ICS 451: Today's plan

- TCP Congestion Control
- Network Layer Protocols
- Packet Forwarding

TCP Congestion Control

- Increase the congestion window regularly
  - the increase is linear: 1MSS/RTT
- Decrease the congestion window once congestion is detected
  - the decrease is multiplicative: reduce to half
- So TCP congestion control is AIMD: Additive Increase and Multiplicative Decrease
  - AIMD is important for network stability

cwnd Transient Behavior

- cwnd is 1 MSS at start of connection
- cwnd grows by 1MSS for every MSS acked
  - this exponential growth is called *slow start*
- When congestion detected, cwnd set to 1 MSS
- then grows exponentially up to half the previous window
  - half the previous window is called the slow start threshold or *ssthresh*

cwnd Linear Growth

- after completion of slow start, cwnd grows by 1MSS every RTT
- this is done by adding to the window, on every new ack,

$$(MSS \times new\text{-}bytes\text{-}acked) / cwnd$$

Congestion Response

- On a fast retransmit, cwnd = cwnd / 2, and cwnd resumes linear growth
- On timer expiration,
  - ssthresh = cwnd / 2,
  - cwnd = 1MSS
  - slow start up to ssthresh, then linear growth

Performance

- Assuming constant available bandwidth, TCP:
- finds it quickly through slow start, then repeatedly:
- drops to half and increases linearly again to the full bandwidth
- so on average, TCP only uses ¾ of the available bandwidth

- Connections with shorter RTT increase faster

Summary of TCP congestion control

- packet loss means congestion
  - safe assumption, not always accurate
- use cwnd on sender to control speed
- on congestion, slow down multiplicatively
  - cut the window to half
- most of the time, grow linearly
- all this is known as TCP Reno

- fairness is impossible, but worth trying for

Alternatives

- Since queue lengths increase as congestion begins, RTTs also increase
- measuring the RTT (or related measures) allows early detection of congestion
  - unfortunately RTT measurements are noisy
- area for research!
- UDP has no congestion control
- Other Internet transport protocols are supposed to implement “TCP-like” congestion control

Network Layer

- destinations identified by a Network Layer address
  - e.g. IPv4 address, IPv6 address
- independent of underlying layers
- data plane: protocols and mechanisms to communicate data
- control plane: protocols and mechanisms to build routing tables
- connectionless or connection-oriented service
  - IP provides connectionless service

Data-Link Layer Types

- Point-to-Point, 2 systems
  - e.g. modems over a telephone line
  - may use Point-to-Point Protocol (PPP)
- Broadcast-based, many systems
  - Ethernet (802.3)
  - WiFi (802.11)
  - MAC addresses identify endpoints
  - support unicast, multicast, and broadcast
- Non-broadcast Many Access (NBMA)
  - address identifies unicast destination

Network Layer Implementation

- uses Data-Link layer to send frames
  - frames may be sent best-effort
    - e.g. over fiber optics or ethernet
  - or with acknowledgment and retransmission
    - e.g. WiFi
- each frame has a fixed maximum size

Network Layer Forwarding

- hop-by-hop forwarding
- every system (hosts and routers) has a *routing table* that lists
  - every destination
  - the interface over which the destination can be reached (and the address of the *next hop*)
- packet headers include the destination address
- packets are forwarded by
  - looking up the destination in the routing table
  - forwarding to the next hop through the interface

Forwarding Example

- book, p. 130, Fig 5.4

Forwarding Constraints

- The routing tables must be consistent:
  - if Alice reaches Bob through Charlie
  - then Charlie must have a route to Bob that does not go through Alice
  - otherwise, there is a *routing loop*
  - and packet to Bob will never be delivered
- The next-hop IP address is used to obtain the MAC address of the next hop
  - this MAC address is used to send the packet through the data-link layer