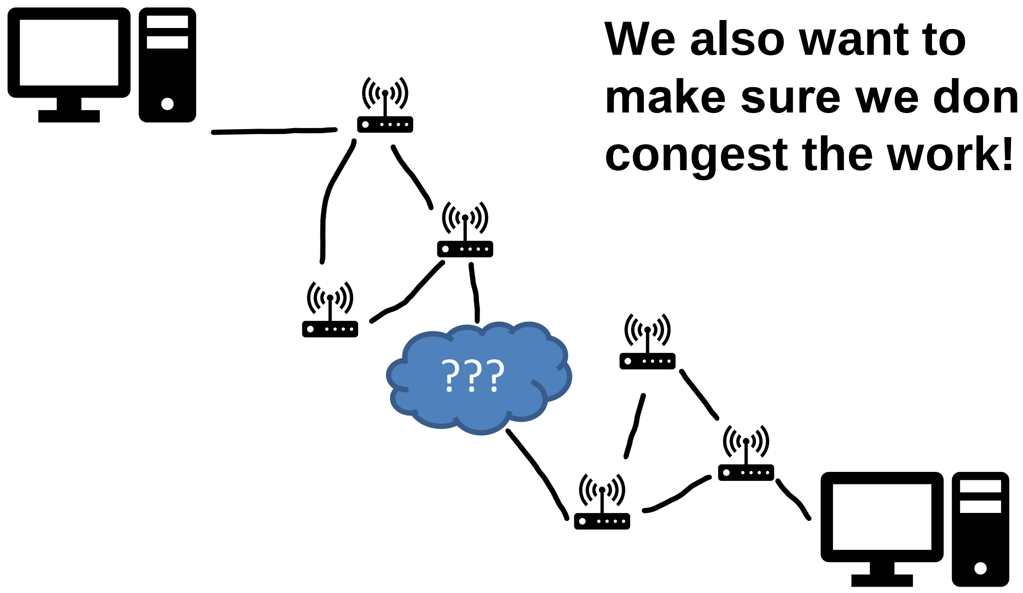
Some ways we could measure how congested the network is

-See how many dropped packets we are getting

-Amount of duplicate ACKs received

-Amount of UnAcked packets

TCP Congestion Control



We also want to make sure we don't congest the work!

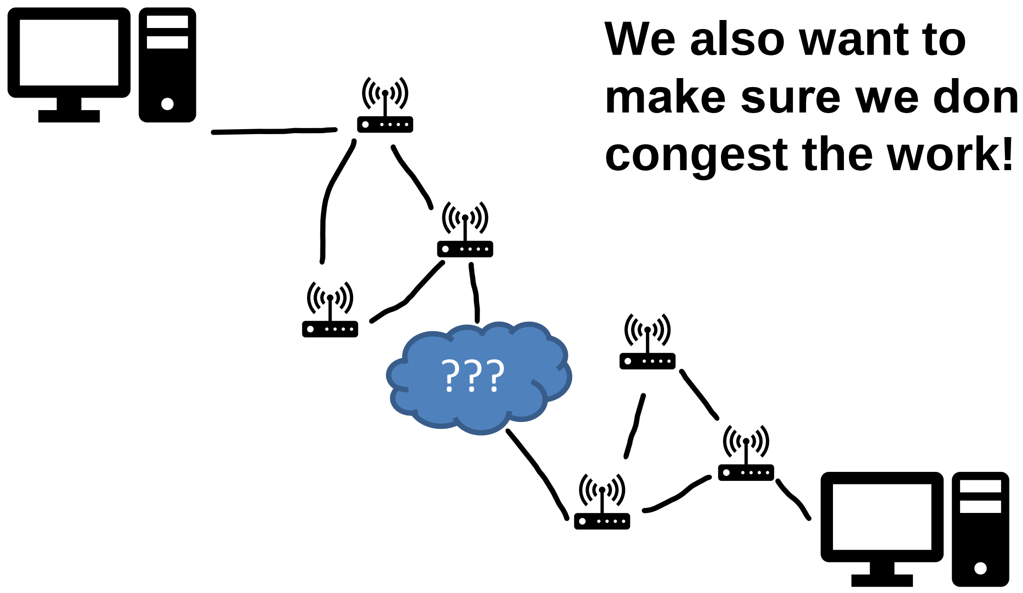
TCP is **self-clocking**

TCP sender also has a **congestion window**, which controls the amount of unAck'd that can be sent out

The amount of unacknowledged data at a sender may not exceed the *minimum* of the congestion window and receiving window

```
LastByteSent – LastByteAcked ≤ min{cwnd, rwnd}
```

TCP Congestion Control



We also want to make sure we don't congest the work!

TCP is **self-clocking**

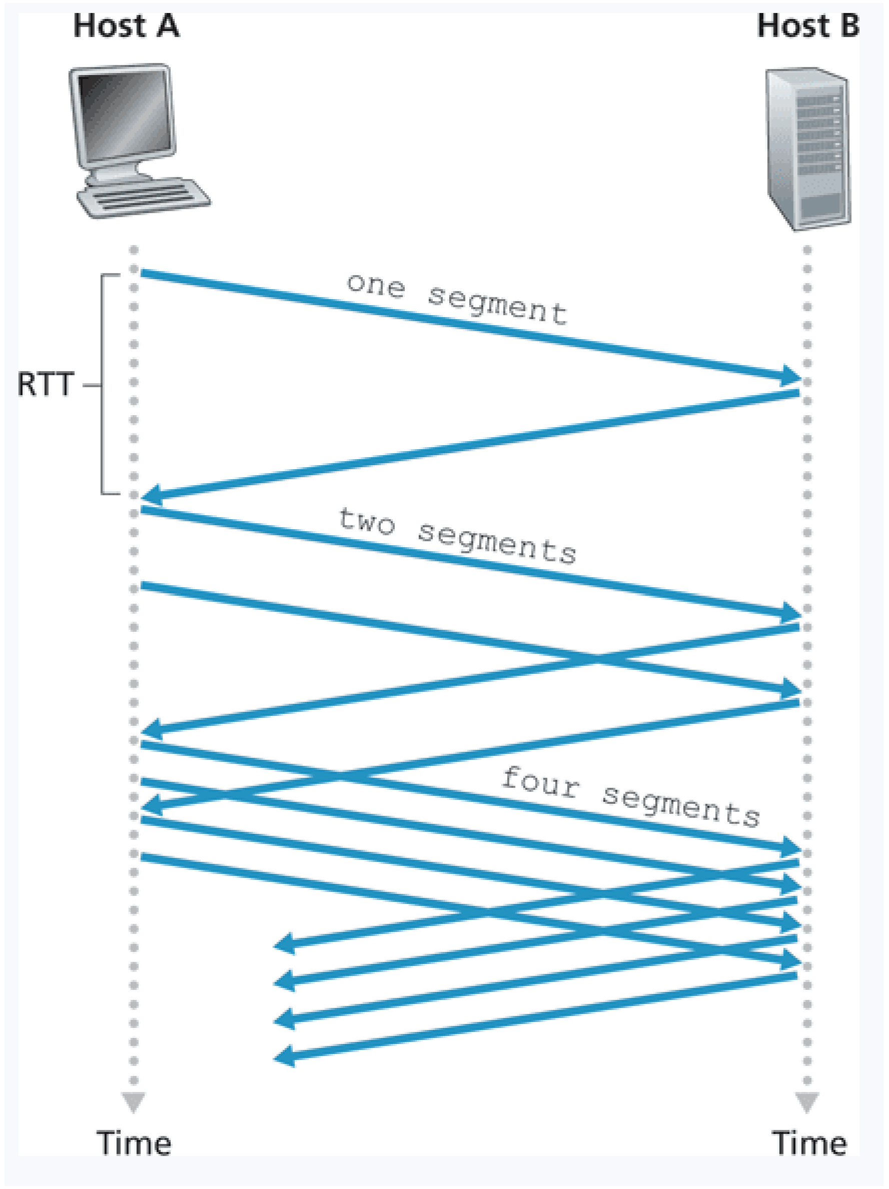
TCP sender also has a **congestion window**, which controls the amount of unAck'd that can be sent out

If TCP notices a **loss event,** it will slow down the sending rate

TCP Congestion Control Algorithm

TCP Algorithm to prevent network congestion

- **Slow Start**
- Congestion Avoidance
- Fast recovery

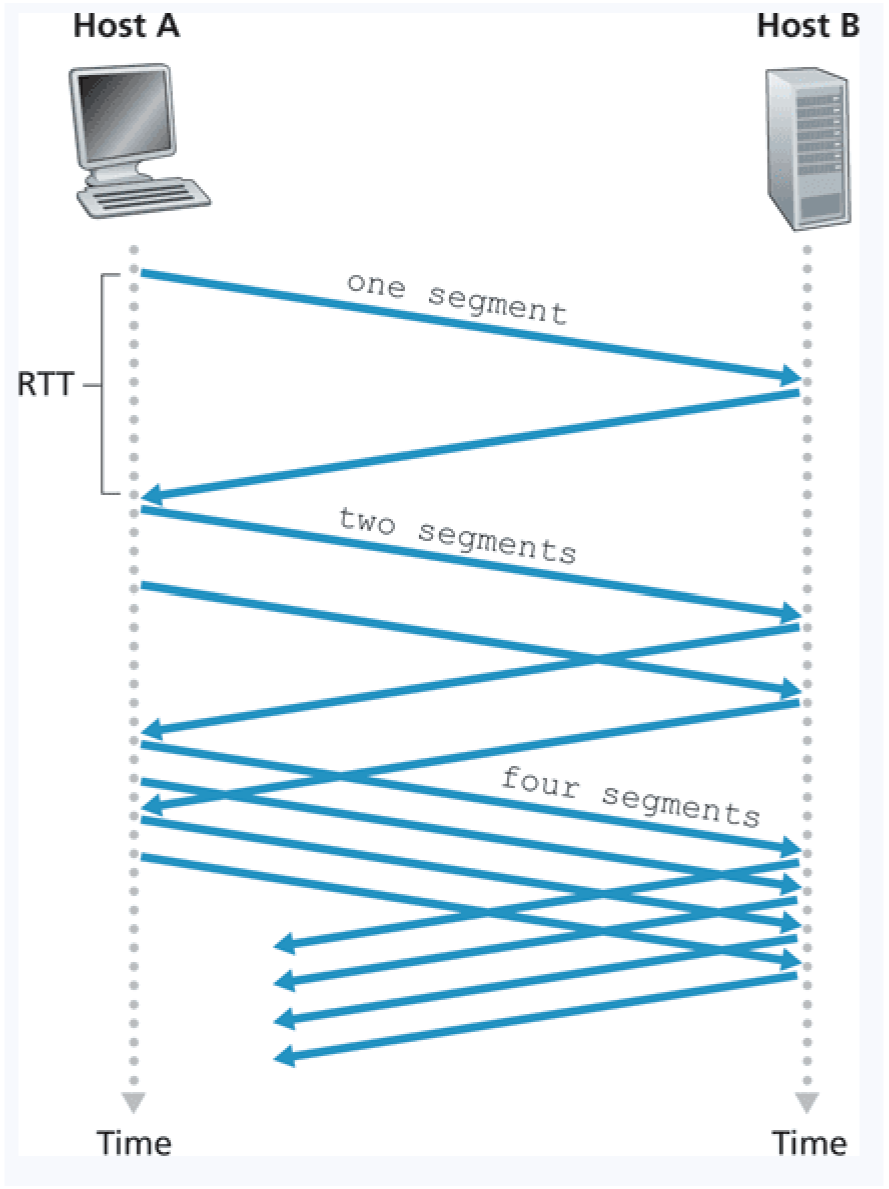


Start sending slow, but exponentially grows up to a *threshold*

TCP Congestion Control Algorithm

TCP Algorithm to prevent network congestion

- Slow Start
- **Congestion Avoidance**
- Fast recovery



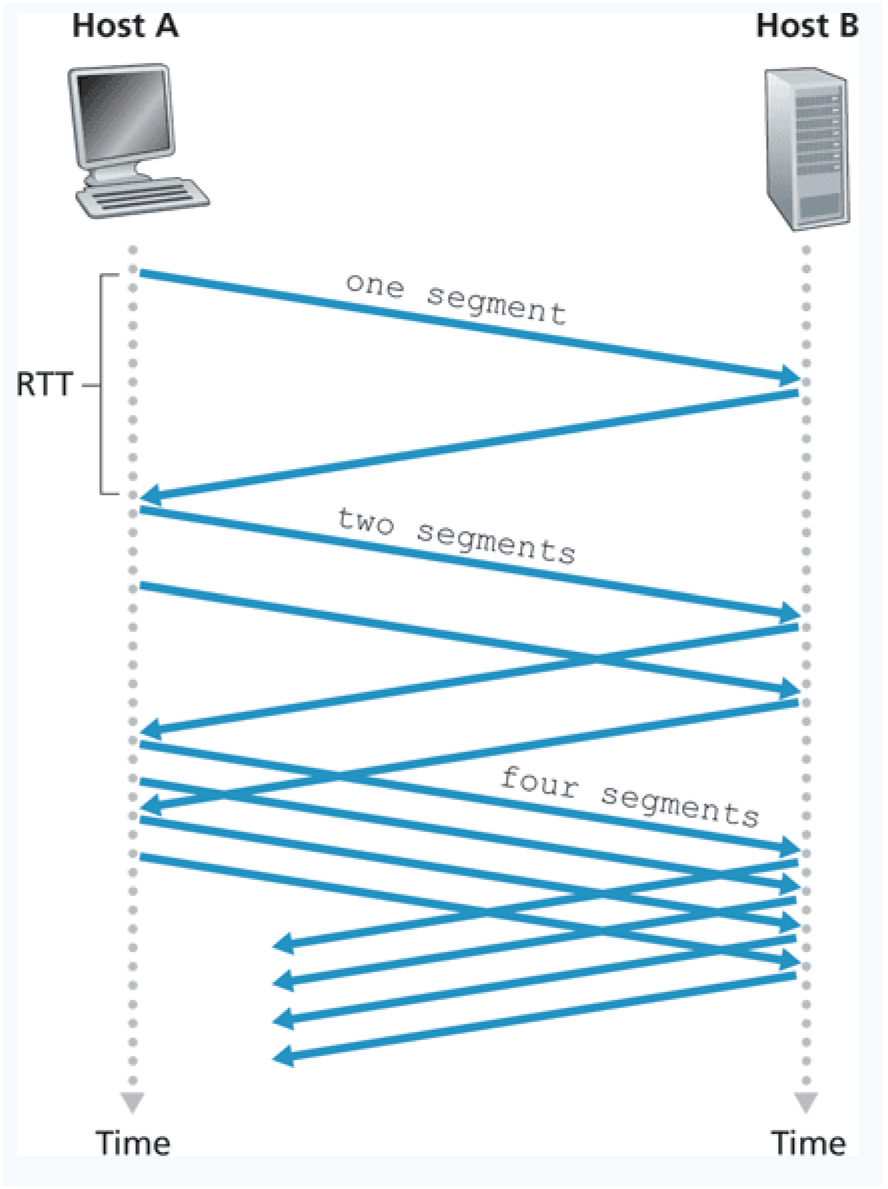
Linearly increase congestion window for each ACK received

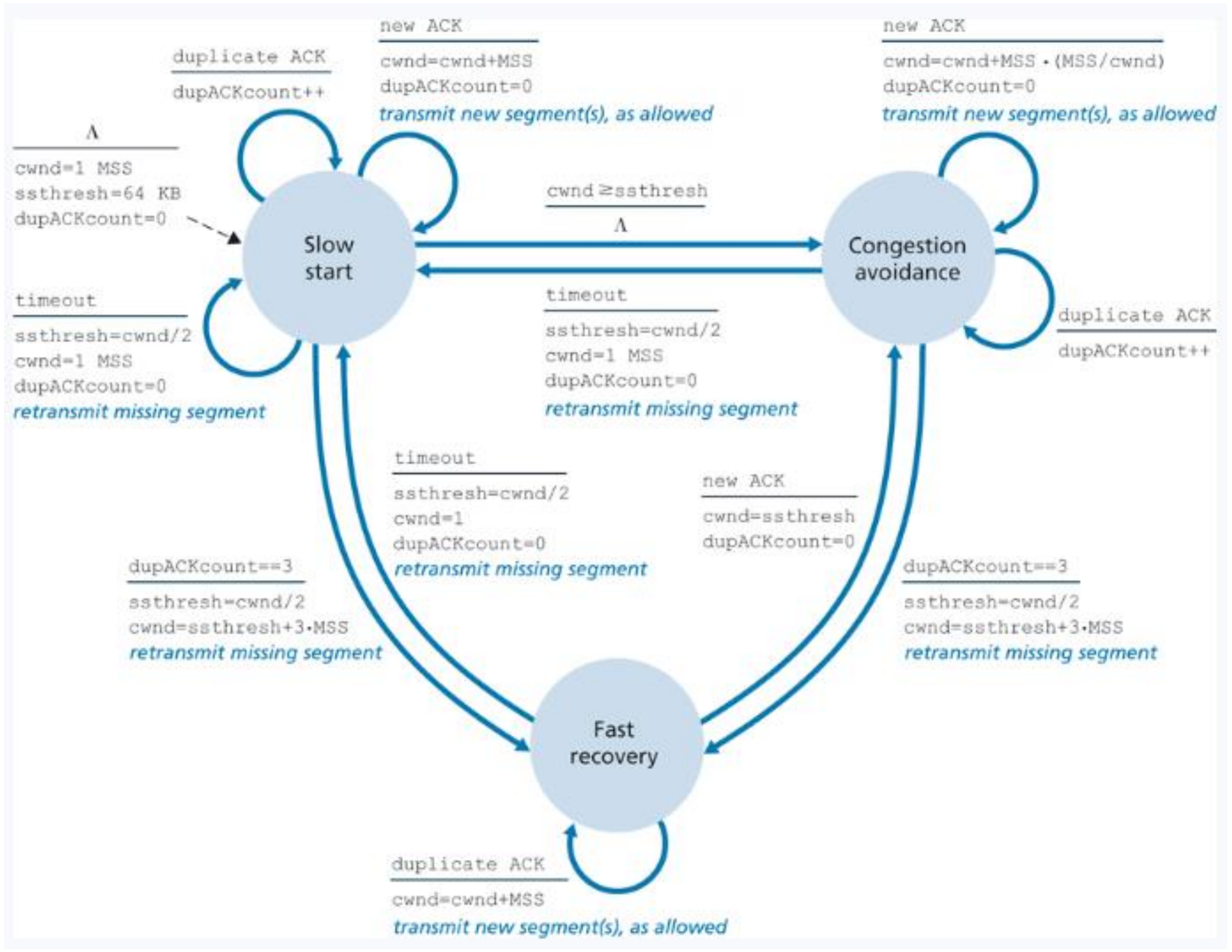
TCP Congestion Control Algorithm

TCP Algorithm to prevent network congestion

- Slow Start
- Congestion Avoidance
- **Fast recovery**

  Upon knowledge of packet loss, throttle the TCP connection and start off slow again




duplicate ACK
dupACKcount++
new ACK
cwnd=cwnd+MSS
dupACKcount=0
transmit new segment(s), as allowed
new ACK
cwnd=cwnd+MSS • (MSS/cwnd)
dupACKcount=0
transmit new segment(s), as allowed
Λ
cwnd=1 MSS
ssthresh=64 KB
dupACKcount=0
Slow start
cwnd ≥ ssthresh
Λ
Congestion avoidance
timeout
ssthresh=cwnd/2
cwnd=1 MSS
dupACKcount=0
retransmit missing segment
timeout
ssthresh=cwnd/2
cwnd=1 MSS
dupACKcount=0
retransmit missing segment
duplicate ACK
dupACKcount++
timeout
ssthresh=cwnd/2
cwnd=1
dupACKcount=0
retransmit missing segment
new ACK
cwnd=ssthresh
dupACKcount=0
dupACKcount==3
ssthresh=cwnd/2
cwnd=ssthresh+3•MSS
retransmit missing segment
dupACKcount==3
ssthresh=cwnd/2
cwnd=ssthresh+3•MSS
retransmit missing segment
Fast recovery
duplicate ACK
cwnd=cwnd+MSS
transmit new segment(s), as allowed

TCP Congestion Control Algorithm

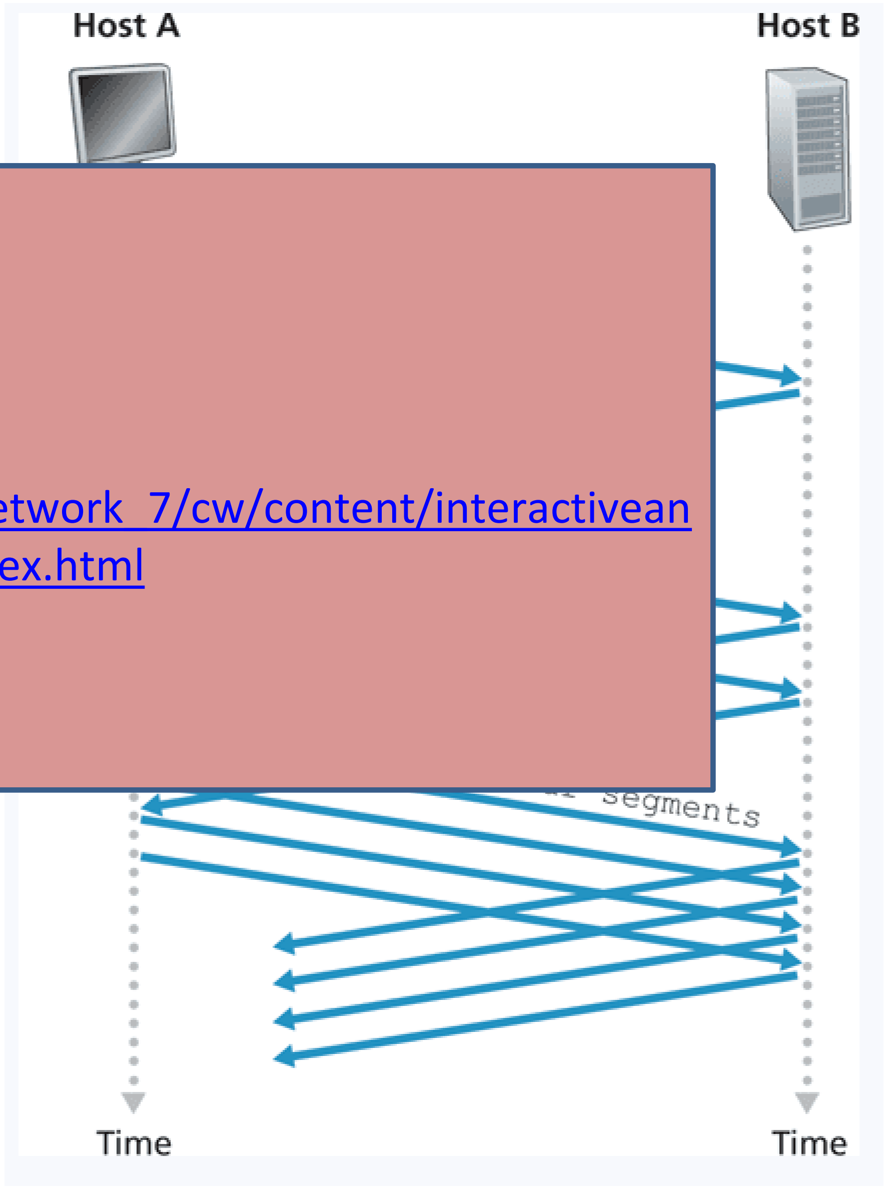
TCP Alg

- Slow
- Con
- **Fast**

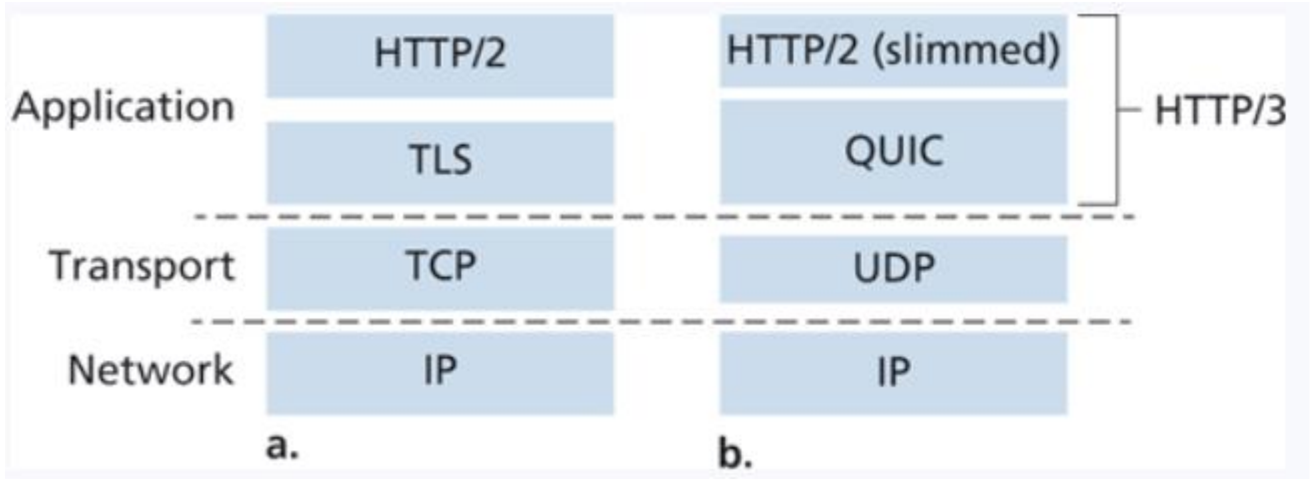
Upon knowledge of packet loss, throttle the TCP connection and start off slow again

Animation time!

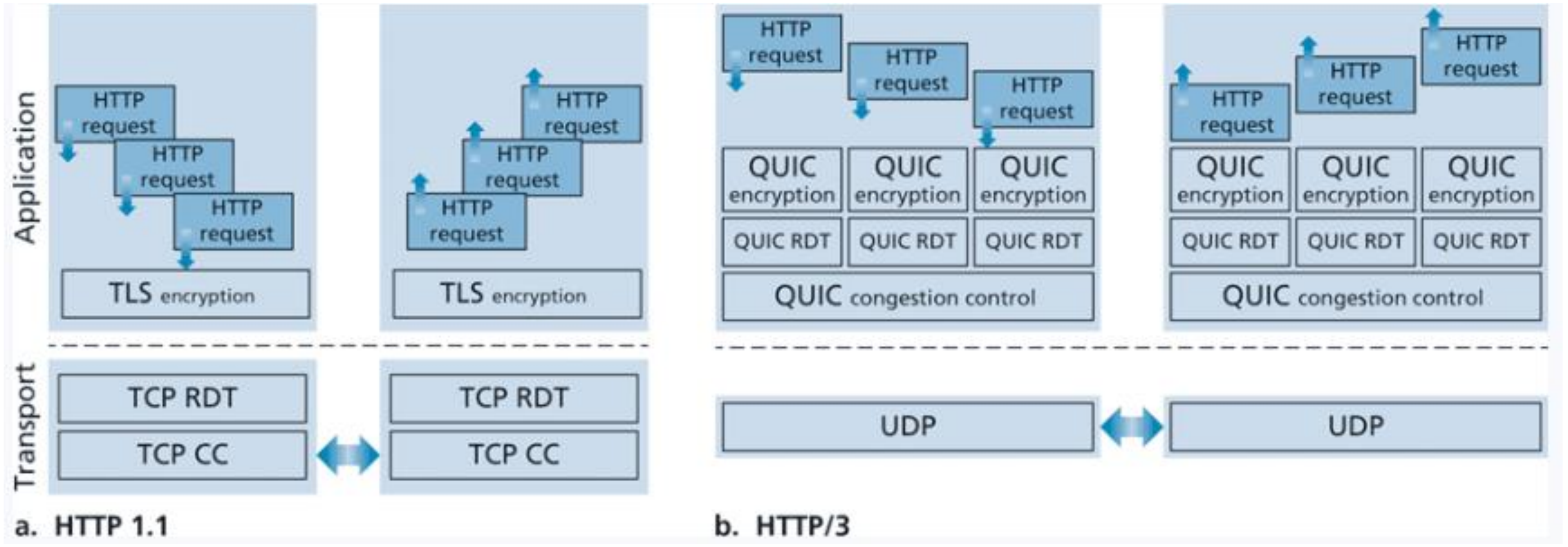
https://media.pearsoncmg.com/aw/ecs_kurose_compnetwork_7/cw/content/interactiveanimations/tcp-congestion/index.html



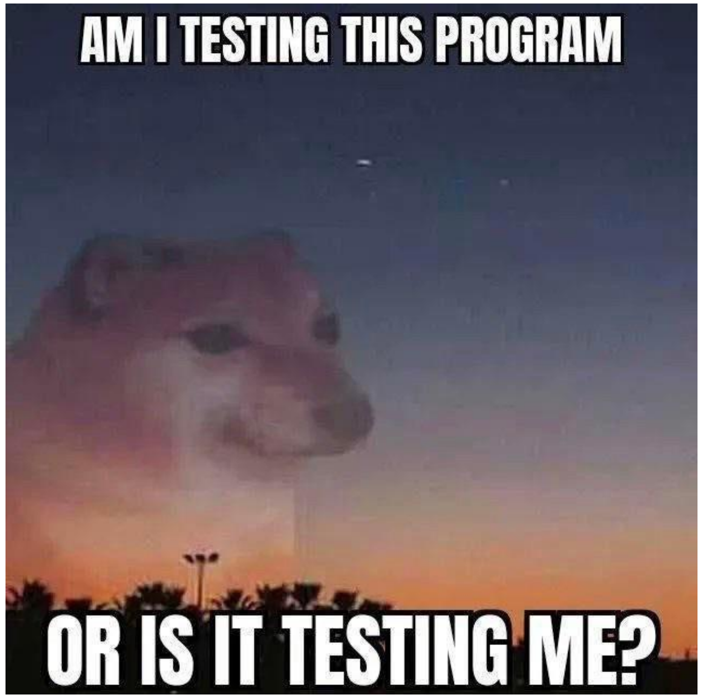
Current transport layer implementation



Transport layer protocols and congestion control is still a heavily researched area!



FIN


AM I TESTING THIS PROGRAM
OR IS IT TESTING ME?

UDP
second packet sent
but the first one is missing
UDP
fifth packet sent
and where are the others?
UDP
last packet sent