6.892 Fall 1999 **Lecture 5: RED Gateways** September 22, 1999

Overview. The gateway side of congestion control. RED. Incentives for end-to-end congestion control.

1 Gateway support for congestion management

- Can't assume cooperating sources any more.
- Even if we did, biases exist that prevent fair allocation (e.g., rtt biases).
- Danger of congestion collapse: (beyond cliff of throughput-delay curve).
- What forms might congestion collapse take?
  1. *Classical collapse:* Redundant retransmissions.
  2. *Fragmentation-based collapse:* Loss of fragments causes entire packet to be retransmitted.
  3. *Control traffic-triggered collapse:* E.g., increased amounts of multicast join/prunes, RTCP traffic, packet headers, routing updates, etc.
  4. *Collapse from stale packets:* Network carries packets that are no longer wanted by user, because of long delays. Or because of ill-informed *push* data to users.
  5. *Undelivered packets:* This is the hardest form to tackle, and is caused by unresponsive traffic. Some applications also incrase their bandwidth use upon detecting loss, sometimes by increased FEC (FEC in itself isn't bad, it's the increased bandwidth use that is).
- Three ways of tackling this:
  1. Flow isolation and per-flow scheduling.
  2. Router mechanisms that encourage end-to-end adaptation to congestion.
  3. Pricing.
- We will discuss RED and its applicability to #2 and fair queueing as a mechanism for #1 in this lecture and next.

2 RED

- Goals: control average queue size and at the same time, and accomodate bursty traffic.
- Key idea: use randomization to avoid synchronizations and biases that could occur in deterministic approaches.

- Basic idea: If average queue size, q_a is between min_{th} and max_{th} implies *mark* or *drop* packet with some probability, p_a. If $q_a > max_{th}$, then drop packet. If $q_a < min_{th}$, then do nothing.
- If marking is used and TCP and other sources understand this, we can avoid some losses. This is called *Explicit Congestion Notification (ECN).*
- Differences from DECbit:
  1. Method for computing q_a. Regeneration vs. EWMA. Latter explicitly controls time constant for average calculation. Biases don't arise because of randomization in packet marking.
  2. Use of randomization. Avoids synchronizations. (Note that this is more likely to happen with TCP's 0.5 decrease factor than DECbit's 7/8).
  3. Which connections to notify of congestion? Claim is DECbit could penalize bursty traffic. (This isn't clear, esp. since DECbit was later modified to allow selective feedback).
  4. RED much more in tune with TCP; DECbit tuned for DNA's transport protocol (& ISO TP4).
- RED can also handle misbehaving or uncooperative sources, by penalizing them ("penalty box").
- Two key issues in RED: how to compute q_a and how to compute p_a.

2.1 Computing q_a

- $q_a = (1 - w_q)q_a + w_q q$. w_q is time constant of low pass filter. Too large implies rapid response to *transient* congestion. Too low implies sluggish behavior.
- Get upper bound based on how much burstiness you want to allow.
- Get lower bound based on how much memory you want to maintain.
- In practice, Floyd sets $w_q = 0.002$.
- $max_{th} \geq 2min_{th}$.

2.2 Computing p_a

- Initially, compute $p_b = max_p(q_a - min_{th})/(max_{th} - min_{th})$. This is standard linear interpolation.
- Method #1: Geometric r.v. for inter-arrivals between markings (or drops). Supose each packet is marked with probability $p_a = p_b$. Consider the r.v X = number of packets between successive marked packets. Then $P(X = n) = (1 - p_b)^{n-1}p_b$. This is a geometric distribution.
- Unfortunately, this doesn't mark packets at fairly regular intervals. $E[X] = 1/p_b$, but because packets get marked in bunches, more synchronization than is good could occur. $Var[X] = 1/p_b^2 - 1/p_b$, which gets big for small p_b.

- Method #2: Uniform r.v. for inter-arrivals between markings or drops. Let X be uniform in $\{1, 2, \ldots, 1/p_b - 1\}$. This has the nice property that we want, and is achieved if we set $p_a = p_b/(1 - count \times p_b)$, where *count* is the number of unmarked packets since last marked packet. $E[X] = 1/(2p_b)$.
- max_p should never have to be large. (Anyway, after $q_a > max_{th}$, all packets are dropped).

3 Encouraging end-to-end congestion control

3.1 Motivation

Increasing amount of non-congestion-controlled best-effort traffic in the Internet today, a trend that could lead to congestion collapse. We need mechanisms to handle this and prevent this from happening.

Why do we need congestion control?

- Bandwidth starvation of, and unfairness towards better-behaved flows. E.g., UDP flows can starve TCPs, TCP biases with RTTs, etc.
- Danger of *congestion collapse.* (Sharp negative slope of throughput-load curve.) Floyd & Fall characterize five kinds of collapse.
  - *Classical collapse.* Due to premature retransmissions.
  - *Fragmentation-based collapse.* Fragments get lost, requiring retransmissions of full datagrams and wasting bandwidth.
  - *Control-triggered collapse.* Non-data traffic.
  - *Stale packets.* Imagine getting 80% of a web image, hitting stop, then reload, and getting the whole thing again because your browser doesn't cache partial images. If this happens, lots of "stale" packets traverse the network.
  - *Collapse from undelivered packets.* This is the hardest form and the most dangerous one for the future. The paper has an excellent description of this.

3.2 Solutions

Three possible approaches towards sharing scarce resources for best-effort traffic:

- Flow isolation, using fair queueing (next class) or variants using per-flow scheduling mechanisms. These are complex to implement, but prevent flows from affecting each other at least at a single router. They don't provide incentives to perform end-to-end congestion control, though (such incentives in FQ are expected to be done using pricing). They also don't provide flow isolation across multiple bottlenecks, increasingly becoming common.
- Pricing.
- End-to-end congestion control with router-based incentives.

Main message of this paper: *need router incentives to encourage end-to-end congestion control!*. They suggest and motivate a mechanism that uses RED's drop (mark) history to classify ill-behaved flows and perhaps penalize them.

3.3 Using RED to detect ill-behaved flows

- In the long run, connection's share of marked packets proportional to its share of bandwidth. Note however that RED doesn't provide flow isolation and isn't fair in the fair queueing ense of max-min fairness.
- For a fixed q_{avg}, n packet drops, and a fraction b of arriving bandwidth, P(flow receives $> 2 \times$ its "fair share" of drops) $\leq e^{-2nb^2}$.

  So, over a sufficiently large number of drops (and a drop rate that's not too large), the RED packet drop history should be a good estimator of the *long-term* arrival rate of a flow.
- Three types of ill-behaved flows:
  - Flows that aren't *TCP-friendly.* These are more aggressive than the throughput $\propto 1/\sqrt{p}$ relationship for conformant TCPs.
  - Flows that are *unresponsive.* These don't reduce their rate in proportion to increases in loss rate.
  - Flows that are using *disproportionate bandwidth.* These are both responsive and TCP-friendly, but using more than their fair share (e.g., "significantly" more than $1/n$ when there are n flows in all).

  The paper proposes tests for these, most of which are probably not precisely the ones that will eventually be deployed.
- Put ill-behaved in a *"penalty box"* and don't service them. I.e., provide a *disincentive* to be unresponsive to congestion.

3.4 What are the contributions of this paper?

- Identifying an important problem and providing a nice basis for future work in this area. Important to have incentives to encourage responsible behavior in the face of congestion.
- The schemes in this paper themselves may not be the ones adopted in reality.
- It raises many questions:
  1. It seems conceivable that we can identify long-running ill-behaved flows. But what about short or bursty bad flows?
  2. To what extent can pricing structures provide the similar incentives?
  3. What are the right schemes to use in practice?