Congestion Control (in-the-middle view)
Today’s Questions

- Why do in-the-middle congestion control?
- What are techniques to control congestion in middle?
Why in-the-middle congestion control?
Why in-the-middle congestion control?

• Congestion occurs “in the middle”
  – Can be directly measured, manipulated, …
  – Resources are local (buffers, queues, link bandwidths)

• Control more tightly integrated
  – No need to wait-and-see if packet was lost, RTT varied, etc..

• Much more attack resilient!
Ways to Control Congestion in the Middle
Ways to Control Congestion in the Middle

- **Scheduling**
  - Decide order in which packets are forwarded
  - FIFO, fair queuing, priority queuing

- **Queue management**
  - Decide how to “manage” buffer resources
  - Drop-tail, random-drop, RED
Scheduling

• FIFO has problems…
Scheduling

• FIFO does not guarantee fairness
  – Flows interference
  – Hosts might pro-actively blast packets

• Alternatives:
  – Priority scheduling
  – Fair queuing
Fair Queuing (FQ)

• Maintain one queue per flow
  – Round-robin service each queue
  – Provides each flow with its “fair share” of bandwidth
    • Packets have different sizes!

• Problem:
  – What is a “flow”? 
FQ Description

• Bit-level round robin doesn’t work in practice
• Approximate using packet “finish” times
  – Finish time depends on number of flows
  – Send in order of finish times
  – Lower delay if flow underutilizes bandwidth
    • VoIP
FQ Problems
FQ Problems

• Implementation complexity
  – Need as many queues as flows
  – Per-flow state
  – $O(\log(#\text{flows}))$ processing per packet <-- ouch!

• Delay increases for low-bandwidth & bursty flows
  – Speak monotonically when you use VoIP + FQ :-)

Ways to Control Congestion in the Middle

• Scheduling
  – Decide order in which packets are forwarded
  – FIFO, fair queuing, priority queuing

• Queue management
  – Decide how to “manage” buffer resources
  – Drop-tail, random-drop, RED
Drop-Tail Queuing Policy

- **Pros:**
  - Simple and efficient to implement

- **Cons:**
  - Bias against bursty connections
  - Creates global synchronization
  - Unfair
Active Queue Management

• Let’s avoid congestion rather than control it

• ECN

• Random early detection (RED)
Random Early Detection (RED)

• Congestion avoidance by detection incipient congestion and dropping packets early
• High-level view:
  – Triggered when average queue length exceeds threshold
  – Packets dropped at random (proportional to bandwidth share)
  – No per-flow state
  – Dropping and marking are equivalent in semantics
  – Agnostic to scheduling discipline
  – Incrementally deployable
Queue Dynamics
Queue Management in RED

- Send “early” signal by probabilistically dropping packet
Setting the Probability Parameter

- Probabilistically drop as queue builds
- Switch to drop-tail if queue too long
RED vs. ECN

- Which one is better, or are they equivalent?
RED vs. ECN

• Which one is better, or are they equivalent?

• Misbehaving users
  – RED punishes them
  – Could implement sampling mechanisms in ECN as well

• ECN helps short flows
Two ways to do congestion control in the middle
  – Scheduling FIFO, fair queuing, priority queuing
  – Queue management: drop-tail, RED, ECN

RED + ECN have seen little deployment
Most router queues are drop-tail, FIFO by default
Still very much an open problem!