TCP and Explicit Congestion Notification

Sally Floyd*

Lawrence Berkeley Laboratory
One Cyclotron Road, Berkeley, CA 94704
floyd@ee.lbl.gov

Abstract

This paper discusses the use of Explicit Congestion Notification (ECN) mechanisms in the TCP/IP protocol. The first part proposes new guidelines for TCP's response to ECN mechanisms (e.g., Source Quench packets, ECN fields in packet headers). Next, using simulations, we explore the benefits and drawbacks of ECN in TCP/IP networks. Our simulations use RED gateways modified to set an ECN bit in the IP packet header as an indication of congestion, with Reno-style TCP modified to respond to ECN as well as to packet drops as indications of congestion. The simulations show that one advantage of ECN mechanisms is in avoiding unnecessary packet drops, and therefore avoiding unnecessary delay for packets from low-bandwidth delay-sensitive TCP connections. A second advantage of ECN mechanisms is in networks (generally LANs) where the effectiveness of TCP retransmit timers is limited by the coarse granularity of the TCP clock. The paper also discusses some implementation issues concerning specific ECN mechanisms in TCP/IP networks.

1 Introduction

This paper proposes guidelines for TCP's response to ECN (Explicit Congestion Notification) mechanisms, and explores the effect upon performance of ECN mechanisms in TCP/IP networks. The paper discusses some implementation issues concerning ECN mechanisms, but does not make specific recommendations concerning the use of ECN mechanisms in TCP/IP networks.

In current TCP/IP networks, TCP relies on packet drops as the indication of congestion. The TCP source detects dropped packets either from the receipt of three duplicate acknowledgements (ACKs) or after the timeout of a retransmit timer, and responds to a dropped packet by reducing the congestion window [J88]. TCP implementations also respond to ICMP Source Quench messages, but Source Quench messages are rarely used, in part because they can consume network bandwidth in times of congestion.

The reliance on packet drops as the indication of congestion is perfectly appropriate for a network with routers whose main function is to route packets to the appropriate output port. Most current routers in TCP/IP networks have no provision for the detection of incipient congestion. When a queue overflows, packets are dropped. When the TCP source detects this packet drop, the TCP source infers the presence of congestion in the network.

Future routers are likely to have more developed mechanisms for the detection of incipient congestion. With the DECbit scheme, for example, routers detect incipient congestion by computing the average queue size, and set the ECN bit in packet headers when the average queue size exceeds a certain threshold [RJ90]. Recently-proposed Random Early Detection (RED) gateways have a similar ability to detect incipient congestion [FJ93]. Gateways with mechanisms for detecting incipient congestion before the queue overflows are not limited to packet drops as the method of informing sources of congestion.

For networks with mechanisms for the detection of incipient congestion, the use of ECN mechanisms for the notification of congestion to the end nodes prevents unnecessary packet drops. For bulk-data connections, the user is concerned only with the arrival time of the last packet of data, and delays of individual packets are of no concern. For some interactive traffic, however, such as telnet traffic, the user is sensitive to the delay of individual packets. For such low-bandwidth delay-sensitive TCP traffic, unnecessary packet drops and packet retransmissions can result in noticeable and unnecessary delays for the user. For some connections, these delays can be exacerbated by a coarse-granularity TCP timer that delays the source's retransmission of the packet.

A second benefit of ECN mechanisms is that with ECN, sources can be informed of congestion quickly and unambiguously, without the source having to wait for either a retransmit timer or three duplicate ACKs to infer a dropped packet. For bulk-data TCP connections,

*This work was supported by the Director, Office of Energy Research, Scientific Computing Staff, of the U.S. Department of Energy under Contract No. DE-AC03-76SF00098.

the delay for the retransmission of an individual packet is not generally an issue. For bulk-data TCP connections in wide-area environments, the congestion window is generally sufficiently large that the dropped packet is detected fairly promptly by the Fast Retransmit procedure. Nevertheless, for those cases where a dropped packet is not detected by the Fast Retransmit procedure, the use of ECN mechanisms can improve a bulk-data connection's response to congestion. If the source is delayed in detecting a dropped packet, perhaps due to a small congestion control window and a coarse-grained TCP timer, the source can lie idle. This delay, when combined with the global synchronization, can result in substantial link idle time.

An additional motivation for the exploration of ECN mechanisms in TCP/IP networks concerns the possibility of TCP/IP traffic traversing networks that have their own congestion control mechanisms (e.g., ATM networks). With current implementations of TCP, such networks are limited to packet drops as the only viable mechanism to inform TCP sources of congestion. Such networks would benefit from the addition to TCP of more intelligent ECN-response methods. If this were the case, then for TCP traffic that travels for all or part of its path over ATM networks, ECN mechanisms could be invoked at the edge of the ATM network and used to inform TCP sources of congestion within the ATM network. This use of ECN mechanisms to inform TCP sources of congestion would be independent of the congestion control mechanisms within the ATM networks.

This paper explores some of the general advantages and disadvantages of ECN mechanisms in TCP/IP mechanisms, but does not make recommendations for or against specific ECN mechanisms (e.g., the addition of an ECN field to IP headers). The results in this paper are intended to be qualitative, not quantitative. For example, while it is clear that the use of ECN can reduce the packet delay for low-bandwidth delay-sensitive TCP traffic, the extent of this benefit depends on the exact network topology, the traffic mix, and the details of the relevant gateway and transport congestion control algorithms. The simulations in this paper show that the use of ECN can reduce packet delay, but they do not quantify the expected reduction in packet delay in a particular network.

Section 2 discusses some current ECN mechanisms, such as Source Quench in TCP/IP networks. Section 3 briefly discusses the role of the router in detecting incipient congestion. Section 4 presents our proposal for guidelines for TCP's response to ECN; these guidelines differ from those of current network mechanisms. Section 5 discusses the application of these guidelines to the implementation of Reno-style TCP in our simulator. Sections 6 and 7 present our simulation results of LAN and WAN environments. Section 8.2 discusses some of the implications, for the evaluation of ECN mechanisms, for some of the proposed modifications to TCP or to gateway scheduling algorithms. Finally, Section 9 discusses various implementation issues concerning ECN mechanisms in TCP/IP networks. These include comparisons between Source Quench and ECN fields in packet headers, possibilities for the incremental deployment of ECN mechanisms in TCP/IP networks, a discussion of TCP clock granularity, and a discussion of IP-level ECN mechanisms for TCP traffic over ATM networks. Section 10 presents conclusions and open questions.

2 Current ECN mechanisms

In this section we discuss briefly ECN mechanisms such as Source Quench messages, DECbit's ECN bit, and FECN/BECN proposals for ATM networks. One purpose of this section is to describe the current ECN mechanisms in TCP/IP networks. This includes a detailed explanation of the inadequacies of current implementations of Source Quench. In this section we also describe the ECN mechanisms in the DECbit congestion avoidance scheme, partly to motivate the somewhat-different guidelines that we propose in the following two sections for ECN in TCP/IP networks. Finally, we provide pointers to other discussions of proposed ECN mechanisms in the literature.

Currently, ICMP Source Quench messages are the only ECN mechanisms in TCP/IP networks, but in fact Source Quench is rarely used in the current Internet. A router or host might send an ICMP Source Quench message when it receives datagrams at a rate that is too fast to be processed [S94]. While RFC 1009 [BP87] required routers to generate Source Quenches when they ran out of buffers, the current draft on Requirements for IP Routers [A94] specifies that a router should not originate Source Quench messages, and that a router that does originate Source Quench messages must be able to limit the rate at which they are generated. In the draft, Source Quench messages are criticized as consuming network bandwidth, and as being both ineffective and unfair. In contrast, as discussed in the next section, the use of RED gateways with Source Quench messages would control the rate at which Source Quench messages were generated, while increasing both the effectiveness and the fairness of these messages.

The guidelines for TCP's response to Source Quench predate such TCP congestion control mechanisms as Fast Retransmit and Fast Recovery [S94]. From the guidelines in [BP87], TCP implementations should respond to a Source Quench by "triggering a slow start, as if a retransmission timeout had occurred". In BSD TCP implementations, the TCP source responds to a Source Quench by waiting until the current retransmit timer expires, and then initiating Slow-Start. Because of the ordering of the Source Quench code and the later timeout recovery code, in BSD implementations the slow start threshold (*ssthresh*) ends up being set to one or two segments. This results in a very slow reopening of the congestion window. However, even without this problem in BSD implementations, the combination of a wait for a retransmit timer and a Slow-Start makes for a very conservative response to a Source Quench. The wait for the retransmit timer makes the use of Source Quench impractical for LANs with TCP implementations constrained by coarse-grained timers. The use of Slow-Start, combined with a small slow start threshold, makes the use of Source Quench unattractive for large-window TCP connections in high-speed environments. This paper proposes alternate guidelines for TCP's response to Source Quench.

In the DECbit congestion avoidance scheme [RJ90], the gateway uses a congestion-notification bit in packet headers to provide feedback about congestion in the network. When a packet arrives at the gateway, the gateway calculates the average queue length. When the average queue size at the gateway exceeds one, the gateway sets the ECN bit in the packet header of arriving packets.

The source uses window flow control, and updates its window once every two roundtrip times. If at least half of the packets in the last window had the ECN bit set, then the congestion window is decreased multiplicatively. Otherwise, the congestion window is increased additively. In contrast to the DECbit scheme, for the ECN mechanism proposed in this paper a single packet with the ECN bit set is to be interpreted by the transport-level source as an indication of congestion.

[WRM91] reports on experiments in an OSI testbed modified to include the congestion-notification bit proposed in [RJ90]. A range of ECN-based rate-based congestion control schemes have been proposed for use within ATM networks. These include proposals both for Forward ECN (FECN) and for Backward ECN (BECN). An introduction to some of these proposals can be found in [N94].

3 The role of the router

This section discusses the role of the router in generating ECN messages. ECN messages could be generated either by an IP router or by a boundary router for an ATM network that carries TCP/IP traffic. This section considers IP networks with RED gateways, where the gateway monitors the average queue size and during congestion uses a probabilistic algorithm to choose which arriving packets to mark (e.g., to drop, or to set the ECN field in the packet header).

In our simulations using RED gateways with ECN, the RED gateways set the ECN field in the packet header; in the simulations using RED gateways without ECN, the RED gateways drop the packet instead. In both variants of RED gateways, the gateway drops a packet when a packet arrives to a full queue.

It would be possible to add ECN mechanisms to a traditional Drop-Tail gateway, where the gateway simply drops arriving packets when the queue buffer is full. For example, Drop-Tail gateways, after dropping a packet, could send Source Quench messages to the TCP source. However, we do not advocate adding ECN mechanisms to Drop-Tail gateways, and we do not investigate such schemes in this paper. If ECN mechanisms are added to a gateway, it makes sense to add at the same time mechanisms to monitor the average queue size. We believe that the use of ECN mechanisms are of most benefit in a gateway that notifies connections of incipient congestion before the queue actually overflows.

To a first approximation, RED gateways mark (e.g., drop) a percentage of arriving packets, where the exact percentage of arriving packets marked should be just enough to control the average queue size over the long run. For example, in a LAN environment, where a TCP connection increases its congestion window quite rapidly, a non-ECN gateway might have to drop a significant fraction of arriving packets to control congestion. For RED gateways with a FIFO queue, if a certain fraction of bulk-data packets have to be dropped to control congestion, the RED gateway drops the same fraction of telnet packets.

Current routers generally have a single queue for each output port. In the future, routers could have separate queues for separate "classes" of traffic [BCS94]. In this case, the ECN mechanisms could apply separately to each queue. As discussed in Section 8.3, this could affect the motivations for ECN mechanisms.

4 Guidelines for TCP's response to ECN

In this section we explain our guidelines for TCP's response to ECN. These guidelines differ from TCP's current response to Source Quench messages, or from the response of transport protocols to DECbit's congestion notification bit. These guidelines provide that the receipt of a single ECN message serves as a notification of congestion to the TCP source. At the same time, the guidelines ensure that the TCP source does not respond to ECN messages more frequently than necessary.

Guidelines:

- TCP's response to ECN should be similar, over longer time scales, to its response to a dropped packet as an indication of congestion.
- Over smaller time scales (e.g., one or two round trip times), TCP's response to ECN can be less conservative than its response to a dropped packet as an indication of congestion. In Tahoe and Reno implementations of TCP, after a packet has been dropped the TCP source stops sending for a time period on the order of a round trip time (half a round trip time for Reno implementations), allowing network queues to dissipate somewhat. This delay is not necessary as a response to an ECN, which does not indicate a queue overflow.
- For TCP, the receipt of a single ECN (e.g., a single Source Quench packet, or a single packet with the ECN bit set) should trigger a response to congestion. This is unlike the DECbit congestion control scheme, where the source responds to congestion only if at least half of the packets in the last window had the ECN bit set [RJ90]. The decision to allow a single ECN message to trigger a response to congestion requires a minimum of overhead. In addition, because the gateway does not set the ECN field in *every* arriving packet when the average queue size is too high, the gateway can use probabilistic algorithms to inform particular sources of congestion [FJ93]. Because the probability that a connection is notified of congestion is proportional to that connection's share of the bandwidth at the congested gateway, these probabilistic algorithms reduce global synchronization and improve fairness.
- TCP should react to an ECN at most once per roundtrip time. The TCP source should ignore succeeding ECNs if the source has reacted to a previous ECN or to a dropped packet in the last roundtrip time. This also means that if, immediately after reacting to an ECN, the TCP source receives three duplicate ACKs indicating a dropped packet, the TCP source should not repeat the reduction of the congestion window; the packet was probably dropped before the source reduced its window in response to the ECN.
- TCP should follow the existing algorithms for sending data packets in response to incoming ACKs. The response to an ECN does not trigger the sending of any new (or retransmitted) data packets.
- TCP should follow the normal procedure after the timeout of a retransmit timer, regardless of a previous response to an ECN. That is, after a retransmit timer timeout, the TCP source should slow-start and retransmit the dropped packet.

5 Implementing ECN in our simulator

In this section we describe the implementation of TCP's response to ECN in our simulator. This implementation of ECN was made to a version of TCP that incorporates the Fast Recovery congestion control algorithm in Reno TCP (4.3-reno BSD TCP), as well as the Slow-Start, Congestion Avoidance, and Fast Retransmit algorithms in the earlier Tahoe TCP (4.3-tahoe BSD TCP [J88, S94]).

For the simulations in this paper the RED gateways were given an option to set the ECN bit in the packet header, rather than dropping the packet, as an indication of congestion when the buffer had not yet overflowed. When the TCP receiver receives a data packet with the ECN bit set in the packet header, the receiver sets the ECN bit in the next outgoing ACK packet.

First we briefly describe the Slow-Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery algorithms in TCP. There are two phases to the window-adjustment algorithm. The connection begins in slow-start phase, and the current congestion window *cwnd* is doubled each roundtrip time until the congestion window reaches the slow-start threshold *ssthresh*. Then the congestion-avoidance phase is entered, and the congestion window is increased by roughly one packet each roundtrip time. The congestion window is never allowed to increase to more than the receiver's advertised window, which we refer to as the "maximum window".

In addition to using retransmit timers to detect lost packets, the source uses the *Fast Retransmit* procedure to discover a packet loss. If three duplicate ACK packets are received acknowledging a previously-acknowledged data packet, the source infers that a packet has been dropped.

In the Tahoe version of TCP, the source reacts to a packet loss by setting the slow start threshold to half the congestion window, decreasing the congestion window to one packet, and entering the slow-start phase. In contrast, with Reno's Fast Recovery algorithm the TCP source does not slow-start after inferring that a packet has been dropped. Instead, the TCP source effectively waits half a round trip time and halves the congestion window. The source retransmits the dropped packet and uses incoming duplicate ACKs to clock additional outgoing packets. The Fast Recovery algorithm is described in more detail in [S94].[1]

[1] The description of the Fast Recovery algorithm in [S94] is quite good, and is the only complete publicly-available description of this algorithm that we are aware of. However, we have a caution to readers. First, the earlier printings (prior to the 4th) do not distinguish between the Fast Retransmit and the Fast Recovery algorithms. Second, in the description of the window increase algorithm in the congestion

Following the guidelines in the previous section, upon receiving an ECN message (e.g., a Source Quench message, or an ACK packet with the ECN bit set) at time *t* when no responses to congestion have been made in roughly the last roundtrip time, the TCP source halves both the congestion window *cwnd* and the slow-start threshold *ssthresh*. Because there is no loss of incoming ACKs to clock outgoing packets and no need for a short pause to recover from severe short-term congestion, the TCP source doesn't slow-start. The TCP source doesn't respond to succeeding ECNs until all packets outstanding at time *t* have been acked.

After receiving three duplicate ACKs at time *t* when no responses to congestion have been made in roughly the last roundtrip time, the TCP source follows the Fast Retransmit and Fast Recovery procedures described above. The source won't respond to an ECN or to another set of three duplicate ACKs until all packets outstanding at time *t* have been acked. ([F94] discusses some of the problems that result if the Fast Retransmit procedure is invoked more than once for one window of data.)

After receiving three duplicate ACKs soon after responding to an ECN (e.g., when some of the packets outstanding at the time of the response to the ECN have not yet been ACKed), the source doesn't reduce *ssthresh* or *cwnd*, since that has recently been done. The source retransmits the dropped packet. After this, the source follows Reno's Fast Recovery procedure, using incoming duplicate ACKs to clock outgoing packets.

6 Simulation results of ECN in LANs

This section discusses the results of simulations of TCP with ECN in local-area networks. The simulation scenario consists of five bulk-data TCP connections and ten telnet connections in a LAN with one congested gateway. We compare several sets of simulations. The first set uses Drop-Tail gateways, and the second set uses RED gateways that rely on packet drops for the notification of congestion. The third set of simulations uses ECN-capable RED gateways and TCP implementations.

The simulations show that for the networks with ECN, the throughput of the bulk-data connections is high, and the telnet packet delay is low, regardless of the buffer size, the TCP clock granularity, the TCP window size, or the RED gateway variation. Thus, the simulations with ECN are robust, delivering essentially optimal performance over a wide range of conditions.

For the simulations of RED gateways without ECN, the results depend on the simulation parameters. For most of the simulations, the telnet packet delay is less than optimal (though not disastrous). For a few of the simulations with a coarse TCP clock granularity, the aggregate throughput is also low. The telnet packet delay is worst for the simulations of Drop-Tail gateways (without ECN). In general, the simulations show that with ECN the performance is less sensitive to the network parameters, and that the use of ECN can improve both delay and throughput. However, while the simulations show the benefits of using ECN, the simulations show that under proper conditions, it is possible to get reasonable performance without ECN.

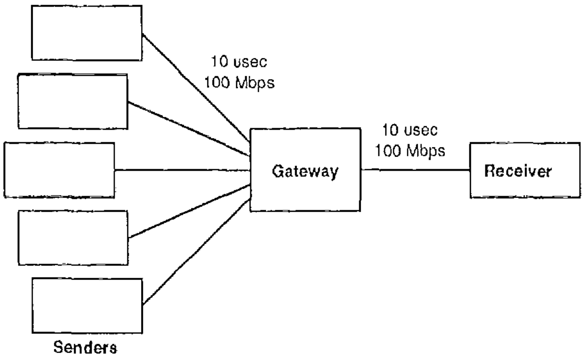

Figure 1: LAN simulation scenario.

6.1 The LAN simulation scenario

The simulation scenario in Figure 1 consists of five bulk-data TCP connections and ten TELNET connections from five sources feeding into a single congested link. The simulations use a range of switch buffer sizes, window sizes, gateway variations, and TCP clock granularities.

The simulations are run for buffer sizes of 60, 120, 180, and 240 kB. In our simulations with RED gateways, the minimum threshold for the average queue size is set to 1/12-th of the buffer size, and the maximum threshold is set to three times the minimum threshold. The RED gateways drop all arriving packets when the average queue size exceeds the maximum threshold.

The TCP in these simulations uses Reno-style congestion control, with Slow-Start, Fast Retransmit, and Fast Recovery, modified as described in the previous section to respond to ECN. One set of simulations uses a TCP clock granularity set to 0.1 msec, for fairly optimal performance in the simulations without ECN. Another set of simulations uses a TCP clock granularity set to 100 msec, which is closer to values of 500 msec used

avoidance phase, the earlier printings say that the congestion window is incremented by 1/*cwnd* plus a small fraction of the segment size each time an ACK is received. The inclusion of the "small fraction of the segment size" is an error in the 4.3 Reno and 4.4 BSD implementations, and should not be emulated in future TCP implementations.

by many current TCP implementations. The maximum TCP window ranges from 8 kB to 64 kB.

The default parameters are set so that for the first packet from a TCP connection, before measurements have been made of the roundtrip time, the retransmit timer is set to three seconds, regardless of the TCP clock granularity. For the simulations without ECN, the worst-case telnet delay is determined mainly by this initial value for the retransmit timer.

The telnet connections send 40-byte packets at random intervals drawn from the tcplib distribution [DJ91]. The ten telnet connections together send several hundred 40-byte data packets in one 15-second simulation. The bulk-data connections send 1000-byte data packets. The start times for the bulk-data connections are staggered over the first second of the 15-second simulation.

In Figures 2 and 3, the graphs illustrate the throughput and delay performance in the simulations. The three columns in Figures 2 and 3 show simulations with Drop-Tail gateways, RED gateways without ECN, and RED gateways with ECN, respectively. The graphs in the top row show the effective throughput of the five bulk-data connections, as a fraction of the total available bandwidth in bits per second. The second row of graphs show the bulk-data connection that receives the smallest throughput over the 15-second simulation.

The third row of graphs shows the telnet packet with the highest one-way telnet delay, in seconds. This one-way delay is the delay from the first time that the packet is transmitted by the source, until the packet is received by the receiver. The fourth row shows the average one-way telnet delay. The fifth row shows the fraction of the telnet packets that have a one-way delay greater than 100 msec. (A roundtrip packet delay greater than 100 msec is likely to be noticeable to the telnet user.) Given the queue sizes and propagation delays in these simulations, a one-way packet delay greater than 100 msec is only possible for a packet that is dropped at the gateway.

For each graph the x-axis shows the switch buffer size in kB. The four lines in each graph correspond to four different simulation sets, with 8 kB and 64 kB maximum TCP windows, and with both byte-based and packet-based gateways. For the (somewhat unrealistic) byte-based gateways, the queue size is measured in bytes rather than in packets, so that a small 40-byte TELNET packet is less likely to arrive to a full buffer than is a larger 1000-byte FTP packet. For the packet-based gateways, the queue size is measured in packets. With packet-based RED gateways the gateway's decision to drop or mark a packet is independent of that packet's size in bytes, while with byte-based RED gateways the probability that a packet is dropped (or marked) is proportional to that packet's size in bytes.

We ran five simulations for each set of parameters. The result of each simulations is marked on the graph, and the lines show the averages of the five simulations.

6.2 Results for LAN simulations

As Figures 2 and 3 show, for all of the simulations of RED with ECN the effective throughput is high and the telnet packet delay is low. Without ECN the network performance can be significantly affected by the TCP clock granularity, by the level of congestion (as a function of the TCP maximum window), and by whether the RED gateway is using a dropping policy that is sensitive to packet size.

For the simulations in Figure 2 with the TCP clock granularity set to 0.1 msec, the throughput is high for all three sets of simulations, Drop-Tail, RED without ECN, and RED with ECN. However, for the simulations of packet-based Drop-Tail and RED gateways without ECN, telnet packets are occasionally dropped, leading to occasional telnet packets with high delay.

For the simulations in Figure 3 of RED without ECN with smaller switch buffers, a TCP clock granularity of 100 msec, and small TCP windows, the throughput of the bulk-data connections suffers. Because of the coarse TCP clock granularity, a number of connections could be waiting for a retransmit timer to expire, resulting in link idle time.

For the simulations with Drop-Tail gateways and small TCP windows, packets should never be dropped at the gateway. For the simulations with Drop-Tail gateways, larger TCP windows, and a TCP clock granularity of 100 msec, the overall throughput is high, but the graph of the "Smallest Bulk-Data Throughput" shows that there is some unfairness; with smaller buffer sizes, the throughput of the smallest of the five bulk-data connections is less than optimal. Note that a Drop-Tail gateway with a certain buffer size cannot be directly compared to a RED gateway with the same buffer size; the more appropriate comparison is between a Drop-Tail gateway and a RED gateway with a similar average queue size.

These simulations show that the delay for small telnet packets is much lower with byte-based gateways. These byte-based gateways might be easy to implement in our simulator, but they are not typical of current gateways. In addition, for low-throughput delay-sensitive interactive traffic where the size of individual packets is similar to the size of bulk-data packets, byte-based gateways would not improve the performance of the interactive traffic.

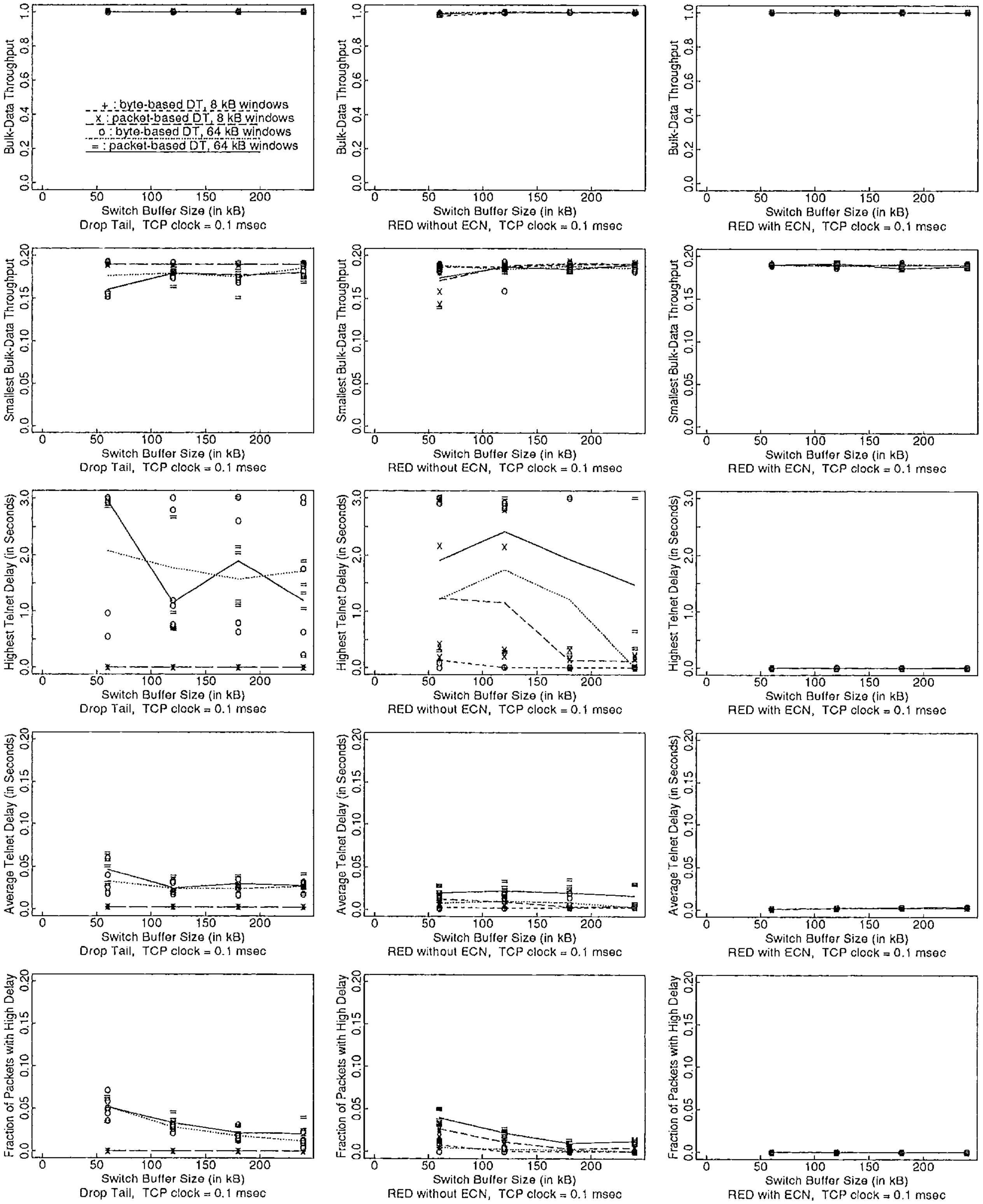

Figure 2: LAN simulations with a 0.1 msec TCP clock.

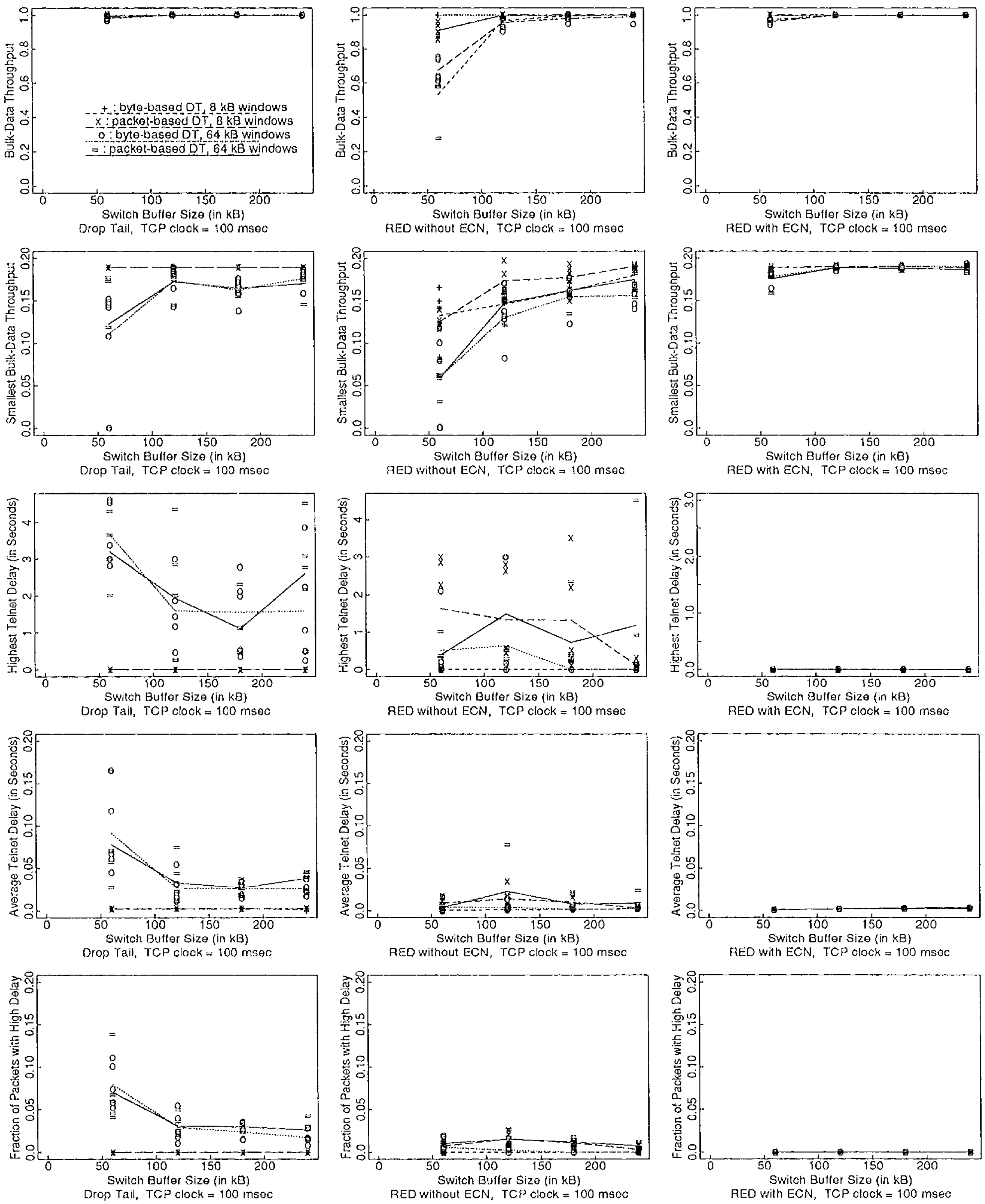

Figure 3: LAN simulations with a 100 msec TCP clock.

7 Simulation results of ECN in WANS

This section gives results from simulations of ECN and non-ECN gateways in a wide-area environment. The throughput for the bulk-data traffic is similar in the simulations with Drop-Tail gateways, RED gateways with ECN, or RED gateways without ECN. However, the packet delay for low-bandwidth telnet traffic is significantly lower for the simulations with ECN.

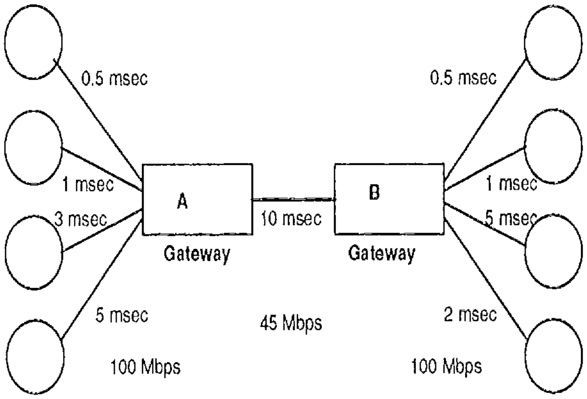

Figure 4: Simulation scenario for wide-area traffic.

7.1 WAN simulation scenario

The simulation network in Figure 4 has two-way traffic consisting of sixteen TELNET connections in each direction, occasional FTP connections with a limited amount of data to transfer (100-300 packets), and a number of bulk-data connections with an unlimited amount of data to transfer. For the simulations in Figure 5 of a moderate traffic load, there are four bulk-data TCP connections in each direction. For the simulations in Figure 6 of a heavy traffic load, there are twenty bulk-data TCP connections in each direction. This is not intended to be a realistic scenario; this is simply intended to illustrate that the performance of non-ECN gateways depends in part on the level of congestion.

The roundtrip propagation delays range from 21 to 40 msec. Given 1000-byte packets, the bandwidth-delay product for a single connection ranges from 118 to 225 packets. Each simulation is run for 10 seconds.

The RED gateways in this simulation have the same minimum and maximum thresholds for the average queue size as described in the previous section. However, as is appropriate for gateways intended for wide-area traffic, the time constant for the low-pass filter used to calculate the average queue size has been increased.[2]

The graphs in Figures 5 and 6 show three sets of simulations, as in the previous section. The first row of graphs shows the aggregate throughput of the bulk-data connections that go from a source on the left to a receiver on the right, and the second row of graphs shows the bulk-data connection from these that has the smallest throughput. The third and fourth rows of graphs show the worst-case and the average telnet packet delay for telnet packets going from left to right. The fifth row of graphs shows the average throughput of the shorter data connections with limited data to send.

Note that for several of the graphs, the y-axis in Figure 6 is different from that in Figure 5.

The simulations use TCP connections with a 100 msec TCP clock granularity. The simulations with TCP connections with a 0.1 msec TCP clock granularity give similar results, and are not shown here.

7.2 Results for WAN simulations

The results of the WAN simulations show that, while throughput is similar in all three sets of simulations, the packet delay for low-bandwidth interactive traffic is much better for the simulations with ECN.

The results are fairly similar for the two simulation sets without ECN, those with Drop Tail gateways and those with RED gateways. However, it would be possible to construct WAN scenarios, like the LAN simulations earlier in this paper, that illustrate some of the advantages of RED gateways (with or without ECN) over Drop-Tail gateways [FJ93, VS94]. This could probably be the case, for example, for scenarios that exhibit either global synchronization and/or unfairness with Drop-Tail gateways.

Note that, for both Drop Tail gateways and RED gateways without ECN, telnet packet delay is worse with a larger number of bulk-data connections or with TCP connections with larger windows. In these simulations with increased demand, an increased number of packet drops is required from a non-ECN gateway to control congestion.

For the Drop-Tail gateways, the throughput and delay performance is particularly good for the simulations with a smaller number of bulk-data connections, smaller windows, and byte-based gateways. In this case the level of congestion is fairly low.

For wide-area traffic, even for those cases where the source has to wait for a retransmit timer to detect a lost packet, a TCP clock granularity of 100 msec is not a major problem, and the additional delay to wait for a retransmit timer has a less significant impact on perfor-

[2] The "weight" used by the exponential weighted moving average filter to calculate the average queue size has been decreased from 0.002 to 0.001 [FJ93].

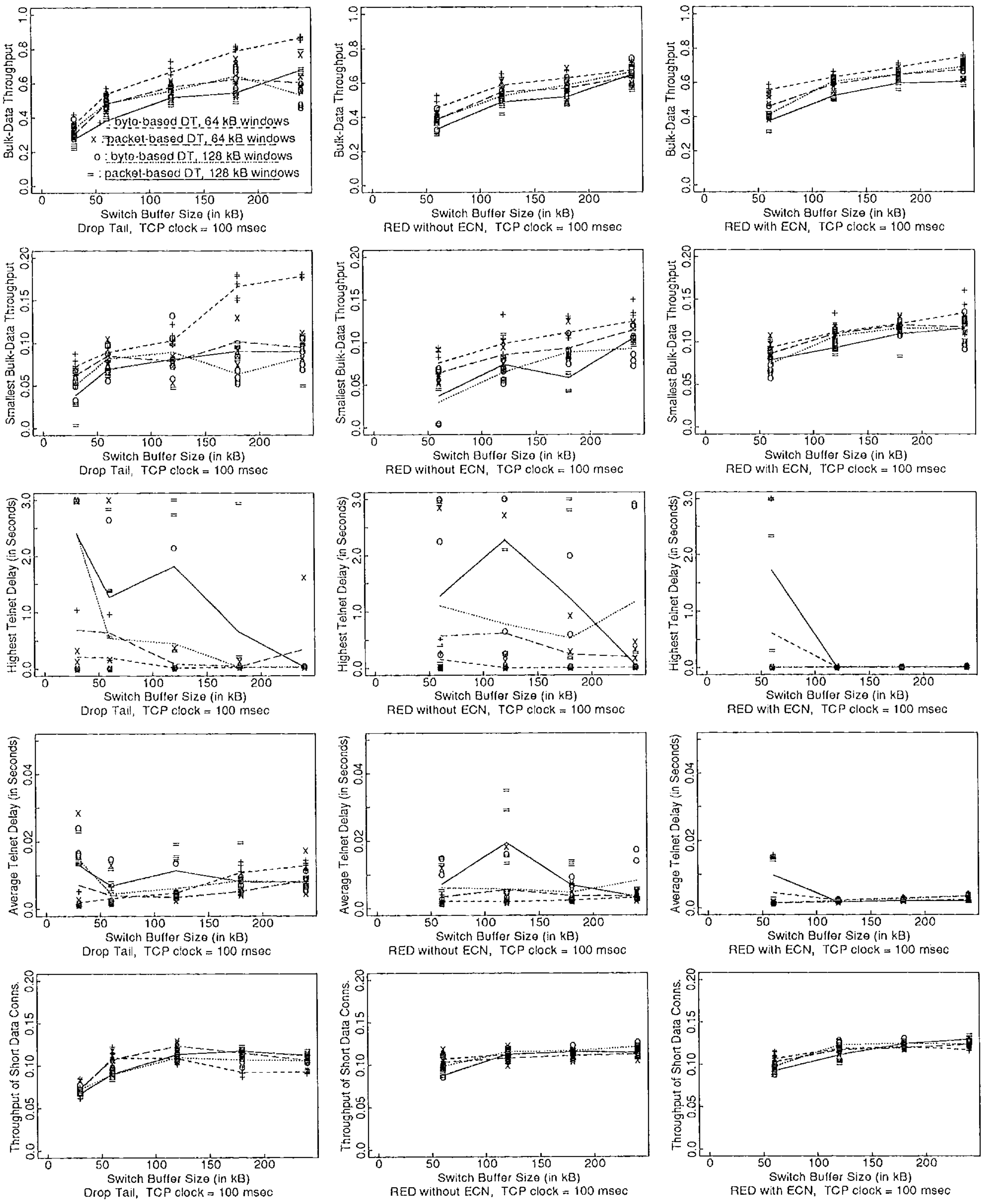

Figure 5: WAN simulations with a 100 msec TCP clock and a moderate number of bulk-data connections.

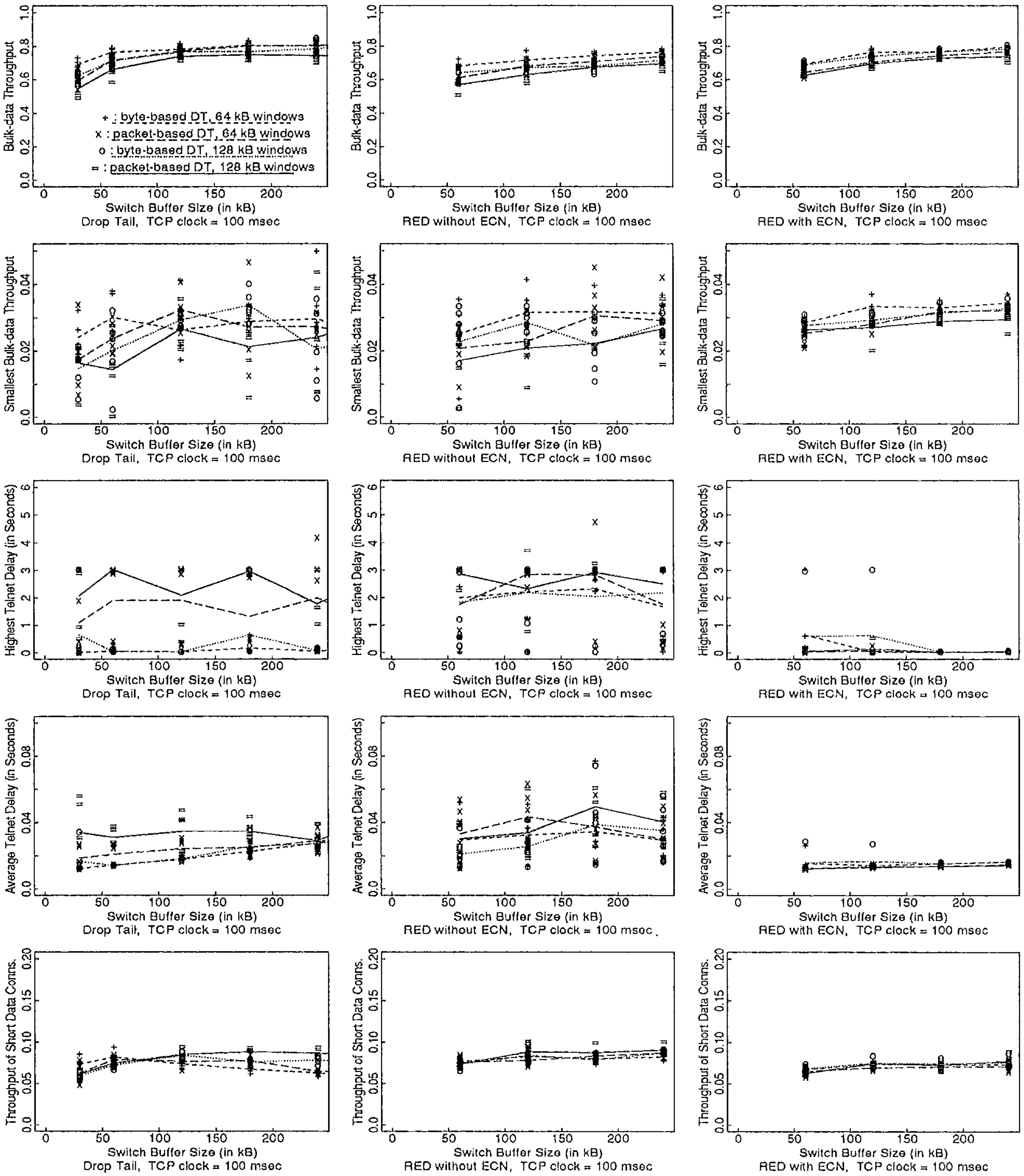

Figure 6: WAN simulations with a 100 msec TCP clock and a large number of bulk-data connections.

mance. In this case, the simulations with 100 msec TCP clocks are similar to those with 0.1 msec TCP clocks.

8 Advantages and disadvantages of ECN

In this section we discuss further some of the advantages and disadvantages of ECN, both for the current Internet environment and for various proposed modifications to that environment. As already discussed in the introduction, the advantages of ECN include a reduction in packet drops and packet delay for low-bandwidth delay-sensitive TCP traffic and a prompt notification to end nodes of network congestion.

8.1 Disadvantages of ECN

Two disadvantages or potential problems with ECN concern non-compliant ECN connections and the potential loss of ECN messages in the network.

A non-compliant TCP connection could set the ECN field to indicate that it was ECN-capable, and then ignore ECN notifications. Non-compliant connections could also ignore Source Quench messages. However, for a network that uses only packet drops for congestion notification, a non-compliant connection could also refrain from making appropriate window decreases in response to packet drops. A non-compliant connection interested in reliable delivery cannot ignore packet drops completely, but in the absence of monitoring and controls, a non-compliant connection could cause congestion problems in either an ECN or a non-ECN environment.

A problem with ECN messages that has no counterpart with packet drops is that an ECN message (e.g., a Source Quench message, or a TCP ACK packet with the ECN field set) could be dropped by the network, and the congestion notification could fail to reach the end node. Thus, neither Source Quench messages nor the use of ECN fields in packet headers can guarantee that the TCP source will receive each notification of congestion. However, with RED gateways the gateway does not rely on the source to respond to each congestion notification. The gateway will continue to set the ECN field in randomly-chosen packets as long as congestion persists at the gateway. In addition, a gateway implementing RED algorithms is particularly unlikely to drop a high fraction of packets. The occasional loss of an ECN message should not be a serious problem.

8.2 ECN with other versions of TCP

The simulations in this paper use Reno-style TCP algorithms, modified as suggested in this paper to respond to ECN messages. In this section we discuss the implications of ECN for some proposed modifications to TCP.

One proposed modification to TCP, the addition of Selective Acknowledgements, would reduce TCP's reliance on the retransmit timer when multiple packets are dropped in one roundtrip time. While this would improve somewhat the robustness of the response of bulk-data TCP connections to packet drops as an indication of congestion, this would not reduce the ability of ECN mechanisms to reduce packet delay for low-bandwidth delay-sensitive TCP connections.

There are several proposed modifications of TCP [BOP94, WC91, WC92] where the TCP source would detect incipient congestion from the traffic dynamics of the connection's traffic stream. For a network such as the Internet without admissions control, a TCP source cannot completely prevent the network from dropping its packets. Nevertheless, with improved detection by TCP of incipient congestion, the TCP source could reduce congestion and consequently reduce the number of packets dropped.

However, the possibility of improved congestion detection by the end nodes does not eliminate the need for improved congestion detection at the gateways. In particular, in the absence of prior information about the fixed propagation delay of a path, it is not possible for end nodes to distinguish between propagation delay and persistent queueing delay (that is, queueing delay that existed when the connection was started, and that has persisted for the duration of the connection). As the delay-bandwidth product of network connections increases, the buffer capacity as the gateways will also increase, and it will become increasingly important that these large buffers not have persistent large queues.

The detection of persistent large queues is appropriately done in the gateway itself, and the notification to the end nodes of this congestion requires either packet drops or ECN. Thus, while proposed modifications of TCP might decrease the number of packets dropped by the gateways as a result of buffer overflows, these modifications do not eliminate the need for some form of congestion notification from the gateways to the end nodes.

It seems unlikely to us that modifications of TCP with newer end-to-end congestion avoidance mechanisms would significantly reduce the usefulness of ECN.

8.3 ECN with proposed modifications to router scheduling algorithms

Other possible changes in future networks such as the introduction of priority-based scheduling or of Fair-Queueing-based scheduling at the gateways could complicate the traffic dynamics of TCP traffic in future networks.

As an example of possible changes, the proposed IPng header [H94] contains a Flow Label with a 4-bit Traffic Class field identifying seven classes of flow-controlled (e.g., TCP) traffic, including classes for unattended data transfer such as email, attended data transfer such as FTP, interactive traffic such as telnet, and interactive control traffic such as SNMP. These Flow Labels could facilitate the use by gateways of separate queues or scheduling algorithms for the different traffic classes [BCS94, F93].

The use of a separate traffic class for interactive traffic would reduce packet drops for this traffic during periods of low demand from the interactive traffic. However, for both attended data transfer and interactive traffic, some notification of congestion from the gateways will continue to be required. Thus, the use of ECN mechanisms would still improve the promptness and robustness of congestion notification for data transfer connections, and reduce unnecessary packet drops for some connections in the interactive traffic class.

While it is difficult to predict the future internet congestion control environment, and the precise advantages and disadvantages of ECN mechanisms in that environment, the basic advantages of ECN mechanisms of reducing unnecessary packet drops for low-bandwidth interactive traffic and of speeding the notification of congestion to bulk data connections seem likely to remain. Further, in a heterogeneous internet some routers might find ECN mechanisms useful for ECN-capable TCP connections, without ECN mechanisms being required for all routers.

9 Implementation issues

9.1 Source Quench messages vs. ECN fields in packet headers

The guidelines in the paper for the response of TCP sources to ECN messages are orthogonal to the question of the mechanisms for delivering this congestion notification to the source. In this section we compare two such mechanisms: Source Quench messages, and ECN fields in packet headers.

One advantage of using ECN fields in IP packet headers is that the ECN field places no additional load on the network in times of congestion. If a Drop-Tail router were to send a Source Quench message for every packet dropped at the router, the overhead of this Source Quench traffic in times of congestion could be a significant problem. However, the use of intelligent gateway mechanisms such as those in RED gateways would limit the overhead of the Source Quench traffic. In addition, in particular environments such as LANs where the overhead of Source Quench messages would be limited by the small number of hops on the return paths, the use of Source Quench messages could be quite justifiable.

An advantage of Source Quench messages (or other forms of "backward ECN") over "forward" ECN mechanisms such as ECN fields in packet headers is that with Source Quench messages, the notification of congestion arrives at the source as quickly as possible. Although we have not explored the dynamics of backward ECN mechanisms such as Source Quench in our simulations, this promptness in the notification of congestion would be an advantage in congestion control schemes.

Another practical advantage of Source Quench messages is that Source Quench messages could be used immediately in appropriate networks, given modified TCP implementations with the response to Source Quench messages proposed in this paper. The use of an ECN field in TCP/IP networks would depend on the presence of the ECN field in IPng headers. Even if the evidence were sufficiently compelling to motivate such an addition, it could be years before such headers were fully deployed.

As already mentioned, both Source Quench messages and TCP acknowledgement packets with the ECN field set could be dropped by a gateway, with the result that the TCP source node might never receive the congestion notification. If the network drops a data packet that has the ECN bit set, the TCP source will still infer congestion when it detects the dropped data packet. However, if the network drops an ACK packet that has the ECN bit set, and the TCP source later receives an ACK packet without the ECN bit set that acknowledges a subsequent data packet, then the TCP source will never receive the notification of congestion. Thus, neither Source Quench messages nor ECN fields ensure reliable delivery of congestion notification.

9.2 Incremental deployment of ECN-capable TCP

One concern regarding ECN fields in IP packet headers is with the incremental deployment of ECN-capable TCP implementations. If a gateway sets the ECN field in a packet header, how does the gateway know that the transport-level protocol is capable of responding appropriately? For an ECN field with two bits, one bit could be used to indicate whether or not the transport protocol could respond to ECN, and the second bit could be used to indicate congestion. For an ECN field with only one bit, these two functions would have to be combined with a single bit.

For an ECN field that consists of a single bit, one value, say the "OFF" value, could indicate "ECN-capable Transport Protocol", and the "ON" value, could indicate "either Non-ECN Transport Protocol or Congestion Notification". For packets from transport-level sources that are not capable of ECN response, the ECN field could be set to ON (the default value). For packets from transport-level sources capable of ECN response, the ECN field could be set to OFF. Non-ECN-capable gateways would ignore the ECN field, simply dropping packets to indicate congestion. ECN-capable gateways, seeing packets with the ECN field OFF, would know that the corresponding transport protocol was ECN-capable, and could set the ECN field to ON for appropriate packets during times of congestion.

For arriving packets with the ECN field ON, the ECN-capable gateway would not know whether that packet came from a non-ECN-capable transport protocol, or whether the ECN field had been set by a previous gateway. In either case, if the gateway wanted to notify a TCP source about congestion, the gateway would drop the packet. This method of incremental deployment with a single-bit ECN field would mean that for packets from ECN-capable transport protocols, that packet would be dropped by a second router attempting to set the ECN bit. This can only happen for packets that pass through multiple congested gateways, where both gateways choose that packet for notifying the source of congestion.

Thus, ECN fields could be deployed in a heterogeneous environment where only some of the TCP implementations were ECN-capable, and where only some of the routers have procedures for setting the ECN field.

Note that this description of a single-bit ECN field assumes a TCP connection with one-way traffic, where all of the data packets travel in one direction and ACK packets travel in the other. For a TCP connection with two-way data transfer, a second bit would be needed in the ECN field, or some additional mechanism would be needed to return an indication of congestion from the receiver to the source.

A concern with incremental deployment also exists for Source Quench messages. If a gateway wants to use Source Quench messages, the gateway would not know whether the TCP implementation was a old implementation with a fairly drastic response to Source Quench messages, or a newer implementation with the responses recommended in this paper. However, in this case the problem with older implementations would not be that they would ignore Source Quench messages entirely, but that they would back off for too long in response to a Source Quench message. This would be an incentive for users to upgrade to newer TCP implementations, given an environment with routers using Source Quench messages.

9.3 Improving the TCP clock granularity?

In the simulations in this section, the advantages of ECN mechanisms in TCP/IP networks are most pronounced for LAN traffic with TCP implementations limited by a coarse-granularity TCP clock. This coarse-grained TCP clock limits the granularity of TCP's measurements of current roundtrip times, used to determine the value for the retransmit timer. The simulation results in this paper, along with other results [RF94], argue in favor of improving the granularity of TCP clocks.

Unfortunately, even if there were no hardware considerations, and TCP designers could set the TCP clock granularity to the optimal value, it is not obvious what that optimal value should be. It seems clear (to us) that a TCP clock granularity of 100 msec or smaller would be more appropriate for current networks than the TCP clock granularity of 500 msec in many current TCP implementations. Nevertheless, the current algorithms for setting the TCP retransmit timer assume a somewhat coarse granularity for the TCP clock relative to the measured roundtrip times [J88], and the range of roundtrip times in current networks is quite wide. Significantly changing the TCP clock granularity would require a rethinking of the algorithms for setting the TCP retransmit timer. The addition of ECN mechanisms to TCP/IP networks has the advantage of reducing the importance of the TCP clock granularity, thereby increasing the general robustness of the network.

9.4 TCP over ATM

The investigation of ECN in this paper concerns only TCP/IP networks; we are not considering the various proposals for ECN in ATM networks. In particular, we are not considering the congestion control strategies that might be used inside the ATM networks.

We do, however, consider a scenario of TCP/IP traffic where part or all of the path might consist of ATM networks. The ATM network needs mechanisms to inform TCP connections of congestion. At the moment, the only viable mechanism is for the ATM network to drop TCP/IP packets, either inside or at the boundaries of the ATM network.

If IP-level ECN mechanisms (e.g., Source Quench, ECN fields in IP packet headers) were available for ATM networks to inform TCP sources about congestion, the ATM networks could invoke these mechanisms at the boundary of the ATM networks where frame segmentation and reassembly occur. For example, for a TCP/IP network where some of the TCP sources were ECN-capable, the ATM boundary router could drop TCP/IP packets to indicate congestion to non-ECN-capable TCP sources, and invoke ECN mechanisms for packets from

ECN-capable TCP sources.

10 Conclusions and Future Work

We have proposed specific guidelines for TCP's response to Source Quench messages or to other ECN mechanisms. We would propose that these guidelines be used to modify TCP's response to Source Quench messages. If TCP implementations had a more clearly-defined response to Source Quench messages, then networks such as ATM LANs could consider whether or not to use Source Quench messages as a controlled notification of congestion to TCP/IP connections traversing that network.

For a wide-area network the overhead of Source Quench messages makes their use problematic. However, the logistical difficulties of adding ECN fields to IP packet headers makes the use of ECN fields problematic as well. The proposed IPng packet header [H94] has no space allocated for an ECN field, and it is not clear if IPng options, with a minimum length of 8 octets, would be an appropriate place for an ECN field.

The simulations in this paper suggest the ECN mechanisms would give a clear, if modest, benefit in TCP/IP networks. However, we see this research as a preliminary investigation of the advantages and disadvantages of ECN mechanisms in TCP/IP networks.

A main advantage of ECN mechanisms is in avoiding unnecessary packet drops, and therefore avoiding unnecessary delay for packets from low-bandwidth delay-sensitive TCP connections. This advantage will be most pronounced in a highly-congested network where a high frequency of packet drops is required to control congestion.

A second advantage of ECN mechanisms is in networks (generally LANs) where the effectiveness of TCP retransmit timers is limited by the coarse granularity of the TCP clock. With ECN, the congestion notification is promptly received by the TCP source, and the connection does not remain idle, waiting for a TCP retransmit timer to expire, after a packet has been dropped. While to some extent the over-coarse granularity of the TCP clock could be corrected, and the TCP retransmit timer algorithms suitably modified, the use of ECN mechanisms, by reducing the number of packet drops, reduces the dependence on the retransmit timer.

One disadvantage of ECN mechanisms discussed earlier in the paper is that ECN messages (e.g., Source Quench messages, or TCP ACK packets with the ECN field set) could be dropped by the network before reaching the TCP source. For a TCP connection, packet drops are a reliable (if sometimes slow) indication of congestion. Preliminary simulations of a wide-area scenario with two-way traffic and multiple congested gateways, some with Drop-Tail gateways and some with ECN-capable RED gateways, do not show performance problems from dropped ECN messages. In addition, the number of dropped ECN messages should be small in a network with ECN mechanisms and RED-style gateways.

11 Acknowledgements

This work benefited from ongoing discussions with Van Jacobson. We also thank Vern Paxson, Allyn Romanow, Curtis Villamizar, and two perceptive anonymous reviewers for much helpful feedback. This work was motivated in part by discussions with members of the Traffic Management Working Group of the ATM Forum.

References

[A94] Almquist, P., "Requirements for IP Routers", Working Draft, IETF, draft-ietf-rreq-iprouters-require-01.txt, April 1994.

[BCS94] Braden, B., Clark, D., and Shenker, S., "Integrated Services in the Internet Architecture: an Overview", Request for Comments (RFC) 1633, IETF, June 1994.

[BP87] Braden, R., and Postel, J., "Requirements for internet gateways," Request for Comments (Standard) RFC 1009, IETF, June 1987.

[BOP94] Brakmo, L.S., O'Malley, S.W., and Peterson, L.L., "TCP Vegas: New Techniques for Congestion Detection and Avoidance", *Proceedings of SIGCOMM 1994*, pp. 24-35.

[DJ91] Danzig, P., and Jamin, S., "tcplib: A Library of TCP Internetwork Traffic Characteristics", *Report CS-SYS-91-01*, Computer Science Department, University of Southern California, 1991. Available via FTP to excalibur.usc.edu as *pub/jamin/tcplib/tcplib.tar.Z.*

[F93] Floyd, S., "Link-sharing and Resource Management Models for Packet Networks", to be submitted to IEEE/ACM Transactions on Networking. Available by anonymous ftp from ftp.ee.lbl.gov in papers/link.ps.Z, September 1993.

[F94] Floyd, S., "TCP and Successive Fast Retransmits", technical report, October 1994. Available by anonymous ftp from ftp.ee.lbl.gov in papers/fastretrans.ps.

[FJ93] Floyd, S., and Jacobson, V., "Random Early Detection Gateways for Congestion Avoidance", *IEEE/ACM Transactions on Networking*, V.1 N.4, August 1993, p. 397-413.

[H94] Hinden, R., "IP Next Generation Overview", IETF internet draft, work in progress, October 1994.

[J88] Jacobson, V., "Congestion Avoidance and Control", *Proceedings of SIGCOMM 1988*, August 1988, pp. 314-329. (An updated version of this paper is available by anonymous ftp from ftp.ee.lbl.gov in congavoid.ps.Z.)

[N94] Newman, P., "Traffic Management for ATM Local Area Networks", *IEEE Communications Magazine*, Vol. 32 No. 8, August 1994, pp. 44-50.

[RF94] Romanow, A., and Floyd, S., "Dynamics of TCP Traffic over ATM Networks", *Proceedings of SIGCOMM 1994*, August 1994, pp. 79-88.

[RJ90] Ramakrishnan, K.K., and Jain, R., "A Binary Feedback Scheme for Congestion Avoidance in Computer Networks", *ACM Transactions on Computer Systems*, Vol. 8 No. 2, pp. 158-181, 1990.

[S94] W. Stevens, *TCP/IP Illustrated*, Volume 1, Addison-Wesley, Reading, MA, 1994.

[VS94] Villamizar, C., and Song, C., "High Performance TCP in ANSNET", submitted to Computer Communications Review, September 1994.

[WC91] Wang, Z., and Crowcroft, J., "A New Congestion Control Scheme: Slow Start and Search (Tri-S)", *ACM Computer Communication Review*, V.21 N.1, January 1991, pp. 32-43.

[WC92] Wang, Z., and Crowcroft, J., "Eliminating Periodic Packet Losses in the 4.3-Tahoe BSD TCP Congestion Control Algorithm", *ACM Computer Communication Review*, V. 22 N. 2, April 1992, pp. 9-16.

[WRM91] Wilder, R., Ramakrishnan, K.K., and Mankin, A., "Dynamics of Congestion Control and Avoidance in Two-Way Traffic in an OSI Testbed", *ACM Computer Communications Review*, V.21, N.2, April 1991, pp.43-58.