May 7 2002. Second half of Networks class

We discussed pros and cons of congestion avoidance. It seemed that if the number of packets sent per connection is small, the price paid for slow start is not worth it. A typical HTTP request would contain 10K. With the typical MTU, this would be around 20 packets.

When a sender of a short flow is notified about congestion, it is anyway almost done sending data. In such a case, no benefit was gained and we also paid a price in terms of lowered throughput when trying to realize the saturation point.

The discussion naturally progressed towards solutions that attempt to aggregate avoidance mechanisms across multiple connections per host and across routes. Linux has already incorporated aggregating this information from multiple connections.

Congestion avoidance is generally done as a collaboration venture between routers and the end hosts. The routers can readily detect congestion and notify the end hosts about it. The end hosts in turn oblige by cutting down their data rates.

One problem is to make sure that all hosts do not simultaneously back off. So the routers select a number of end hosts to notify, but not all of them at once. (I remembered Tom's earlier analogy – 520 is blocked so everyone stays home, then 520 is wide open again…)

Some routers notify end hosts of congestion by dropping packets early. The end hosts will get duplicate ACKS and figure out that packets are getting dropped, so they will in turn cut down the data rates.

Some routers will mark a flag CONGESTION in the packets as they pass through. The receiver will copy the flag back onto the ACK packet. The sender, upon receiving packets marked with CONGESTION flag, will reduce the send rate. The problem with this approach is that it requires changes at the router, sender and receiver. You also need to find space in the IP header to set the CONGESTION flag.

The thresholds that the routers choose are problematic at best. There is current research that attempts to dynamically set thresholds.

Since routes are asymmetric, it is not possible for the router to mark the ACK packets already in it's receive queue when it discovers congestion on the send queue.

TCP behaves badly if the ACK path gets congested and not the send path.

Some esoteric designs that attempt to control congestion at the router don't make the cut due to the increased hardware complexity. Since data throughput is a high priority, most logic in the router needs to be hard wired.

One scheme that attempts fair queuing needs a queue for each flow. If a flow exceeds its queue capacity, its packets will be dropped. This is fair, and much like how processes get their time slice in an operating system. And similarly, priorities can be implemented with priority queues.

Penalty Box is a design that attempts to monitor senders and upon congestion, start dropping packets from the big senders. This is an attempt at fair queuing. I'm not sure how complicated the hardware need be for monitoring.

There was an argument that front-drop may notify the sender early on about congestion. The argument against front-drop was that this could lead to lots of small traffic getting dropped. It is basically a question of fairness – FIFO seems fairer than LIFO, and indiscriminate LIFO would ensure that huge packets have a higher chance of getting through. Also, it was dubious as to how much sooner the sender would have learnt of congestion.