CMSC 417 Spring 2016 Lecture #15 3/30/2016

Agenda:

⇒TCP
- flow control
- retransmission
- congestion control/avoidance

CMSC417 Spring 2016 Lecture #15 3/30/2016

TCP Flow Control

⇒ flow control is a way to allow a receiver to limit how fast a sender sends to them

⇒ receiver sends an "advertised window", basically the buffer space in bytes they have avail.
- sender will send no more unacked bytes than this

⇒ advertised window is 16-bits, TCP seq #s are 32-bits ⇒ $SWS \leq RWS \ll |\text{seq \#s}|$

⇓

safe from accepting "old" packets when send/delivery is in the same order

⇒ if advertised window is 0, sender can't send and thus can't get an ack to hear about the window opening
- send 1-byte segments periodically

⇒ 16 bit window size allows for $2^{16}-1$ bytes per RTT
- faster links make this insufficient
- 10 Gbps links can send 2^{16} bytes in < 53 ms
- if your RTT is greater, you can't fill the link

⇒ window scaling option
- lets you bit shift your advertised window by 0 to 14 bits ⇒ max window size $= (2^{16}-1) \times 2^{14}$
  $= 1,073,725,440$
  $\approx$ 1 Gigabyte
- allows TCP to fill an 85 ms RTT 100 Gbps link

CMSC417 Spring 2016 Lecture #15 3/30/2016

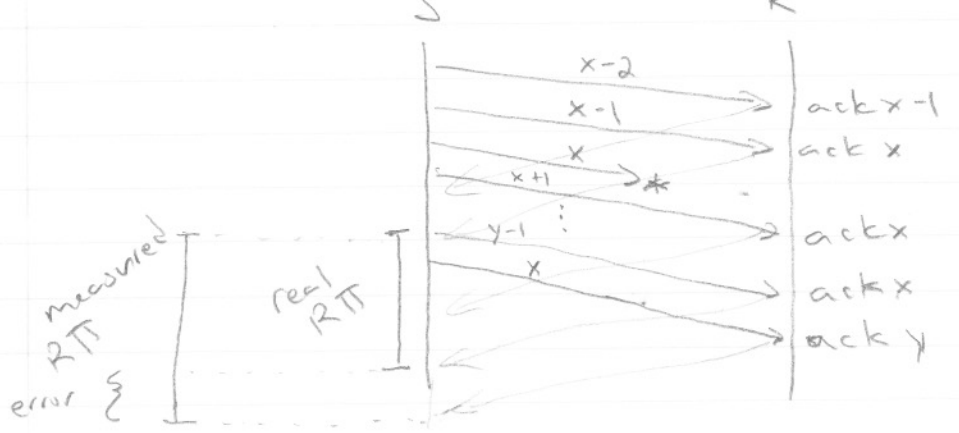
TCP Retransmission

⇒ when should you retransmit an unacked packet?
- when you hit a timeout? how long?
- when else?

⇒ TCP keeps an RTT estimate
- on every ack
  estRTT = α * estRTT + (1-α) * (RTT for this ack)
- α ~ (0.8, 0.9)

⇒ assume a packet was lost when not acked for 2x estRTT

⇒ how to avoid "weird" RTT estimates?
- what happens when a packet is lost?
  ack x, ack x, ack x, ...
- avoid this by only interpreting the first ack for x as an RTT est
- still other issues around things like
  - first ack for y being caused by a retransmit for x, e.g.,



CMSC 417 Spring 2016 Lecture #15 3/30/2016

TCP retransmission cont'd

⇒ on RTO, set est RTT = 2 × est RTT
- why?

(retransmit timeout)

⇒ also account for variance (β in RFC 793)

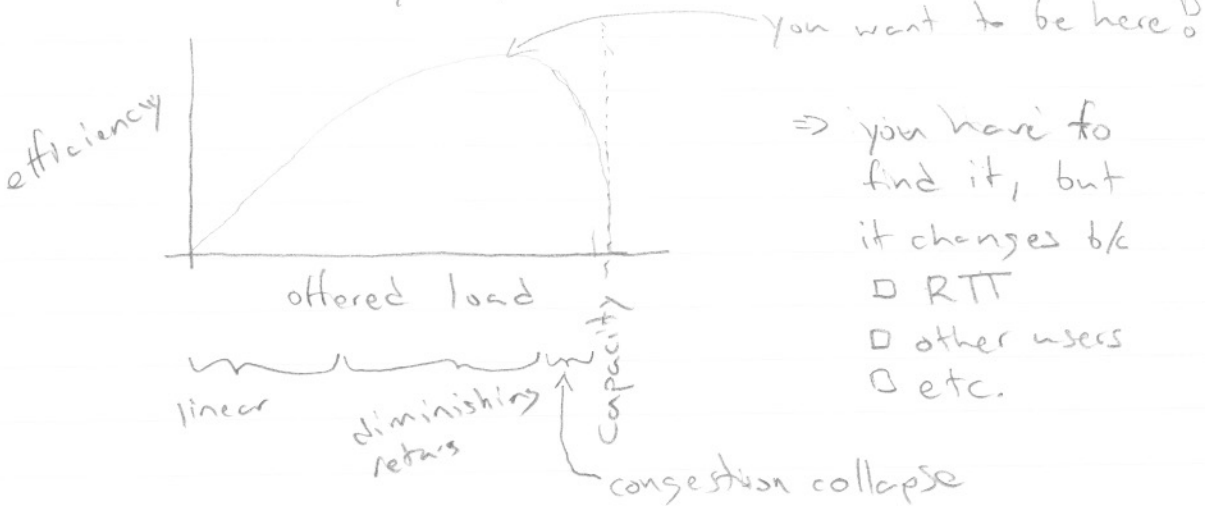
Fast Retransmit

⇒ TCP Tahoe
- many TCP variants/enhancements named after cities in Nevada

⇒ on seeing 3 duplicate acks assume the packet was lost and retransmit it
- resends quicker than a timeout
- useful in the (common) case where losses are infrequent

congestion control/avoidance

information theory says: (for a given "channel")



⇒ you have to find it, but it changes b/c
- RTT
- other users
- etc.

CMSC 417 Spring 2016 Lecture #15 3/30/2016

Congestion Collapse

⇒ once congestion gets to a certain point, efficiency goes off a cliff
- congestion ⇒ loss
- loss ⇒ retransmission
- retransmission ⇒ more data
- more data ⇒ more congestion

⇒ in 1986 the NSFNet (Internet) backbone went from 32 Kbps to 40 bps for this reason
- ~1000x loss in efficiency

Key lesson

⇒ there is a fixed load the network can take at a given point in time
⇒ when handling loss/retransmission, don't violate ~~that~~ load limit

Problem

⇒ have to estimate that load over time
- despite other people's data
- despite varying RTT
- despite loss rate
- ...

solution

⇒ dynamically adjust the sender's window size — called congestion window or cwnd
⇒ ensure that min(cwnd, rwnd) bytes are ever in flight