Announcements

- Homework #2 was returned
- Reading
  - Today: 3.1
  - Thursday: 3.2

TCP Timer Management

- Problem: How to pick timeout value?
  - need to estimate round-trip latency
  - need low variance in round trip latency
- Solution: dynamic estimates of RTT
  - $RTT = \alpha \, RTT + (1 - \alpha) \, M$

    M time of an ACK

    $\alpha = 7/8$
  - Need to pick retransmission time
    - old policy, use Timeout = RTT β , with $\beta = 2$
    - estimate standard deviation of RTT using mean deviation

      $D = \alpha \; D + (1 - \alpha) \, | RTT - M |$

      Timeout = $RTT + 4 * D$
  - How to update RTT on retransmission's
    - double Timeout on a retransmission

Other TCP Timers

- Persistence Timer
  - Prevents deadlock due to dropped window packets
    - This is a problem if the window is set to 0
- Keepalive Timer
  - Prevents half dead connections
  - may consume bandwidth
  - may kill live connections when net hiccups
- TIMED Wait
  - prevents re-use of a connection before max packet life is over
  - set to twice max packet lifetime

Performance Issues

- Broadcast storms
  - response to a broadcast packet sent by many hosts
  - caused by:
    - bad parameter resulting in an error message
    - asking a question everyone has the answer to
- Reboot storms
  - RARP queries
  - file servers responding to page requests
- Delay-bandwidth product
  - need to buffer at least as many bytes as can be “in flight”
- Jitter
  - keep standard deviation of packet arrivals low
  - important for continuous media traffic

How to Measure Performance

- Ensure sample size is large
  - repeat experiments for several iterations
- Make sure samples are representative
  - consider time of day, location, day of week, etc.
- Watch for clock resolution/accuracy
  - don't use two clocks at opposite ends of the network
  - if the clock resolution is poor, aggregate over multiple iterations
- Know what you are measuring
  - is a cache going to distort results?
  - is the hardware, OS, device driver, compiler the same?
- Careful not to extrapolate too far
  - results generally hold for an operating region, not all values

How to Design in Performance

- CPU Speed is more important than link speed
  - protocol processing time is the critical time for most networks
  - use simple algorithms for your network
- Reduce packet count
  - there is a large per packet cost in most levels
  - big packets amortize this overhead over more bytes
- Minimize Context Switches
  - user/kernel boundary crossings are expensive
    - require many cache misses, pipeline stalls, etc.
  - send large units of data
- Minimize Copying
  - each copy is extra time
  - memory operations are often 10 times slower than other insns

How To Design In Performance (cont.)

- Bandwidth is growing, but latency isn't shrinking as fast
  - fundamental limits of how many rounds trips are possible
  - need to design to transfer large requests
- Congestion Avoidance beats Recovery
  - getting the network out of a bad state will take time
  - better to prevent getting it there in the first place
- Avoid Timeouts
  - use NACKs to get info back
  - use long values for timeouts
  - timeouts result in:
    - interrupts (slow for the processor)
    - re-transmission (slow for the link)
- Make The Common Case Run Fast
  - data transmission is more common than connect