Today’s Lecture

- Project 3
  - In-Class Exercise
- Network Issues
  - Congestion Avoidance
  - Quality of Service

Project 3

- Ninja C / C++
  - Pointers / etc.
- Discussion / in-class exercise
Congestion Control

Outline
- Congestion Avoidance
- RED
- TCP Vegas

Random Early Detection (RED)

- Notification is implicit
  - just drop the packet (TCP will timeout)
  - could make explicit by marking the packet
- Early random drop
  - rather than wait for queue to become full, drop each arriving packet with some drop probability whenever the queue length exceeds some drop level

RED Details (cont)

- Computing probability $P$
  
  $TempP = \text{MaxP} \times \frac{\text{AvgLen} - \text{MinThreshold}}{\text{MaxThreshold} - \text{MinThreshold}}$
  $P = \frac{TempP}{1 - \text{count} \times TempP}$

- Drop Probability Curve
Tuning RED

- Probability of dropping a particular flow’s packet(s) is roughly proportional to the share of the bandwidth that flow is currently getting
- MaxP is typically set to 0.02, meaning that when the average queue size is halfway between the two thresholds, the gateway drops roughly one out of 50 packets.
- If traffic is bursty, then MinThreshold should be sufficiently large to allow link utilization to be maintained at an acceptably high level
- Difference between two thresholds should be larger than the typical increase in the calculated average queue length in one RTT; setting MaxThreshold to twice MinThreshold is reasonable for traffic on today’s Internet

TCP Vegas

- Idea: source watches for some sign that router’s queue is building up and congestion will happen too; e.g.,
  - RTT grows
  - sending rate flattens

Algorithm

- Let BaseRTT be the minimum of all measured RTTs (commonly the RTT of the first packet)
- If not overflowing the connection, then
  \[
  \text{ExpectedRate} = \frac{\text{CongestionWindow}}{\text{BaseRTT}}
  \]
- Source calculates sending rate (ActualRate) once per RTT
- Source compares ActualRate with ExpectedRate
- If Diff < α, increase CongestionWindow linearly
- Else if Diff > β, decrease CongestionWindow linearly
- Else, leave CongestionWindow unchanged
Algorithm (cont)

- Parameters
  - \( \alpha = 1 \) packet
  - \( \beta = 3 \) packets

Derivation

- How long will something take via TCP?
- Derive
  - ROM – Rough Order of Magnitude Estimates
  - See in-class notes

Quality of Service

Outline
- Realtime Applications
- Integrated Services
- Differentiated Services
Realtime Applications

- Require “deliver on time” assurances
  - must come from inside the network

- Example application (audio)
  - sample voice once every 125us
  - each sample has a playback time
  - packets experience variable delay in network
  - add constant factor to playback time: playback point

![Diagram of microphone, sampler, buffer, and speaker]

Playback Buffer

Example Distribution of Delays

![Graph showing distribution of delays]
**Integrated Services**

- **Service Classes**
  - guaranteed
  - controlled-load

- **Mechanisms**
  - resource reservation (signalling)
  - admission control
  - policing
  - packet scheduling

**Flowspec**

- **Rspec**: describes service requested from network
  - controlled-load: none
  - guaranteed: delay target

- **Tspec**: describes flow’s traffic characteristics
  - average bandwidth + burstiness: token bucket filter
    - token rate $r$
    - bucket depth $B$
    - must have a token to send a byte
    - must have $n$ tokens to send $n$ bytes
    - start with no tokens
    - accumulate tokens at rate of $r$ per second
    - can accumulate no more than $B$ tokens
Per-Router Mechanisms

- Admission Control
  - decide if a new flow can be supported
  - answer depends on service class
  - not the same as policing
- Packet Processing
  - classification: associate each packet with the appropriate reservation
  - scheduling: manage queues so each packet receives the requested service

Reservation Protocol

- Called signalling in ATM
- Proposed Internet standard: RSVP
- Consistent with robustness of today’s connectionless model
- Uses soft state (refresh periodically)
- Designed to support multicast
- Receiver-oriented
- Two messages: PATH and RESV
- Source transmits PATH messages every 30 seconds
- Destination responds with RESV message
- Merge requirements in case of multicast
- Can specify number of speakers

RSVP Example
RSVP

- Associate packet with reservation (classifying):
  - source address, destination address, protocol number,
    source port, destination port
- Manage packets in queues (scheduling).

RSVP versus ATM (Q.2931)

- RSVP
  - receiver generates reservation
  - soft state (refresh/timeout)
  - separate from route establishment
  - QoS can change dynamically
  - receiver heterogeneity
- ATM
  - sender generates connection request
  - hard state (explicit delete)
  - concurrent with route establishment
  - QoS is static for life of connection
  - uniform QoS to all receivers

Differentiated Services

- Problem with IntServ: scalability
- Idea: segregate packets into a small number of classes
  - e.g., premium vs best-effort
- Packets marked according to class at edge of network
- Core routers implement some per-hop-behavior (PHB)
- Example: Expedited Forwarding (EF)
  - rate-limit EF packets at the edges
  - PHB implemented with class-based priority queues or WFQ
DiffServ (cont)

- Assured Forwarding (AF)
  - customers sign service agreements with ISPs
  - edge routers mark packets as being "in" or "out" of profile
  - core routers run RIO: RED with in/out