CSEP 561 – Retransmissions

David Wetherall
djw@cs.washington.edu
Retransmissions

• Focus
  – How do we decide when to retransmit?

• Topics
  – RTT estimation
  – Karn/Partridge algorithm
  – Jacobson/Karels algorithm
Deciding When to Retransmit

• How do you know when a packet has been lost?
  – Ultimately sender uses timers to decide when to retransmit

• But how long should the timer be?
  – Too long: inefficient (large delays, poor use of bandwidth)
  – Too short: may retransmit unnecessarily (causing extra traffic)
  – A good retransmission timer is important for good performance

• Right timer is based on the round trip time (RTT)
  – Easy for LANs, varies greatly for wide area (path length, queuing)
Effects of Early Retransmissions

Same packet sent several times
Congestion Collapse

• In the limit, early retransmissions lead to **congestion collapse**
  – Sending more packets into the network when it is overloaded exacerbates the problem of congestion
  – Network stays busy but very little useful work is being done

• This happened in real life ~1987
  – Led to Van Jacobson’s TCP algorithms, which form the basis of congestion control in the Internet today
    [See “Congestion Avoidance and Control”, SIGCOMM’88]
Congestion Collapse

- Retransmissions lead to throughput, but not goodput
Estimating RTTs

- Idea: Adapt based on recent past measurements

- Simple algorithm:
  - For each packet, note time sent and time ack received
  - Compute RTT samples and average recent samples for timeout
  - EstimatedRTT = $\alpha \times$ EstimatedRTT + $(1 - \alpha) \times$ SampleRTT
  - This is an exponentially-weighted moving average (low pass filter) that smoothes the samples. Typically, $\alpha = 0.8$ to 0.9.
  - Set timeout to small multiple (2) of the estimate
Estimated Retransmit Timer

Close call!

But has large gap to keep estimate > actual
Karn/Partridge Algorithm

- Problem: RTT for retransmitted packets ambiguous

- Solution: Don’t measure RTT for retransmitted packets and do not relax backed-off timeout until valid RTT measurements
Jacobson/Karels Algorithm

• Problem:
  – Variance in RTTs gets large as network gets loaded
  – So an average RTT isn’t a good predictor when we need it most

• Solution: Track variance too.

  – Difference = SampleRTT – EstimatedRTT
  – EstimatedRTT = EstimatedRTT + (δ x Difference)
  – Deviation = Deviation + δ(|Difference| - Deviation)

  – Timeout = μ x EstimatedRTT + φ x Deviation
  – In practice, δ = 1/8, μ = 1 and φ = 4
Estimate with Mean + Variance