A Proposal to add Explicit Congestion Notification (ECN) to IPv6 and to TCP

K. K. Ramakrishnan,
Sally Floyd

References:

Ramakrishnan, K.K., and Floyd, S.,
A Proposal to add Explicit Congestion Notification (ECN) to IPv6 and to TCP,
Internet draft draft-kksjf-ecn-00.txt, work in progress,
November 1997.

The ECN Web Page: http://www-nrg.ee.lbl.gov/floyd/ecn.html

The viewgraphs:

- Where would the bits come from? (feedback?)

- What is needed from IPv6?

- Why two bits instead of one?

- Why not Source Quench?

- How would the router decide whether to drop the packet or set the ECN bit?

- Would ECN make it easy for evil applications?

- What applications would benefit from ECN?

- Simulations and experiments. (feedback?)

Where would the bits come from?

The IPv4 TOS byte and/or the IPv6 Traffic Class byte.

Our original proposal was only in terms of IPv6.

IPv4: the current draft of draft-ellesson-tos-00.txt includes the CE (Congestion Experienced) and ECT (ECN-capable transport) bits.

What would be the time scale of making this decision?
Where? Intserv/diff-serv? or ipng?

Why two bits instead of one?

THE ONE-BIT APPROACH: the single bit would have two values: “ECN-Capable”, and “either not ECN-Capable, or Congestion Notification”.

The ECN-Capable functionality is needed to allow a viable incentive for the incremental deployment of ECN-aware hosts.

Robustness issues for the one-bit approach:
(1) The default setting **must** be “either not ECN-Capable, or Congestion Notification”.
(2) The receiver has to know in advance that the sender is sending all packets with the bit set as “ECN-Capable”.

Why not Source Quench?

(1) The delay difference (between an RTT and part of an RTT) is not a critical issue for most environments.

(2) The ECN bit is applicable for multicast as well as for unicast applications; Source Quench-style mechanisms are not.

(3) With Source Quence, care must be taken not to add significant extra traffic to the network in times of congestion.

How would the router decide whether to drop the packet or set the ECN bit?

For ECN-capable packets, the router would set the ECN bit when it is in the regime of using RED to “mark” packets probabilistically.
When the average queue size exceeds the maximum queue threshold, the router would drop arriving packets.

For most routers today, the packet drop rate is less than 10%.
If these routers deploy RED, they are therefore generally in the regime of probabilistic marking.

Does ECN make it easy for “evil” applications to ignore congestion control?

It is already easy for “evil” applications to ignore congestion control, if they so desire. Just use UDP, and add a little FEC to counteract the packet drop rate. Or “turn off” congestion control for TCP.

In the absence of congestion control in the end-nodes, a moderate (less than 10%) packet drop rate is not, in and of itself, an effective mechanism of congestion control, or an effective deterrent to any application that wishes to “turn off” end-to-end congestion control.

ECN does add another way that applications could ignore congestion control, by setting the “ECN-capable” bit but ignoring the ECN bit in received packets.

ECN Simulations and Experiments

• Simulations: the ns-1 and ns-2 simulators, [Floyd 1994].

• Experiments: UCLA (Chris Chen, Hariharam Krishnan, Steve Leung, Nelson Tang) has an ECN implementation in IPv6 in FreeBSD 2.2.2. Using the RED implementation from ALTQ.

• Two bits to use in the IPv6 header for experimental purposes? Feedback?

Does ECN make it harder to deploy mechanisms in the routers that would detect applications that are not using congestion control in times of congestion?

The use of ECN does not make it more difficult for routers to deploy mechanisms that protect the Internet against applications that ignore congestion control. Instead, the use of ECN instead of packet drops would make it slightly easier to deploy such mechanisms.

Applications that could benefit from ECN:

• Reliable multicast, where the overhead costs of retransmitting dropped packets can be nontrivial.

• Realtime flows with (fixed or adaptive) playback times, where, given a choice between receiving the packet or not, the user would rather receive the packet.

• Very short web transfers, where the user would prefer not to wait for TCP's retransmit timer to expire to detect the loss of a single packet in a single-packet transfer.

• Low-bandwidth telnet connections.

• Any other application where the user would rather have some form of congestion notification other than a dropped packet.