TCP Congestion Control

Network Programming

Overview

- TCP (Transmission Control Protocol) provides reliability
  - In-order
  - Connection-oriented

- Also provides two important items:
  - Flow Control
  - Congestion Control

TCP Segment Format

https://tools.ietf.org/html/rfc793#section-3.1

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          Source Port          |       Destination Port        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Sequence Number                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Acknowledgment Number                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Data |           |U|A|P|R|S|F|                               |
| Offset| Reserved  |R|C|S|S|Y|I|            Window             |
|       |           |G|K|H|T|N|N|                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           Checksum            |         Urgent Pointer        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Options                    |    Padding    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                             data                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

                        TCP Header Format

      Note that one tick mark represents one bit position.
```

Flow Control

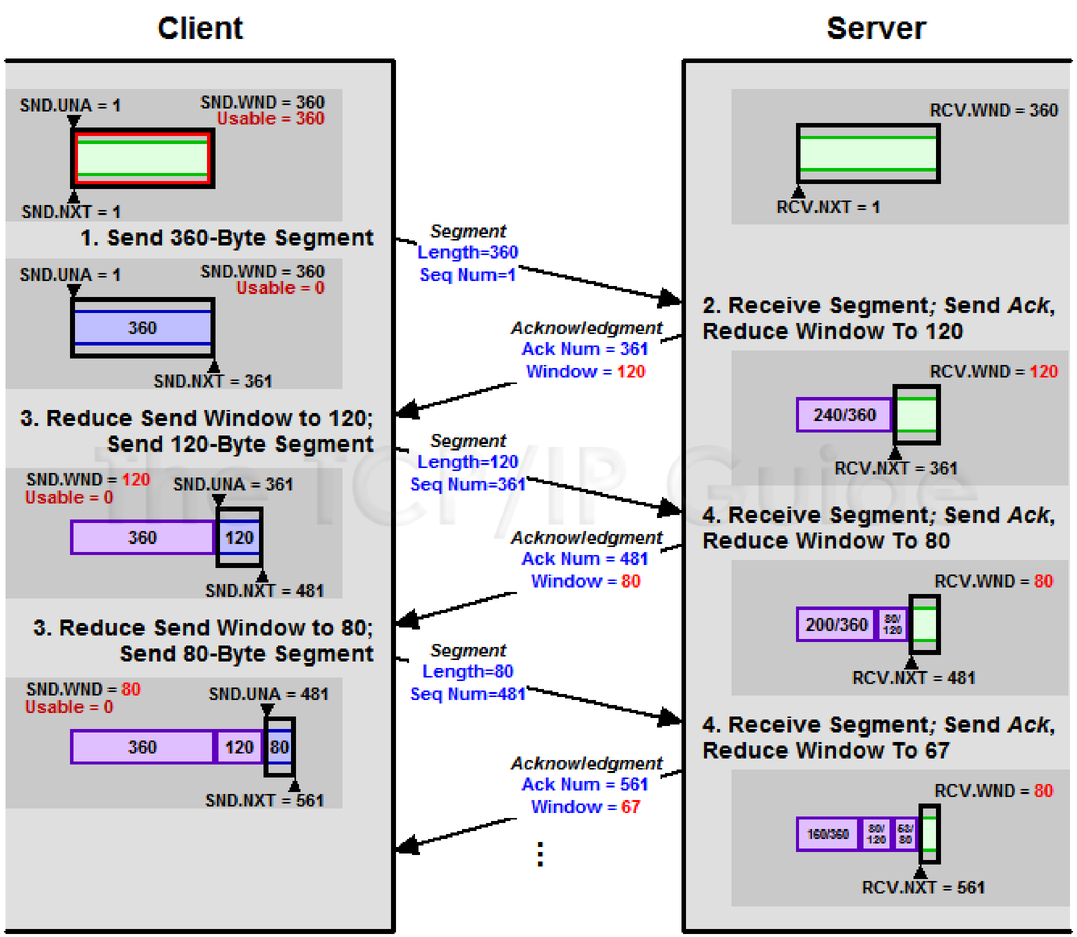
- Endpoints today are fast!
- Can easily saturate most connections
  - In turn, can easily saturate receiving hosts
- Requires a mechanism to tell sending host to slow down

Sliding Window

- Allows receiver to send feedback to sender
- Communicates *receive window* for bytes it's willing to accept
  - If 0, can't send at all!
- Sender cannot send more than advertised window space

Silly Window Syndrome

- Slow consumer / producer problem



Nagle's Algorithm

- With small interactive sessions, small (1 byte?) data transmissions
- With a 40 byte TCP header, huge overheads!
- Combine these small messages
  - Send all at once, achieve greater efficiency

Congestion Control

- Congestion leads to packet loss
  - Increases retransmissions
- TCP generally follows pattern of Additive Increase, Multiplicative Decrease (AIMD)
  - Basically, accelerate slowly but brake hard
- Force sender to slow down

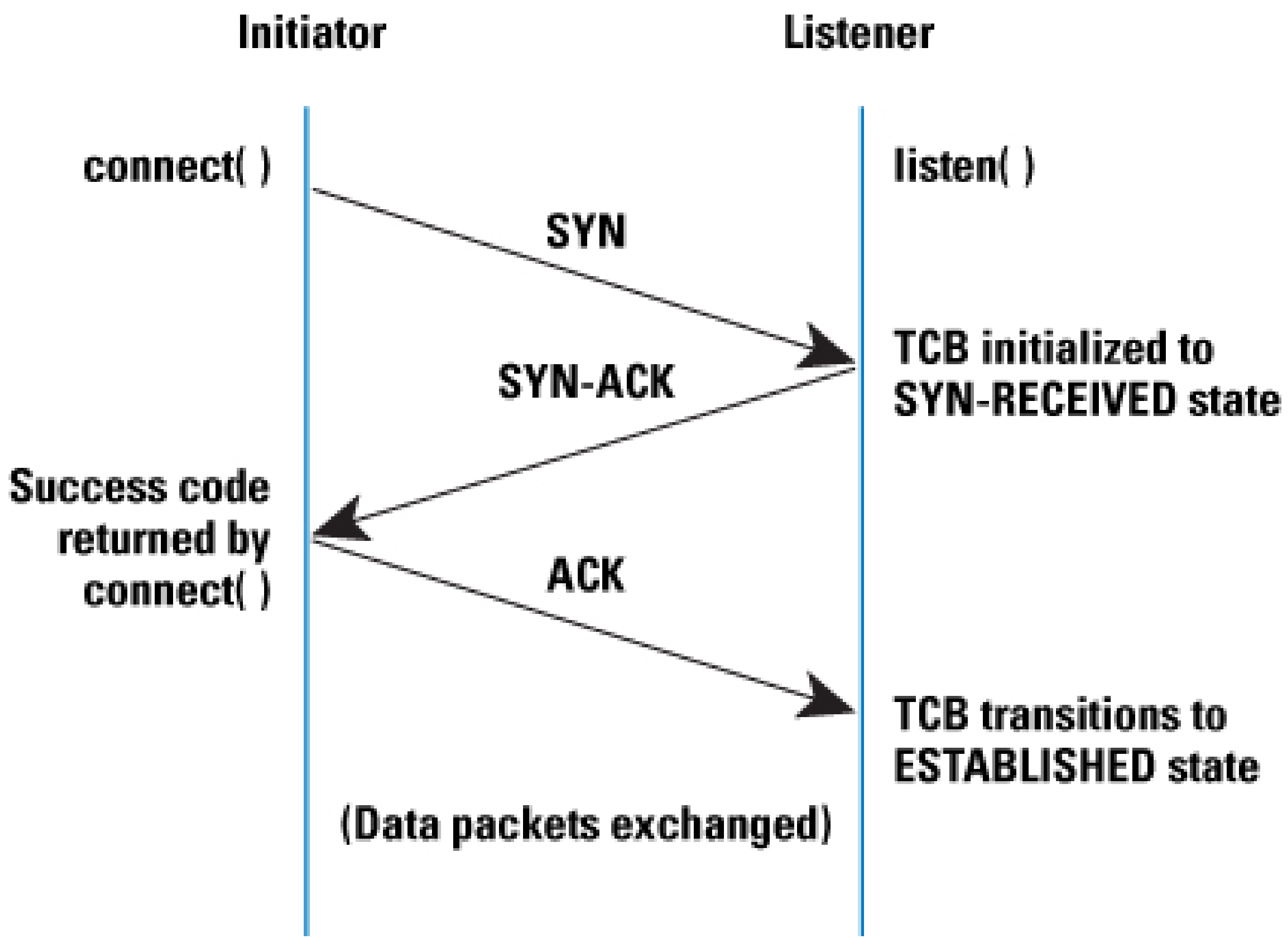
Congestion Collapse

- If left unchecked, could destroy useful bandwidth of network
- In 1986, connection between LBL and UC Berkeley (400 yards) dropped to 40 bps due to this problem
  - Factor of ~1000 drop

CC Countermeasures

- https://tools.ietf.org/html/rfc5681
- Slow Start
- Congestion Avoidance
- Fast Retransmit
- Fast Recovery

Three Way Handshake



Slow Start

- Initialize congestion window to one segment (MSS)
  - Established during 3WHS
- On each ACK increase cwnd
  - cwnd $\leftarrow$ cwnd + MSS
- Effect? Exponential growth of cwnd during each RTT
  - Named for initial size, not slow!

Congestion Avoidance

- Slow Start initially used
  - But packet gets dropped
  - Or slow start threshold (ssthresh) exceeded
- Network congestion is happening
  - Reduce to linear increase
- On each ACK increase cwnd
  - cwnd $\leftarrow$ cwnd + MSS^2/cwnd

Fast Retransmit

- Duplicate ACK received. Why?
  - Lost? Delayed?
- If three or more duplicate ACKs are received, immediately retransmit last segment
  - Bypass timers

Fast Recovery

- Duplicate ACKs can only be generated when a segment is received
  - Network may not be as bad as imagined

- Enter Congestion Avoidance mode instead of resetting window size by jumping into Slow Start

RED / ECN

- Random Early Discard takes place at burdened routers
  - As burden goes up, so does likelihood of RED happening
  - Triggers multiplicative decrease

- Explicit Congestion Notification
  - RED may be too extreme
  - Routers toggle bit in header, receiver echos it, sender slows transmission

TCP Congestion Avoidance Flavors

Network Programming

Slightly Deeper Look…

- We just discussed TCP congestion control
- One of the big things TCP provides is Flow Control
- We said “AIMD” and left it at roughly that.
  - Is it that simple?

Evolving Options

- The basic principle is the same:
  - Get to about the right amount of traffic
  - React to congestion appropriately
  - Increase traffic
- The overall goal is to minimize
  - Underutilization of bandwidth
  - Flooding of hosts / internetwork (i.e. dropped packets)

Wikipedia To the Rescue

- Great table

TCP Tahoe

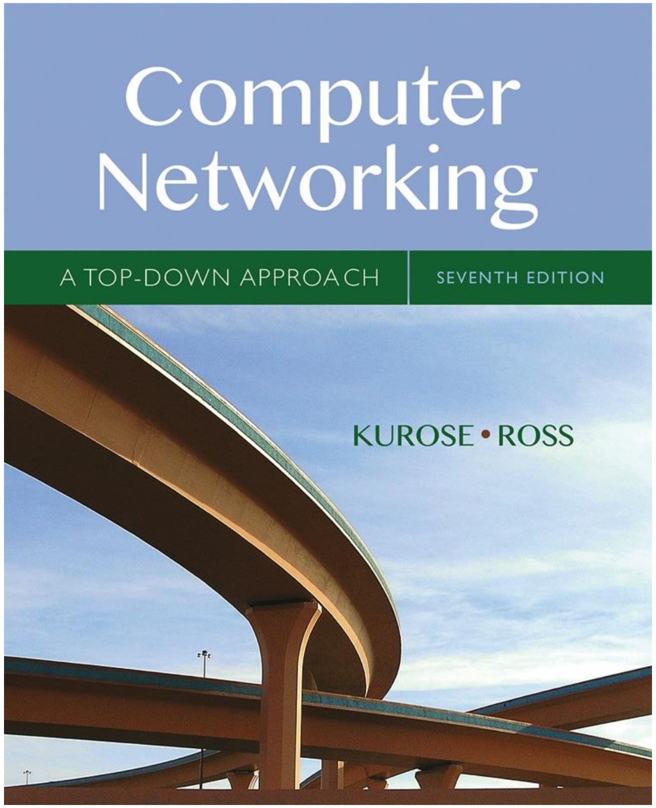
- If 3 duplicate ACKS are received
  - Things are really bad!
- Fast Retransmit
- ssthresh = cwnd/2
- cwnd = 1 MSS
- Slow Start

TCP Reno

- If 3 duplicate ACKS are received
  - Things are really bad! (Again)
- Fast Retransmit
- tcp.cwnd *= 0.5
- ssthresh = cwnd
- Fast Recovery

Next 2 slides are courtesy of…

Chapter 3
Transport Layer



Computer Networking: A Top Down Approach

7th edition
Jim Kurose, Keith Ross
Pearson/Addison Wesley
April 2016

A note on the use of these Powerpoint slides:

We're making these slides freely available to all (faculty, students, readers). They're in PowerPoint form so you see the animations; and can add, modify, and delete slides (including this one) and slide content to suit your needs. They obviously represent a *lot* of work on our part. In return for use, we only ask the following:

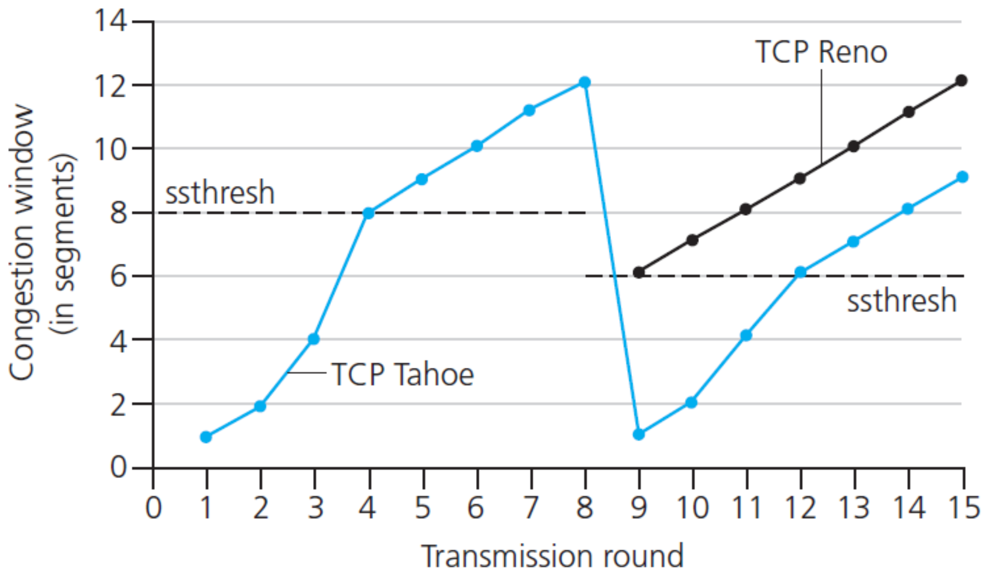
- If you use these slides (e.g., in a class) that you mention their source (after all, we'd like people to use our book!)
- If you post any slides on a www site, that you note that they are adapted from (or perhaps identical to) our slides, and note our copyright of this material.

Thanks and enjoy! JFK/KWR

TCP: switching from slow start to CA

Q: when should the exponential increase switch to linear?

A: when **cwnd** gets to 1/2 of its value before timeout.

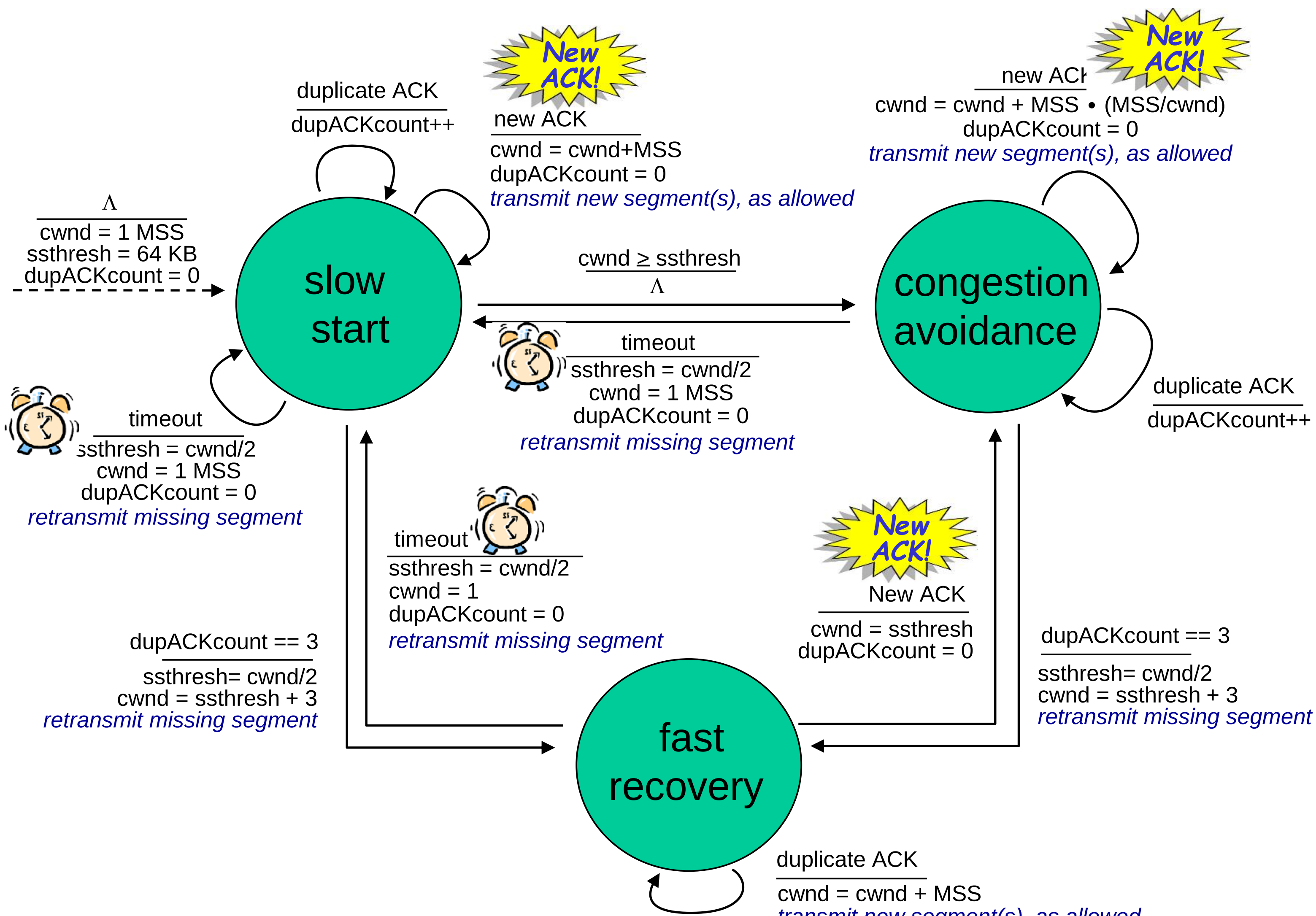


Implementation:

- variable **ssthresh**
- on loss event, **ssthresh** is set to 1/2 of **cwnd** just before loss event

* Check out the online interactive exercises for more examples: http://gaia.cs.umass.edu/kurose_ross/interactive/

Summary: TCP Congestion Control



Some More…

- TCP Vegas
  - Use RTT changes to detect congestion
  - Not very widespread
- TCP Westwood
  - Like Reno but use bandwidth estimation instead of just cutting the window
  - EWMA, other “low pass” solutions?
- TCP New Reno
  - During Fast Recovery
  - Any duplicate ACK yields a new packet

TCP BIC/CUBIC

- Used in Linux Kernel
  - BIC 2.6.8 – 2.6.18
  - CUBIC 2.6.19+
- BIC is built for LFNs
- CUBIC uses a cubic “smoothing”
  - Like BIC but doesn’t grow as fast