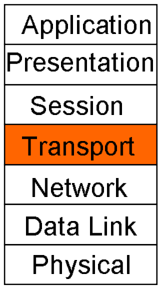
CSE/EE 461 – Lecture 15

Retransmission and Timers

David Wetherall
djw@cs.washington.edu

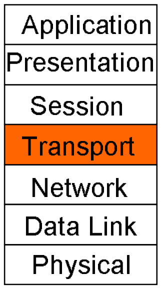
Last Time …

- More on the Transport Layer
- Focus
  - How do we manage connections?
- Topics
  - Three-Way Handshake
  - Close and TIME_WAIT



This Lecture

- More on the Transport Layer

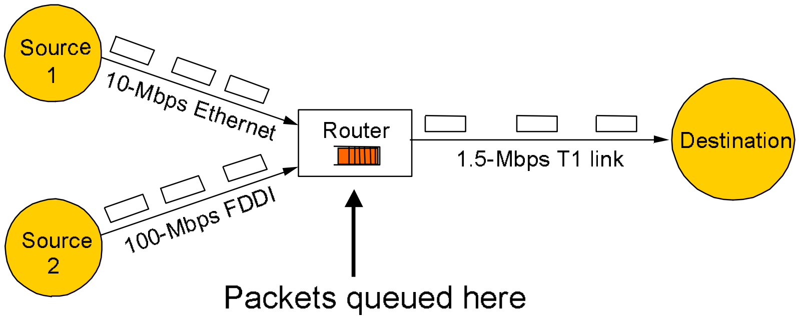


- Focus
  - How do we decide when to retransmit?
- Topics
  - RTT estimation
  - Karn/Partridge algorithm
  - Jacobson/Karels algorithm

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)
  - Which varies greatly in the wide area (path length and queuing)

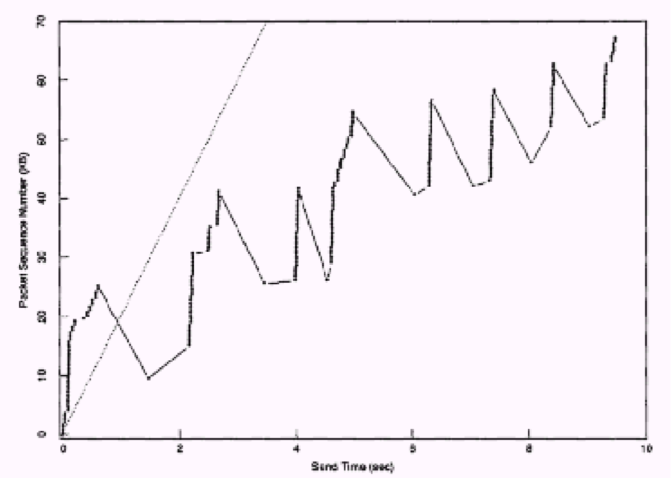
A Simple Network Model



Packets queued here

- Buffers at routers used to absorb bursts when input rate > output
- Loss (drops) occur when sending rate is persistently > drain rate

Effects of Early Retransmissions



Trace data of the start of a TCP conversation between two Sun 3/50s running Sun OS 3.5 (the 4.3BSD TCP). The two Suns were on different Ethernets connected by IP gateways

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

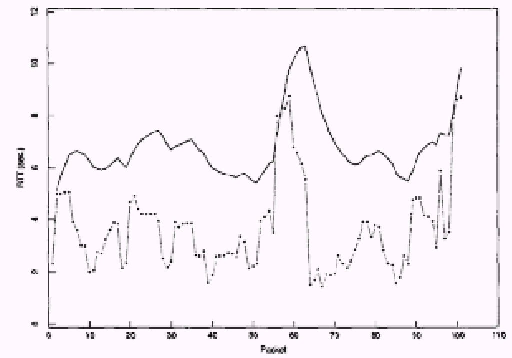
  [See "Congestion Avoidance and Control", SIGCOMM'88]

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 = α x EstimatedRTT + (1 - α) x 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

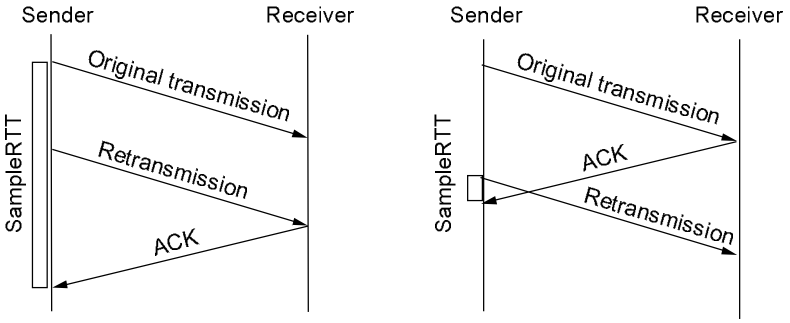


Trace data showing per-packet round trip time on a well-behaved Arpanet connection. The x-axis is the packet number (packets were numbered sequentially, starting with one) and the y-axis is the elapsed time from the send of the packet to the sender's receipt of its ack. During this portion of the trace, no packets were dropped or retransmitted.

The packets are indicated by a dot. A dashed line connects them to make the sequence

Karn/Partridge Algorithm

- Problem: RTT for retransmitted packets ambiguous



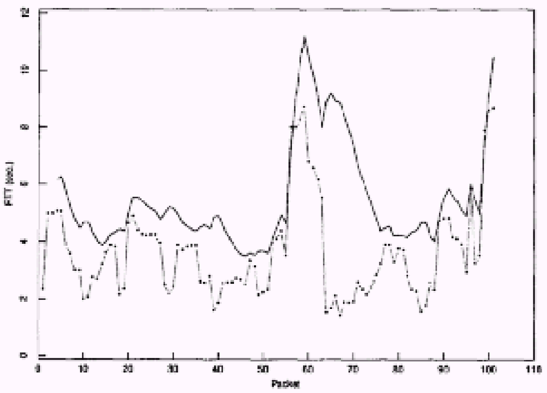
- Solution: Don't measure RTT for retransmitted packets and do not relax backed of timeout until valid RTT measurements

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

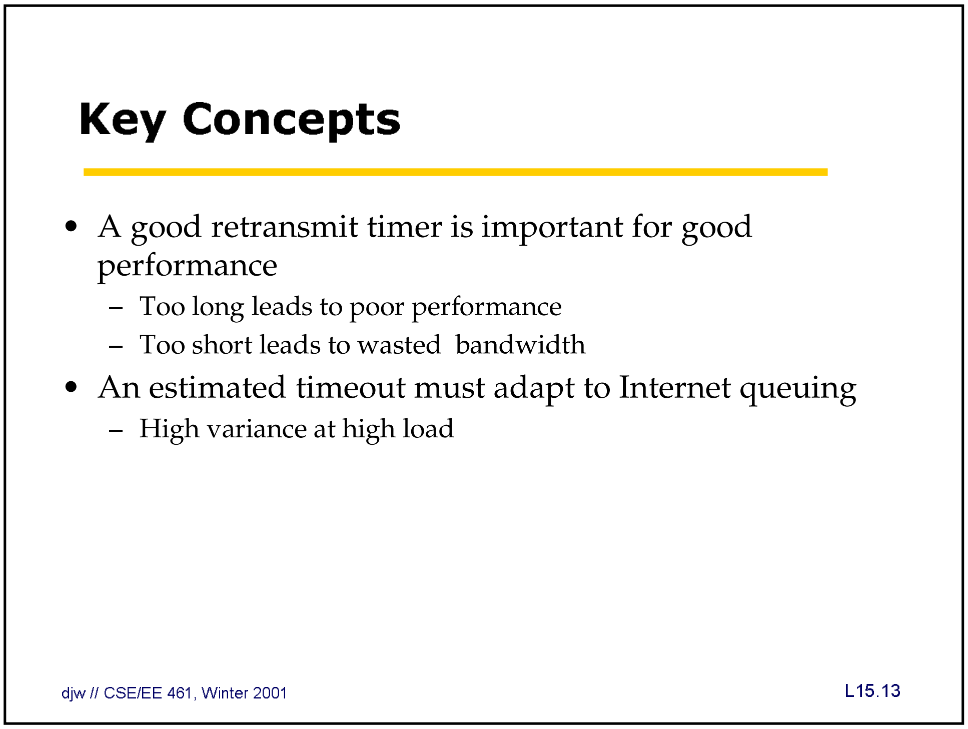
- Solution: Track variance too.
  - Difference = SampleRTT – EstimatedRTT
  - EstimatedRTT = EstimatedRTT + (δ x Difference)
  - Deviation = Deviation + δ(| Difference | - Deviation)
  - Timeout = μ x EstimatedRTT + ϕ x Deviation
  - In practice, δ = 1/8, μ = 1 and ϕ = 4

Estimate with Mean + Variance



Same data as above but the solid line shows a retransmit timer computed according to the algorithm in appendix A.

Figure 6: Performance of a Mean+Variance retransmit timer



Key Concepts

- A good retransmit timer is important for good performance
  - Too long leads to poor performance
  - Too short leads to wasted bandwidth
- An estimated timeout must adapt to Internet queuing
  - High variance at high load