Internet Protocols
Fall 2006

Lecture 16
TCP Flavors, RED, ECN
Andreas Terzis
Outline

• TCP congestion control
  - Quick Review
  - TCP flavors
  - Impact of losses
  - Cheating

• Router-based support
  - RED
  - ECN
Quick Review

• Slow-Start: cwnd++ upon every new ACK
• Congestion avoidance: AIMD if cwnd > ssthresh
  - ACK: cwnd = cwnd + 1/cwnd
  - Drop: ssthresh =cwnd/2 and cwnd=1
• Fast Recovery:
  - duplicate ACKS: cwnd=cwnd/2
  - Timeout: cwnd=1
TCP Flavors

- **TCP-Tahoe**
  - $cwnd = 1$ whenever drop is detected

- **TCP-Reno**
  - $cwnd = 1$ on timeout
  - $cwnd = cwnd/2$ on dupack

- **TCP-SACK**
TCP-SACK

- SACK = Selective Acknowledgements

- ACK packets identify exactly which packets have arrived

- Makes recovery from multiple losses much easier
Standards?

• How can all these algorithms coexist?

• Don’t we need a single, uniform standard?

• What happens if I’m using Reno and you are using Tahoe, and we try to communicate?
Cheating

• Three main ways to cheat:
  – increasing cwnd faster than 1 per RTT
  – using large initial cwnd
  – Opening many connections
Increasing cwnd Faster

x increases by 2 per RTT
y increases by 1 per RTT

Limit rates:
x = 2y
Increasing cwnd Faster

- A
- D
- B
- E

Graph showing oscillations over time.
Larger Initial cwnd

x starts SS with cwnd = 4
y starts SS with cwnd = 1
Open Many Connections

Assume
• A starts 10 connections to B
• D starts 1 connection to E
• Each connection gets about the same throughput

Then A gets 10 times more throughput than D
Cheating and Game Theory

Individual incentives: cheating pays
Social incentives: better off without cheating

Too aggressive → Losses
Too aggressive → Throughput falls
Lossy Links

• TCP assumes that all losses are due to congestion

• What happens when the link is lossy?

• Tput $\sim \frac{1}{\sqrt{p}}$ where $p$ is loss prob.

• This applies even for non-congestion losses
Example
What can routers do to help?
Paradox

- Routers are in middle of action
- But traditional routers are very passive in terms of congestion control
  - FIFO
  - Drop-tail
FIFO: First-In First-Out

- Maintain a queue to store all packets
- Send packet at the head of the queue
Tail-drop Buffer Management

- Drop packets only when buffer is full
- Drop arriving packet
Ways Routers Can Help

- Packet scheduling: non-FIFO scheduling

- Packet dropping:
  - not drop-tail
  - not only when buffer is full

- Congestion signaling
Question!

• Why not use infinite buffers?
  – no packet drops!
The Buffer Size Quandary

• Small buffers:
  - often drop packets due to bursts
  - but have small delays

• Large buffers:
  - reduce number of packet drops (due to bursts)
  - but increase delays

• Can we have the best of both worlds?
Random Early Detection (RED)

• Basic premise:
  - router should signal congestion when the queue first starts building up (by dropping a packet)
  - but router should give flows time to reduce their sending rates before dropping more packets

• Therefore, packet drops should be:
  - early: don’t wait for queue to overflow
  - random: don’t drop all packets in burst, but space drops out
RED

- FIFO scheduling
- Buffer management:
  - Probabilistically discard packets
  - Probability is computed as a function of average queue length (why average?)

![Graph showing discard probability vs. average queue length]
RED (cont’d)

- min_th – minimum threshold
- max_th – maximum threshold
- avg_len – average queue length
  - $avg\_len = (1-w)*avg\_len + w*sample\_len$
RED (cont’d)

• If (avg_len < min_th) → enqueue packet
• If (avg_len > max_th) → drop packet
• If (avg_len ≥ min_th and avg_len < max_th) → enqueue packet with probability $P$

Discard Probability (P)

<table>
<thead>
<tr>
<th>min_th</th>
<th>max_th</th>
<th>queue_len</th>
<th>Average Queue Length</th>
</tr>
</thead>
</table>
RED (cont’d)

- \( P = \max_P \cdot \frac{(avg\_len - min\_th)}{(max\_th - min\_th)} \)
Average vs. Instantaneous Queue
RED Advantages

• High network utilization with low delays

• Average queue length small, but capable of absorbing large bursts

• Many refinements to basic algorithm make it more adaptive (requires less tuning)
Explicit Congestion Notification

• Rather than drop packets to signal congestion, router can send an explicit signal

• Explicit congestion notification (ECN):
  - instead of optionally dropping packet, router sets a bit in the packet header
  - If data packet has bit set, then ACK has ECN bit set

• Backward compatibility:
  - bit in header indicates if host implements ECN
  - note that not all routers need to implement ECN
ECN Advantages

• No need for retransmitting optionally dropped packets

• No confusion between congestion losses and corruption losses