Network Working Group
Request for Comments: 2481
Category: Experimental

K. Ramakrishnan
AT&T Labs Research
S. Floyd
LBNL

January 1999

A Proposal to add Explicit Congestion Notification (ECN) to IP

Status of this Memo

This memo defines an Experimental Protocol for the Internet community. It does not specify an Internet standard of any kind. Discussion and suggestions for improvement are requested. Distribution of this memo is unlimited.

Copyright Notice

Abstract

This note describes a proposed addition of ECN (Explicit Congestion Notification) to IP. TCP is currently the dominant transport protocol used in the Internet. We begin by describing TCP's use of packet drops as an indication of congestion. Next we argue that with the addition of active queue management (e.g., RED) to the Internet infrastructure, where routers detect congestion before the queue overflows, routers are no longer limited to packet drops as an indication of congestion. Routers could instead set a Congestion Experienced (CE) bit in the packet header of packets from ECN-capable transport protocols. We describe when the CE bit would be set in the routers, and describe what modifications would be needed to TCP to make it ECN-capable. Modifications to other transport protocols (e.g., unreliable unicast or multicast, reliable multicast, other reliable unicast transport protocols) could be considered as those protocols are developed and advance through the standards process.

1. Conventions and Acronyms

The keywords MUST, MUST NOT, REQUIRED, SHALL, SHALL NOT, SHOULD, SHOULD NOT, RECOMMENDED, MAY, and OPTIONAL, when they appear in this document, are to be interpreted as described in [B97].

2. Introduction

TCP's congestion control and avoidance algorithms are based on the notion that the network is a black-box [Jacobson88, Jacobson90]. The network's state of congestion or otherwise is determined by end-systems probing for the network state, by gradually increasing the load on the network (by increasing the window of packets that are outstanding in the network) until the network becomes congested and a packet is lost. Treating the network as a "black-box" and treating loss as an indication of congestion in the network is appropriate for pure best-effort data carried by TCP which has little or no sensitivity to delay or loss of individual packets. In addition, TCP's congestion management algorithms have techniques built-in (such as Fast Retransmit and Fast Recovery) to minimize the impact of losses from a throughput perspective.

However, these mechanisms are not intended to help applications that are in fact sensitive to the delay or loss of one or more individual packets. Interactive traffic such as telnet, web-browsing, and transfer of audio and video data can be sensitive to packet losses (using an unreliable data delivery transport such as UDP) or to the increased latency of the packet caused by the need to retransmit the packet after a loss (for reliable data delivery such as TCP).

Since TCP determines the appropriate congestion window to use by gradually increasing the window size until it experiences a dropped packet, this causes the queues at the bottleneck router to build up. With most packet drop policies at the router that are not sensitive to the load placed by each individual flow, this means that some of the packets of latency-sensitive flows are going to be dropped. Active queue management mechanisms detect congestion before the queue overflows, and provide an indication of this congestion to the end nodes. The advantages of active queue management are discussed in RFC 2309 [RFC2309]. Active queue management avoids some of the bad properties of dropping on queue overflow, including the undesirable synchronization of loss across multiple flows. More importantly, active queue management means that transport protocols with congestion control (e.g., TCP) do not have to rely on buffer overflow as the only indication of congestion. This can reduce unnecessary queueing delay for all traffic sharing that queue.

Active queue management mechanisms may use one of several methods for indicating congestion to end-nodes. One is to use packet drops, as is currently done. However, active queue management allows the router to separate policies of queueing or dropping packets from the policies for indicating congestion. Thus, active queue management allows routers to use the Congestion Experienced (CE) bit in a packet header as an indication of congestion, instead of relying solely on packet drops.

3. Assumptions and General Principles

In this section, we describe some of the important design principles and assumptions that guided the design choices in this proposal.

(1) Congestion may persist over different time-scales. The time scales that we are concerned with are congestion events that may last longer than a round-trip time.
(2) The number of packets in an individual flow (e.g., TCP connection or an exchange using UDP) may range from a small number of packets to quite a large number. We are interested in managing the congestion caused by flows that send enough packets so that they are still active when network feedback reaches them.
(3) New mechanisms for congestion control and avoidance need to co-exist and cooperate with existing mechanisms for congestion control. In particular, new mechanisms have to co-exist with TCP's current methods of adapting to congestion and with routers' current practice of dropping packets in periods of congestion.
(4) Because ECN is likely to be adopted gradually, accommodating migration is essential. Some routers may still only drop packets to indicate congestion, and some end-systems may not be ECN-capable. The most viable strategy is one that accommodates incremental deployment without having to resort to "islands" of ECN-capable and non-ECN-capable environments.
(5) Asymmetric routing is likely to be a normal occurrence in the Internet. The path (sequence of links and routers) followed by data packets may be different from the path followed by the acknowledgment packets in the reverse direction.
(6) Many routers process the "regular" headers in IP packets more efficiently than they process the header information in IP options. This suggests keeping congestion experienced information in the regular headers of an IP packet.
(7) It must be recognized that not all end-systems will cooperate in mechanisms for congestion control. However, new mechanisms shouldn't make it easier for TCP applications to disable TCP congestion control. The benefit of lying about participating in new mechanisms such as ECN-capability should be small.

4. Random Early Detection (RED)

Random Early Detection (RED) is a mechanism for active queue management that has been proposed to detect incipient congestion [FJ93], and is currently being deployed in the Internet backbone [RFC2309]. Although RED is meant to be a general mechanism using one of several alternatives for congestion indication, in the current environment of the Internet RED is restricted to using packet drops as a mechanism for congestion indication. RED drops packets based on the average queue length exceeding a threshold, rather than only when the queue overflows. However, when RED drops packets before the queue actually overflows, RED is not forced by memory limitations to discard the packet.

RED could set a Congestion Experienced (CE) bit in the packet header instead of dropping the packet, if such a bit was provided in the IP header and understood by the transport protocol. The use of the CE bit would allow the receiver(s) to receive the packet, avoiding the potential for excessive delays due to retransmissions after packet losses. We use the term 'CE packet' to denote a packet that has the CE bit set.

5. Explicit Congestion Notification in IP

We propose that the Internet provide a congestion indication for incipient congestion (as in RED and earlier work [RJ90]) where the notification can sometimes be through marking packets rather than dropping them. This would require an ECN field in the IP header with two bits. The ECN-Capable Transport (ECT) bit would be set by the data sender to indicate that the end-points of the transport protocol are ECN-capable. The CE bit would be set by the router to indicate congestion to the end nodes. Routers that have a packet arriving at a full queue would drop the packet, just as they do now.

Bits 6 and 7 in the IPv4 TOS octet are designated as the ECN field. Bit 6 is designated as the ECT bit, and bit 7 is designated as the CE bit. The IPv4 TOS octet corresponds to the Traffic Class octet in IPv6. The definitions for the IPv4 TOS octet [RFC791] and the IPv6 Traffic Class octet are intended to be superseded by the DS (Differentiated Services) Field [DIFFSERV]. Bits 6 and 7 are listed in [DIFFSERV] as Currently Unused. Section 19 gives a brief history of the TOS octet.

Because of the unstable history of the TOS octet, the use of the ECN field as specified in this document cannot be guaranteed to be backwards compatible with all past uses of these two bits. The potential dangers of this lack of backwards compatibility are discussed in Section 19.

Upon the receipt by an ECN-Capable transport of a single CE packet, the congestion control algorithms followed at the end-systems MUST be essentially the same as the congestion control response to a *single* dropped packet. For example, for ECN-Capable TCP the source TCP is required to halve its congestion window for any window of data containing either a packet drop or an ECN indication. However, we would like to point out some notable exceptions in the reaction of the source TCP, related to following the shorter-time-scale details of particular implementations of TCP. For TCP's response to an ECN indication, we do not recommend such behavior as the slow-start of Tahoe TCP in response to a packet drop, or Reno TCP's wait of roughly half a round-trip time during Fast Recovery.

One reason for requiring that the congestion-control response to the CE packet be essentially the same as the response to a dropped packet is to accommodate the incremental deployment of ECN in both end-systems and in routers. Some routers may drop ECN-Capable packets (e.g., using the same RED policies for congestion detection) while other routers set the CE bit, for equivalent levels of congestion. Similarly, a router might drop a non-ECN-Capable packet but set the CE bit in an ECN-Capable packet, for equivalent levels of congestion. Different congestion control responses to a CE bit indication and to a packet drop could result in unfair treatment for different flows.

An additional requirement is that the end-systems should react to congestion at most once per window of data (i.e., at most once per roundtrip time), to avoid reacting multiple times to multiple indications of congestion within a roundtrip time.

For a router, the CE bit of an ECN-Capable packet should only be set if the router would otherwise have dropped the packet as an indication of congestion to the end nodes. When the router's buffer is not yet full and the router is prepared to drop a packet to inform end nodes of incipient congestion, the router should first check to see if the ECT bit is set in that packet's IP header. If so, then instead of dropping the packet, the router MAY instead set the CE bit in the IP header.

An environment where all end nodes were ECN-Capable could allow new criteria to be developed for setting the CE bit, and new congestion control mechanisms for end-node reaction to CE packets. However, this is a research issue, and as such is not addressed in this document.

When a CE packet is received by a router, the CE bit is left unchanged, and the packet transmitted as usual. When severe congestion has occurred and the router's queue is full, then the router has no choice but to drop some packet when a new packet arrives. We anticipate that such packet losses will become relatively infrequent when a majority of end-systems become ECN-Capable and participate in TCP or other compatible congestion control mechanisms. In an adequately-provisioned network in such an ECN-Capable environment, packet losses should occur primarily during transients or in the presence of non-cooperating sources.

We expect that routers will set the CE bit in response to incipient congestion as indicated by the average queue size, using the RED algorithms suggested in [FJ93, RFC2309]. To the best of our knowledge, this is the only proposal currently under discussion in the IETF for routers to drop packets proactively, before the buffer overflows. However, this document does not attempt to specify a particular mechanism for active queue management, leaving that endeavor, if needed, to other areas of the IETF. While ECN is inextricably tied up with active queue management at the router, the reverse does not hold; active queue management mechanisms have been developed and deployed independently from ECN, using packet drops as indications of congestion in the absence of ECN in the IP architecture.

6. Support from the Transport Protocol

ECN requires support from the transport protocol, in addition to the functionality given by the ECN field in the IP packet header. The transport protocol might require negotiation between the endpoints during setup to determine that all of the endpoints are ECN-capable, so that the sender can set the ECT bit in transmitted packets. Second, the transport protocol must be capable of reacting appropriately to the receipt of CE packets. This reaction could be in the form of the data receiver informing the data sender of the received CE packet (e.g., TCP), of the data receiver unsubscribing to a layered multicast group (e.g., RLM [MJV96]), or of some other action that ultimately reduces the arrival rate of that flow to that receiver.

This document only addresses the addition of ECN Capability to TCP, leaving issues of ECN and other transport protocols to further research. For TCP, ECN requires three new mechanisms: negotiation between the endpoints during setup to determine if they are both ECN-capable; an ECN-Echo flag in the TCP header so that the data receiver can inform the data sender when a CE packet has been received; and a Congestion Window Reduced (CWR) flag in the TCP header so that the data sender can inform the data receiver that the congestion window has been reduced. The support required from other transport protocols is likely to be different, particular for unreliable or reliable multicast transport protocols, and will have to be determined as other transport protocols are brought to the IETF for standardization.

6.1. TCP

The following sections describe in detail the proposed use of ECN in TCP. This proposal is described in essentially the same form in [Floyd94]. We assume that the source TCP uses the standard congestion control algorithms of Slow-start, Fast Retransmit and Fast Recovery [RFC 2001].

This proposal specifies two new flags in the Reserved field of the TCP header. The TCP mechanism for negotiating ECN-Capability uses the ECN-Echo flag in the TCP header. (This was called the ECN Notify flag in some earlier documents.) Bit 9 in the Reserved field of the TCP header is designated as the ECN-Echo flag. The location of the 6-bit Reserved field in the TCP header is shown in Figure 3 of RFC 793 [RFC793].

To enable the TCP receiver to determine when to stop setting the ECN-Echo flag, we introduce a second new flag in the TCP header, the Congestion Window Reduced (CWR) flag. The CWR flag is assigned to Bit 8 in the Reserved field of the TCP header.

The use of these flags is described in the sections below.

6.1.1. TCP Initialization

In the TCP connection setup phase, the source and destination TCPs exchange information about their desire and/or capability to use ECN. Subsequent to the completion of this negotiation, the TCP sender sets the ECT bit in the IP header of data packets to indicate to the network that the transport is capable and willing to participate in ECN for this packet. This will indicate to the routers that they may mark this packet with the CE bit, if they would like to use that as a method of congestion notification. If the TCP connection does not wish to use ECN notification for a particular packet, the sending TCP sets the ECT bit equal to 0 (i.e., not set), and the TCP receiver ignores the CE bit in the received packet.

When a node sends a TCP SYN packet, it may set the ECN-Echo and CWR flags in the TCP header. For a SYN packet, the setting of both the ECN-Echo and CWR flags are defined as an indication that the sending TCP is ECN-Capable, rather than as an indication of congestion or of response to congestion. More precisely, a SYN packet with both the ECN-Echo and CWR flags set indicates that the TCP implementation transmitting the SYN packet will participate in ECN as both a sender and receiver. As a receiver, it will respond to incoming data packets that have the CE bit set in the IP header by setting the ECN-Echo flag in outgoing TCP Acknowledgement (ACK) packets. As a sender, it will respond to incoming packets that have the ECN-Echo flag set by reducing the congestion window when appropriate.

When a node sends a SYN-ACK packet, it may set the ECN-Echo flag, but it does not set the CWR flag. For a SYN-ACK packet, the pattern of the ECN-Echo flag set and the CWR flag not set in the TCP header is defined as an indication that the TCP transmitting the SYN-ACK packet is ECN-Capable.

There is the question of why we chose to have the TCP sending the SYN set two ECN-related flags in the Reserved field of the TCP header for the SYN packet, while the responding TCP sending the SYN-ACK sets only one ECN-related flag in the SYN-ACK packet. This asymmetry is necessary for the robust negotiation of ECN-capability with deployed TCP implementations. There exists at least one TCP implementation in which TCP receivers set the Reserved field of the TCP header in ACK packets (and hence the SYN-ACK) simply to reflect the Reserved field of the TCP header in the received data packet. Because the TCP SYN packet sets the ECN-Echo and CWR flags to indicate ECN-capability, while the SYN-ACK packet sets only the ECN-Echo flag, the sending TCP correctly interprets a receiver’s reflection of its own flags in the Reserved field as an indication that the receiver is not ECN-capable.

6.1.2. The TCP Sender

For a TCP connection using ECN, data packets are transmitted with the ECT bit set in the IP header (set to a "1"). If the sender receives an ECN-Echo ACK packet (that is, an ACK packet with the ECN-Echo flag set in the TCP header), then the sender knows that congestion was encountered in the network on the path from the sender to the receiver. The indication of congestion should be treated just as a congestion loss in non-ECN-Capable TCP. That is, the TCP source halves the congestion window "cwnd" and reduces the slow start threshold "ssthresh". The sending TCP does NOT increase the congestion window in response to the receipt of an ECN-Echo ACK packet.

A critical condition is that TCP does not react to congestion indications more than once every window of data (or more loosely, more than once every round-trip time). That is, the TCP sender’s congestion window should be reduced only once in response to a series of dropped and/or CE packets from a single window of data, In addition, the TCP source should not decrease the slow-start threshold, ssthresh, if it has been decreased within the last round trip time. However, if any retransmitted packets are dropped or have the CE bit set, then this is interpreted by the source TCP as a new instance of congestion.

After the source TCP reduces its congestion window in response to a CE packet, incoming acknowledgements that continue to arrive can "clock out" outgoing packets as allowed by the reduced congestion window. If the congestion window consists of only one MSS (maximum segment size), and the sending TCP receives an ECN-Echo ACK packet, then the sending TCP should in principle still reduce its congestion window in half. However, the value of the congestion window is bounded below by a value of one MSS. If the sending TCP were to continue to send, using a congestion window of 1 MSS, this results in the transmission of one packet per round-trip time. We believe it is desirable to still reduce the sending rate of the TCP sender even further, on receipt of an ECN-Echo packet when the congestion window is one. We use the retransmit timer as a means to reduce the rate further in this circumstance. Therefore, the sending TCP should also reset the retransmit timer on receiving the ECN-Echo packet when the congestion window is one. The sending TCP will then be able to send a new packet when the retransmit timer expires.

[Floyd94] discusses TCP’s response to ECN in more detail. [Floyd98] discusses the validation test in the ns simulator, which illustrates a wide range of ECN scenarios. These scenarios include the following: an ECN followed by another ECN, a Fast Retransmit, or a Retransmit Timeout; a Retransmit Timeout or a Fast Retransmit followed by an ECN, and a congestion window of one packet followed by an ECN.

TCP follows existing algorithms for sending data packets in response to incoming ACKs, multiple duplicate acknowledgements, or retransmit timeouts [RFC2001].

6.1.3. The TCP Receiver

When TCP receives a CE data packet at the destination end-system, the TCP data receiver sets the ECN-Echo flag in the TCP header of the subsequent ACK packet. If there is any ACK withholding implemented, as in current "delayed-ACK" TCP implementations where the TCP receiver can send an ACK for two arriving data packets, then the ECN-Echo flag in the ACK packet will be set to the OR of the CE bits of all of the data packets being acknowledged. That is, if any of the received data packets are CE packets, then the returning ACK has the ECN-Echo flag set.

To provide robustness against the possibility of a dropped ACK packet carrying an ECN-Echo flag, the TCP receiver must set the ECN-Echo flag in a series of ACK packets. The TCP receiver uses the CWR flag to determine when to stop setting the ECN-Echo flag.

When an ECN-Capable TCP reduces its congestion window for any reason (because of a retransmit timeout, a Fast Retransmit, or in response to an ECN Notification), the TCP sets the CWR flag in the TCP header of the first data packet sent after the window reduction. If that data packet is dropped in the network, then the sending TCP will have to reduce the congestion window again and retransmit the dropped packet. Thus, the Congestion Window Reduced message is reliably delivered to the data receiver.

After a TCP receiver sends an ACK packet with the ECN-Echo bit set, that TCP receiver continues to set the ECN-Echo flag in ACK packets until it receives a CWR packet (a packet with the CWR flag set). After the receipt of the CWR packet, acknowledgements for subsequent non-CE data packets do not have the ECN-Echo flag set. If another CE packet is received by the data receiver, the receiver would once again send ACK packets with the ECN-Echo flag set. While the receipt of a CWR packet does not guarantee that the data sender received the ECN-Echo message, this does indicate that the data sender reduced its congestion window at some point *after* it sent the data packet for which the CE bit was set.

We have already specified that a TCP sender reduces its congestion window at most once per window of data. This mechanism requires some care to make sure that the sender reduces its congestion window at most once per ECN indication, and that multiple ECN messages over several successive windows of data are properly reported to the ECN sender. This is discussed further in [Floyd98].

6.1.4. Congestion on the ACK-path

For the current generation of TCP congestion control algorithms, pure acknowledgement packets (e.g., packets that do not contain any accompanying data) should be sent with the ECT bit off. Current TCP receivers have no mechanisms for reducing traffic on the ACK-path in response to congestion notification. Mechanisms for responding to congestion on the ACK-path are areas for current and future research. (One simple possibility would be for the sender to reduce its congestion window when it receives a pure ACK packet with the CE bit set). For current TCP implementations, a single dropped ACK generally has only a very small effect on the TCP’s sending rate.

7. Summary of changes required in IP and TCP

Two bits need to be specified in the IP header, the ECN-Capable Transport (ECT) bit and the Congestion Experienced (CE) bit. The ECT bit set to "0" indicates that the transport protocol will ignore the

CE bit. This is the default value for the ECT bit. The ECT bit set to "1" indicates that the transport protocol is willing and able to participate in ECN.

The default value for the CE bit is "0". The router sets the CE bit to "1" to indicate congestion to the end nodes. The CE bit in a packet header should never be reset by a router from "1" to "0".

TCP requires three changes, a negotiation phase during setup to determine if both end nodes are ECN-capable, and two new flags in the TCP header, from the "reserved" flags in the TCP flags field. The ECN-Echo flag is used by the data receiver to inform the data sender of a received CE packet. The Congestion Window Reduced flag is used by the data sender to inform the data receiver that the congestion window has been reduced.

8. Non-relationship to ATM's EFCI indicator or Frame Relay's FECN

Since the ATM and Frame Relay mechanisms for congestion indication have typically been defined without any notion of average queue size as the basis for determining that an intermediate node is congested, we believe that they provide a very noisy signal. The TCP-sender reaction specified in this draft for ECN is NOT the appropriate reaction for such a noisy signal of congestion notification. It is our expectation that ATM's EFCI and Frame Relay's FECN mechanisms would be phased out over time within the ATM network. However, if the routers that interface to the ATM network have a way of maintaining the average queue at the interface, and use it to come to a reliable determination that the ATM subnet is congested, they may use the ECN notification that is defined here.

We emphasize that a *single* packet with the CE bit set in an IP packet causes the transport layer to respond, in terms of congestion control, as it would to a packet drop. As such, the CE bit is not a good match to a transient signal such as one based on the instantaneous queue size. However, experiments in techniques at layer 2 (e.g., in ATM switches or Frame Relay switches) should be encouraged. For example, using a scheme such as RED (where packet marking is based on the average queue length exceeding a threshold), layer 2 devices could provide a reasonably reliable indication of congestion. When all the layer 2 devices in a path set that layer's own Congestion Experienced bit (e.g., the EFCI bit for ATM, the FECN bit in Frame Relay) in this reliable manner, then the interface router to the layer 2 network could copy the state of that layer 2 Congestion Experienced bit into the CE bit in the IP header. We recognize that this is not the current practice, nor is it in current standards. However, encouraging experimentation in this manner may provide the information needed to enable evolution of existing layer 2 mechanisms to provide a more reliable means of congestion indication, when they use a single bit for indicating congestion.

9. Non-compliance by the End Nodes

This section discusses concerns about the vulnerability of ECN to non-compliant end-nodes (i.e., end nodes that set the ECT bit in transmitted packets but do not respond to received CE packets). We argue that the addition of ECN to the IP architecture would not significantly increase the current vulnerability of the architecture to unresponsive flows.

Even for non-ECN environments, there are serious concerns about the damage that can be done by non-compliant or unresponsive flows (that is, flows that do not respond to congestion control indications by reducing their arrival rate at the congested link). For example, an end-node could "turn off congestion control" by not reducing its congestion window in response to packet drops. This is a concern for the current Internet. It has been argued that routers will have to deploy mechanisms to detect and differentially treat packets from non-compliant flows. It has also been argued that techniques such as end-to-end per-flow scheduling and isolation of one flow from another, differentiated services, or end-to-end reservations could remove some of the more damaging effects of unresponsive flows.

It has been argued that dropping packets in itself may be an adequate deterrent for non-compliance, and that the use of ECN removes this deterrent. We would argue in response that (1) ECN-capable routers preserve packet-dropping behavior in times of high congestion; and (2) even in times of high congestion, dropping packets in itself is not an adequate deterrent for non-compliance.

First, ECN-Capable routers will only mark packets (as opposed to dropping them) when the packet marking rate is reasonably low. During periods where the average queue size exceeds an upper threshold, and therefore the potential packet marking rate would be high, our recommendation is that routers drop packets rather then set the CE bit in packet headers.

During the periods of low or moderate packet marking rates when ECN would be deployed, there would be little deterrent effect on unresponsive flows of dropping rather than marking those packets. For example, delay-insensitive flows using reliable delivery might have an incentive to increase rather than to decrease their sending rate in the presence of dropped packets. Similarly, delay-sensitive flows using unreliable delivery might increase their use of FEC in response to an increased packet drop rate, increasing rather than decreasing their sending rate. For the same reasons, we do not believe that packet dropping itself is an effective deterrent for non-compliance even in an environment of high packet drop rates.

Several methods have been proposed to identify and restrict non-compliant or unresponsive flows. The addition of ECN to the network environment would not in any way increase the difficulty of designing and deploying such mechanisms. If anything, the addition of ECN to the architecture would make the job of identifying unresponsive flows slightly easier. For example, in an ECN-Capable environment routers are not limited to information about packets that are dropped or have the CE bit set at that router itself; in such an environment routers could also take note of arriving CE packets that indicate congestion encountered by that packet earlier in the path.

10. Non-compliance in the Network

The breakdown of effective congestion control could be caused not only by a non-compliant end-node, but also by the loss of the congestion indication in the network itself. This could happen through a rogue or broken router that set the ECT bit in a packet from a non-ECN-capable transport, or "erased" the CE bit in arriving packets. As one example, a rogue or broken router that "erased" the CE bit in arriving CE packets would prevent that indication of congestion from reaching downstream receivers. This could result in the failure of congestion control for that flow and a resulting increase in congestion in the network, ultimately resulting in subsequent packets dropped for this flow as the average queue size increased at the congested gateway.

The actions of a rogue or broken router could also result in an unnecessary indication of congestion to the end-nodes. These actions can include a router dropping a packet or setting the CE bit in the absence of congestion. From a congestion control point of view, setting the CE bit in the absence of congestion by a non-compliant router would be no different than a router dropping a packet unecessarily. By "erasing" the ECT bit of a packet that is later dropped in the network, a router's actions could result in an unnecessary packet drop for that packet later in the network.

Concerns regarding the loss of congestion indications from encapsulated, dropped, or corrupted packets are discussed below.

10.1. Encapsulated packets

Some care is required to handle the CE and ECT bits appropriately when packets are encapsulated and de-encapsulated for tunnels.

When a packet is encapsulated, the following rules apply regarding the ECT bit. First, if the ECT bit in the encapsulated ('inside') header is a 0, then the ECT bit in the encapsulating ('outside') header MUST be a 0. If the ECT bit in the inside header is a 1, then the ECT bit in the outside header SHOULD be a 1.

When a packet is de-encapsulated, the following rules apply regarding the CE bit. If the ECT bit is a 1 in both the inside and the outside header, then the CE bit in the outside header MUST be ORed with the CE bit in the inside header. (That is, in this case a CE bit of 1 in the outside header must be copied to the inside header.) If the ECT bit in either header is a 0, then the CE bit in the outside header is ignored. This requirement for the treatment of de-encapsulated packets does not currently apply to IPsec tunnels.

A specific example of the use of ECN with encapsulation occurs when a flow wishes to use ECN-capability to avoid the danger of an unnecessary packet drop for the encapsulated packet as a result of congestion at an intermediate node in the tunnel. This functionality can be supported by copying the ECN field in the inner IP header to the outer IP header upon encapsulation, and using the ECN field in the outer IP header to set the ECN field in the inner IP header upon decapsulation. This effectively allows routers along the tunnel to cause the CE bit to be set in the ECN field of the unencapsulated IP header of an ECN-capable packet when such routers experience congestion.

10.2. IPsec Tunnel Considerations

The IPsec protocol, as defined in [ESP, AH], does not include the IP header's ECN field in any of its cryptographic calculations (in the case of tunnel mode, the outer IP header's ECN field is not included). Hence modification of the ECN field by a network node has no effect on IPsec's end-to-end security, because it cannot cause any IPsec integrity check to fail. As a consequence, IPsec does not provide any defense against an adversary's modification of the ECN field (i.e., a man-in-the-middle attack), as the adversary's modification will also have no effect on IPsec's end-to-end security. In some environments, the ability to modify the ECN field without affecting IPsec integrity checks may constitute a covert channel; if it is necessary to eliminate such a channel or reduce its bandwidth, then the outer IP header's ECN field can be zeroed at the tunnel ingress and egress nodes.

The IPsec protocol currently requires that the inner header's ECN field not be changed by IPsec decapsulation processing at a tunnel egress node. This ensures that an adversary's modifications to the ECN field cannot be used to launch theft- or denial-of-service attacks across an IPsec tunnel endpoint, as any such modifications will be discarded at the tunnel endpoint. This document makes no change to that IPsec requirement. As a consequence of the current specification of the IPsec protocol, we suggest that experiments with ECN not be carried out for flows that will undergo IPsec tunneling at the present time.

If the IPsec specifications are modified in the future to permit a tunnel egress node to modify the ECN field in an inner IP header based on the ECN field value in the outer header (e.g., copying part or all of the outer ECN field to the inner ECN field), or to permit the ECN field of the outer IP header to be zeroed during encapsulation, then experiments with ECN may be used in combination with IPsec tunneling.

This discussion of ECN and IPsec tunnel considerations draws heavily on related discussions and documents from the Differentiated Services Working Group.

10.3. Dropped or Corrupted Packets

An additional issue concerns a packet that has the CE bit set at one router and is dropped by a subsequent router. For the proposed use for ECN in this paper (that is, for a transport protocol such as TCP for which a dropped data packet is an indication of congestion), end nodes detect dropped data packets, and the congestion response of the end nodes to a dropped data packet is at least as strong as the congestion response to a received CE packet.

However, transport protocols such as TCP do not necessarily detect all packet drops, such as the drop of a "pure" ACK packet; for example, TCP does not reduce the arrival rate of subsequent ACK packets in response to an earlier dropped ACK packet. Any proposal for extending ECN-Capability to such packets would have to address concerns raised by CE packets that were later dropped in the network.

Similarly, if a CE packet is dropped later in the network due to corruption (bit errors), the end nodes should still invoke congestion control, just as TCP would today in response to a dropped data packet. This issue of corrupted CE packets would have to be considered in any proposal for the network to distinguish between packets dropped due to corruption, and packets dropped due to congestion or buffer overflow.

11. A summary of related work.

[Floyd94] considers the advantages and drawbacks of adding ECN to the TCP/IP architecture. As shown in the simulation-based comparisons, one advantage of ECN is to avoid unnecessary packet drops for short or delay-sensitive TCP connections. A second advantage of ECN is in avoiding some unnecessary retransmit timeouts in TCP. This paper discusses in detail the integration of ECN into TCP's congestion control mechanisms. The possible disadvantages of ECN discussed in the paper are that a non-compliant TCP connection could falsely advertise itself as ECN-capable, and that a TCP ACK packet carrying an ECN-Echo message could itself be dropped in the network. The first of these two issues is discussed in Section 8 of this document, and the second is addressed by the proposal in Section 5.1.3 for a CWR flag in the TCP header.

[CKLTZ97] reports on an experimental implementation of ECN in IPv6. The experiments include an implementation of ECN in an existing implementation of RED for FreeBSD. A number of experiments were run to demonstrate the control of the average queue size in the router, the performance of ECN for a single TCP connection as a congested router, and fairness with multiple competing TCP connections. One conclusion of the experiments is that dropping packets from a bulk-data transfer can degrade performance much more severely than marking packets.

Because the experimental implementation in [CKLTZ97] predates some of the developments in this document, the implementation does not conform to this document in all respects. For example, in the experimental implementation the CWR flag is not used, but instead the TCP receiver sends the ECN-Echo bit on a single ACK packet.

[K98] and [CKLTZ98] build on [CKLTZ97] to further analyze the benefits of ECN for TCP. The conclusions are that ECN TCP gets moderately better throughput than non-ECN TCP; that ECN TCP flows are fair towards non-ECN TCP flows; and that ECN TCP is robust with two-way traffic, congestion in both directions, and with multiple congested gateways. Experiments with many short web transfers show that, while most of the short connections have similar transfer times with or without ECN, a small percentage of the short connections have very long transfer times for the non-ECN experiments as compared to the ECN experiments. This increased transfer time is particularly dramatic for those short connections that have their first packet dropped in the non-ECN experiments, and that therefore have to wait six seconds for the retransmit timer to expire.

The ECN Web Page [ECN] has pointers to other implementations of ECN in progress.

12. Conclusions

Given the current effort to implement RED, we believe this is the right time for router vendors to examine how to implement congestion avoidance mechanisms that do not depend on packet drops alone. With the increased deployment of applications and transports sensitive to the delay and loss of a single packet (e.g., realtime traffic, short web transfers), depending on packet loss as a normal congestion notification mechanism appears to be insufficient (or at the very least, non-optimal).

13. Acknowledgements

Many people have made contributions to this RFC. In particular, we would like to thank Kenjiro Cho for the proposal for the TCP mechanism for negotiating ECN-Capability, Kevin Fall for the proposal of the CWR bit, Steve Blake for material on IPv4 Header Checksum Recalculation, Jamal Hadi Salim for discussions of ECN issues, and Steve Bellovin, Jim Bound, Brian Carpenter, Paul Ferguson, Stephen Kent, Greg Minshall, and Vern Paxson for discussions of security issues. We also thank the Internet End-to-End Research Group for ongoing discussions of these issues.

14. References

[AH] Kent, S. and R. Atkinson, "IP Authentication Header", RFC 2402, November 1998.

[B97] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[CKLT98] Chen, C., Krishnan, H., Leung, S., Tang, N., and Zhang, L., "Implementing ECN for TCP/IPv6", presentation to the ECN BOF at the L.A. IETF, March 1998, URL "http://www.cs.ucla.edu/~hari/ecn-ietf.ps".

[DIFFSERV] Nichols, K., Blake, S., Baker, F. and D. Black, "Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers", RFC 2474, December 1998.

[ECN] "The ECN Web Page", URL "http://www-nrg.ee.lbl.gov/floyd/ecn.html".

[ESP] Kent, S. and R. Atkinson, "IP Encapsulating Security Payload", RFC 2406, November 1998.

[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. URL "ftp://ftp.ee.lbl.gov/papers/early.pdf".

[Floyd94] Floyd, S., "TCP and Explicit Congestion Notification", ACM Computer Communication Review, V. 24 N. 5, October 1994, p. 10-23. URL "ftp://ftp.ee.lbl.gov/papers/tcp_ecn.4.ps.Z".

[Floyd97] Floyd, S., and Fall, K., "Router Mechanisms to Support End-to-End Congestion Control", Technical report, February 1997. URL "http://www-nrg.ee.lbl.gov/floyd/end2end-paper.html".

[Floyd98] Floyd, S., "The ECN Validation Test in the NS Simulator", URL "http://www-mash.cs.berkeley.edu/ns/", test tcl/test/test-all-ecn.

[K98] Krishnan, H., "Analyzing Explicit Congestion Notification (ECN) benefits for TCP", Master's thesis, UCLA, 1998, URL "http://www.cs.ucla.edu/~hari/software/ecn/ecn_report.ps.gz".

[FRED] Lin, D., and Morris, R., "Dynamics of Random Early Detection", SIGCOMM '97, September 1997. URL "http://www.inria.fr/rodeo/sigcomm97/program.html#ab078".

[Jacobson88] V. Jacobson, "Congestion Avoidance and Control", Proc. ACM SIGCOMM '88, pp. 314-329. URL "ftp://ftp.ee.lbl.gov/papers/congavoid.ps.Z".

[Jacobson90] V. Jacobson, "Modified TCP Congestion Avoidance Algorithm", Message to end2end-interest mailing list, April 1990. URL "ftp://ftp.ee.lbl.gov/email/vanj.90apr30.txt".

[MJV96] S. McCanne, V. Jacobson, and M. Vetterli, "Receiver-driven Layered Multicast", SIGCOMM '96, August 1996, pp. 117-130.

[RFC791] Postel, J., "Internet Protocol", STD 5, RFC 791, September 1981.

[RFC793] Postel, J., "Transmission Control Protocol", STD 7, RFC 793, September 1981.

[RFC1141] Mallory, T. and A. Kullberg, "Incremental Updating of the Internet Checksum", RFC 1141, January 1990.

[RFC1349] Almquist, P., "Type of Service in the Internet Protocol Suite", RFC 1349, July 1992.

[RFC1455] Eastlake, D., "Physical Link Security Type of Service", RFC 1455, May 1993.

[RFC2001] Stevens, W., "TCP Slow Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery Algorithms", RFC 2001, January 1997.

[RFC2309] Braden, B., Clark, D., Crowcroft, J., Davie, B., Deering, S., Estrin, D., Floyd, S., Jacobson, V., Minshall, G., Partridge, C., Peterson, L., Ramakrishnan, K., Shenker, S., Wroclawski, J. and L. Zhang, "Recommendations on Queue Management and Congestion Avoidance in the Internet", RFC 2309, April 1998.

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

15. Security Considerations

Security considerations have been discussed in Section 9.

16. IPv4 Header Checksum Recalculation

IPv4 header checksum recalculation is an issue with some high-end router architectures using an output-buffered switch, since most if not all of the header manipulation is performed on the input side of the switch, while the ECN decision would need to be made local to the output buffer. This is not an issue for IPv6, since there is no IPv6 header checksum. The IPv4 TOS octet is the last byte of a 16-bit half-word.

RFC 1141 [RFC1141] discusses the incremental updating of the IPv4 checksum after the TTL field is decremented. The incremental updating of the IPv4 checksum after the CE bit was set would work as follows: Let HC be the original header checksum, and let HC' be the new header checksum after the CE bit has been set. Then for header checksums calculated with one's complement subtraction, HC' would be recalculated as follows:

```
HC' = { HC - 1     HC > 1
      { 0x0000     HC = 1
```

For header checksums calculated on two's complement machines, HC' would be recalculated as follows after the CE bit was set:

```
HC' = { HC - 1     HC > 0
      { 0xFFFE     HC = 0
```

17. The motivation for the ECT bit.

The need for the ECT bit is motivated by the fact that ECN will be deployed incrementally in an Internet where some transport protocols and routers understand ECN and some do not. With the ECT bit, the router can drop packets from flows that are not ECN-capable, but can *instead* set the CE bit in flows that *are* ECN-capable. Because the ECT bit allows an end node to have the CE bit set in a packet *instead* of having the packet dropped, an end node might have some incentive to deploy ECN.

If there was no ECT indication, then the router would have to set the CE bit for packets from both ECN-capable and non-ECN-capable flows. In this case, there would be no incentive for end-nodes to deploy ECN, and no viable path of incremental deployment from a non-ECN world to an ECN-capable world. Consider the first stages of such an incremental deployment, where a subset of the flows are ECN-capable. At the onset of congestion, when the packet dropping/marking rate would be low, routers would only set CE bits, rather than dropping packets. However, only those flows that are ECN-capable would understand and respond to CE packets. The result is that the ECN-capable flows would back off, and the non-ECN-capable flows would be unaware of the ECN signals and would continue to open their congestion windows.

In this case, there are two possible outcomes: (1) the ECN-capable flows back off, the non-ECN-capable flows get all of the bandwidth, and congestion remains mild, or (2) the ECN-capable flows back off, the non-ECN-capable flows don't, and congestion increases until the router transitions from setting the CE bit to dropping packets. While this second outcome evens out the fairness, the ECN-capable flows would still receive little benefit from being ECN-capable, because the increased congestion would drive the router to packet-dropping behavior.

A flow that advertised itself as ECN-Capable but does not respond to CE bits is functionally equivalent to a flow that turns off congestion control, as discussed in Sections 8 and 9.

Thus, in a world when a subset of the flows are ECN-capable, but where ECN-capable flows have no mechanism for indicating that fact to the routers, there would be less effective and less fair congestion control in the Internet, resulting in a strong incentive for end nodes not to deploy ECN.

18. Why use two bits in the IP header?

Given the need for an ECT indication in the IP header, there still remains the question of whether the ECT (ECN-Capable Transport) and CE (Congestion Experienced) indications should be overloaded on a single bit. This overloaded-one-bit alternative, explored in [Floyd94], would involve a single bit with two values. One value, "ECT and not CE", would represent an ECN-Capable Transport, and the other value, "CE or not ECT", would represent either Congestion Experienced or a non-ECN-Capable transport.

One difference between the one-bit and two-bit implementations concerns packets that traverse multiple congested routers. Consider a CE packet that arrives at a second congested router, and is selected by the active queue management at that router for either marking or dropping. In the one-bit implementation, the second congested router has no choice but to drop the CE packet, because it cannot distinguish between a CE packet and a non-ECT packet. In the two-bit implementation, the second congested router has the choice of either dropping the CE packet, or of leaving it alone with the CE bit set.

Another difference between the one-bit and two-bit implementations comes from the fact that with the one-bit implementation, receivers in a single flow cannot distinguish between CE and non-ECT packets. Thus, in the one-bit implementation an ECN-capable data sender would have to unambiguously indicate to the receiver or receivers whether each packet had been sent as ECN-Capable or as non-ECN-Capable. One possibility would be for the sender to indicate in the transport header whether the packet was sent as ECN-Capable. A second possibility that would involve a functional limitation for the one-bit implementation would be for the sender to unambiguously indicate that it was going to send *all* of its packets as ECN-Capable or as non-ECN-Capable. For a multicast transport protocol, this unambiguous indication would have to be apparent to receivers joining an on-going multicast session.

Another advantage of the two-bit approach is that it is somewhat more robust. The most critical issue, discussed in Section 8, is that the default indication should be that of a non-ECN-Capable transport. In a two-bit implementation, this requirement for the default value simply means that the ECT bit should be 'OFF' by default. In the one-bit implementation, this means that the single overloaded bit should by default be in the "CE or not ECT" position. This is less clear and straightforward, and possibly more open to incorrect implementations either in the end nodes or in the routers.

In summary, while the one-bit implementation could be a possible implementation, it has the following significant limitations relative to the two-bit implementation. First, the one-bit implementation has more limited functionality for the treatment of CE packets at a second congested router. Second, the one-bit implementation requires either that extra information be carried in the transport header of packets from ECN-Capable flows (to convey the functionality of the second bit elsewhere, namely in the transport header), or that senders in ECN-Capable flows accept the limitation that receivers must be able to determine a priori which packets are ECN-Capable and which are not ECN-Capable. Third, the one-bit implementation is possibly more open to errors from faulty implementations that choose the wrong default value for the ECN bit. We believe that the use of the extra bit in the IP header for the ECT-bit is extremely valuable to overcome these limitations.

19. Historical definitions for the IPv4 TOS octet

RFC 791 [RFC791] defined the ToS (Type of Service) octet in the IP header. In RFC 791, bits 6 and 7 of the ToS octet are listed as "Reserved for Future Use", and are shown set to zero. The first two fields of the ToS octet were defined as the Precedence and Type of Service (TOS) fields.

```
   0     1     2     3     4     5     6     7
+-----+-----+-----+-----+-----+-----+-----+-----+
|   PRECEDENCE    |       TOS       |  0  |  0  |    RFC 791
+-----+-----+-----+-----+-----+-----+-----+-----+
```

RFC 1122 included bits 6 and 7 in the TOS field, though it did not discuss any specific use for those two bits:

```
   0     1     2     3     4     5     6     7
+-----+-----+-----+-----+-----+-----+-----+-----+
|   PRECEDENCE    |       TOS                   |    RFC 1122
+-----+-----+-----+-----+-----+-----+-----+-----+
```

The IPv4 TOS octet was redefined in RFC 1349 [RFC1349] as follows:

```
   0     1     2     3     4     5     6     7
+-----+-----+-----+-----+-----+-----+-----+-----+
|   PRECEDENCE    |       TOS             | MBZ |    RFC 1349
+-----+-----+-----+-----+-----+-----+-----+-----+
```

Bit 6 in the TOS field was defined in RFC 1349 for "Minimize Monetary Cost". In addition to the Precedence and Type of Service (TOS) fields, the last field, MBZ (for "must be zero") was defined as currently unused. RFC 1349 stated that "The originator of a datagram sets [the MBZ] field to zero (unless participating in an Internet protocol experiment which makes use of that bit)."

RFC 1455 [RFC 1455] defined an experimental standard that used all four bits in the TOS field to request a guaranteed level of link security.

RFC 1349 is obsoleted by "Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers" [DIFFSERV], in which bits 6 and 7 of the DS field are listed as Currently Unused (CU). The first six bits of the DS field are defined as the Differentiated Services CodePoint (DSCP):

```
     0     1     2     3     4     5     6     7
  +-----+-----+-----+-----+-----+-----+-----+-----+
  |               DSCP                |    CU     |
  +-----+-----+-----+-----+-----+-----+-----+-----+
```

Because of this unstable history, the definition of the ECN field in this document cannot be guaranteed to be backwards compatible with all past uses of these two bits. The damage that could be done by a non-ECN-capable router would be to "erase" the CE bit for an ECN-capable packet that arrived at the router with the CE bit set, or set the CE bit even in the absence of congestion. This has been discussed in Section 10 on "Non-compliance in the Network".

The damage that could be done in an ECN-capable environment by a non-ECN-capable end-node transmitting packets with the ECT bit set has been discussed in Section 9 on "Non-compliance by the End Nodes".

AUTHORS' ADDRESSES

K. K. Ramakrishnan
AT&T Labs. Research

Phone: +1 (973) 360-8766
EMail: kkrama@research.att.com
URL: http://www.research.att.com/info/kkrama

Sally Floyd
Lawrence Berkeley National Laboratory

Phone: +1 (510) 486-7518
EMail: floyd@ee.lbl.gov
URL: http://www-nrg.ee.lbl.gov/floyd/