Network Working Group
Request for Comments: 3168
Updates: 2474, 2401, 793
Obsoletes: 2481
Category: Standards Track
September 2001

K. Ramakrishnan
TeraOptic Networks
S. Floyd
ACIRI
D. Black
EMC

The Addition of Explicit Congestion Notification (ECN) to IP

Status of this Memo

This document specifies an Internet standards track protocol for the Internet community, and requests discussion and suggestions for improvements. Please refer to the current edition of the "Internet Official Protocol Standards" (STD 1) for the standardization state and status of this protocol. Distribution of this memo is unlimited.

Copyright Notice

Abstract

This memo specifies the incorporation of ECN (Explicit Congestion Notification) to TCP and IP, including ECN's use of two bits in the IP header.

Table of Contents

1. Introduction

We begin by describing TCP's use of packet drops as an indication of congestion. Next we explain 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 can instead set the Congestion Experienced (CE) codepoint in the IP header of packets from ECN-capable transports. We describe when the CE codepoint is to be set in routers, and describe modifications 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. We also describe in this document the issues involving the use of ECN within IP tunnels, and within IPsec tunnels in particular.

One of the guiding principles for this document is that, to the extent possible, the mechanisms specified here be incrementally deployable. One challenge to the principle of incremental deployment has been the prior existence of some IP tunnels that were not compatible with the use of ECN. As ECN becomes deployed, non-compatible IP tunnels will have to be upgraded to conform to this document.

This document obsoletes RFC 2481, "A Proposal to add Explicit Congestion Notification (ECN) to IP", which defined ECN as an Experimental Protocol for the Internet Community. This document also updates RFC 2474, "Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers", in defining the ECN field in the IP header, RFC 2401, "Security Architecture for the Internet Protocol" to change the handling of IPv4 TOS Byte and IPv6 Traffic Class Octet in tunnel mode header construction to be compatible with the use of ECN, and RFC 793, "Transmission Control Protocol", in defining two new flags in the TCP header.

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, with 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 (especially when 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 (with the reliable data delivery semantics provided by 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 (e.g., tail-drop on queue overflow), this means that some of the packets of latency-sensitive flows may be dropped. In addition, such drop policies lead to synchronization of loss across multiple flows.

Active queue management mechanisms detect congestion before the queue overflows, and provide an indication of this congestion to the end nodes. Thus, active queue management can reduce unnecessary queuing delay for all traffic sharing that queue. 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 mechanisms for congestion control (e.g., TCP) do not have to rely on buffer overflow as the only indication of congestion.

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 queuing or dropping packets from the policies for indicating congestion. Thus, active queue management allows routers to use the Congestion Experienced (CE) codepoint in a packet header as an indication of congestion, instead of relying solely on packet drops. This has the potential of reducing the impact of loss on latency-sensitive flows.

There exist some middleboxes (firewalls, load balancers, or intrusion detection systems) in the Internet that either drop a TCP SYN packet configured to negotiate ECN, or respond with a RST. This document specifies procedures that TCP implementations may use to provide robust connectivity even in the presence of such equipment.

2. 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 [RFC2119].

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.

* 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.

* 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.

* 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.

* 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.

* 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.

* 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.

* 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. Active Queue Management (AQM)

Random Early Detection (RED) is one mechanism for Active Queue Management (AQM) that has been proposed to detect incipient congestion [FJ93], and is currently being deployed in the Internet [RFC2309]. AQM is meant to be a general mechanism using one of several alternatives for congestion indication, but in the absence of ECN, AQM is restricted to using packet drops as a mechanism for congestion indication. AQM drops packets based on the average queue length exceeding a threshold, rather than only when the queue overflows. However, because AQM may drop packets before the queue actually overflows, AQM is not always forced by memory limitations to discard the packet.

AQM can set a Congestion Experienced (CE) codepoint in the packet header instead of dropping the packet, when such a field is provided in the IP header and understood by the transport protocol. The use of the CE codepoint with ECN allows 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 codepoint set.

5. Explicit Congestion Notification in IP

This document specifies 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 uses an ECN field in the IP header with two bits, making four ECN codepoints, '00' to '11'. The ECN-Capable Transport (ECT) codepoints '10' and '01' are set by the data sender to indicate that the end-points of the transport protocol are ECN-capable; we call them ECT(0) and ECT(1) respectively. The phrase "the ECT codepoint" in this documents refers to either of the two ECT codepoints. Routers treat the ECT(0) and ECT(1) codepoints as equivalent. Senders are free to use either the ECT(0) or the ECT(1) codepoint to indicate ECT, on a packet-by-packet basis.

The use of both the two codepoints for ECT, ECT(0) and ECT(1), is motivated primarily by the desire to allow mechanisms for the data sender to verify that network elements are not erasing the CE codepoint, and that data receivers are properly reporting to the sender the receipt of packets with the CE codepoint set, as required by the transport protocol. Guidelines for the senders and receivers to differentiate between the ECT(0) and ECT(1) codepoints will be addressed in separate documents, for each transport protocol. In particular, this document does not address mechanisms for TCP end-nodes to differentiate between the ECT(0) and ECT(1) codepoints. Protocols and senders that only require a single ECT codepoint SHOULD use ECT(0).

The not-ECT codepoint '00' indicates a packet that is not using ECN. The CE codepoint '11' is set by a router to indicate congestion to the end nodes. Routers that have a packet arriving at a full queue drop the packet, just as they do in the absence of ECN.

```
+-----+-----+
| ECN FIELD |
+-----+-----+
  ECT   CE         [Obsolete] RFC 2481 names for the ECN bits.
   0     0         Not-ECT
   0     1         ECT(1)
   1     0         ECT(0)
   1     1         CE
```

Figure 1: The ECN Field in IP.

The use of two ECT codepoints essentially gives a one-bit ECN nonce in packet headers, and routers necessarily "erase" the nonce when they set the CE codepoint [SCWA99]. For example, routers that erased the CE codepoint would face additional difficulty in reconstructing the original nonce, and thus repeated erasure of the CE codepoint would be more likely to be detected by the end-nodes. The ECN nonce also can address the problem of misbehaving transport receivers lying to the transport sender about whether or not the CE codepoint was set in a packet. The motivations for the use of two ECT codepoints is discussed in more detail in Section 20, along with some discussion of alternate possibilities for the fourth ECT codepoint (that is, the codepoint '01'). Backwards compatibility with earlier ECN implementations that do not understand the ECT(1) codepoint is discussed in Section 11.

In RFC 2481 [RFC2481], the ECN field was divided into the ECN-Capable Transport (ECT) bit and the CE bit. The ECN field with only the ECN-Capable Transport (ECT) bit set in RFC 2481 corresponds to the ECT(0) codepoint in this document, and the ECN field with both the

ECT and CE bit in RFC 2481 corresponds to the CE codepoint in this document. The '01' codepoint was left undefined in RFC 2481, and this is the reason for recommending the use of ECT(0) when only a single ECT codepoint is needed.

```
     0     1     2     3     4     5     6     7
  +-----+-----+-----+-----+-----+-----+-----+-----+
  |          DS FIELD, DSCP           | ECN FIELD |
  +-----+-----+-----+-----+-----+-----+-----+-----+

    DSCP: differentiated services codepoint
    ECN:  Explicit Congestion Notification
```

Figure 2: The Differentiated Services and ECN Fields in IP.

Bits 6 and 7 in the IPv4 TOS octet are designated as the ECN field. The IPv4 TOS octet corresponds to the Traffic Class octet in IPv6, and the ECN field is defined identically in both cases. The definitions for the IPv4 TOS octet [RFC791] and the IPv6 Traffic Class octet have been superseded by the six-bit DS (Differentiated Services) Field [RFC2474, RFC2780]. Bits 6 and 7 are listed in [RFC2474] as Currently Unused, and are specified in RFC 2780 as approved for experimental use for ECN. Section 22 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 those past uses of these two bits that pre-date ECN. The potential dangers of this lack of backwards compatibility are discussed in Section 22.

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.

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 AQM policies for congestion detection) while other routers set the CE codepoint, for equivalent levels of congestion. Similarly, a router might drop a non-ECN-Capable packet but set the CE codepoint in an ECN-Capable packet, for equivalent levels of congestion. If there were different congestion control responses to a CE codepoint than to a packet drop, this could result in unfair treatment for different flows.

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

For a router, the CE codepoint 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 codepoint is set in that packet's IP header. If so, then instead of dropping the packet, the router MAY instead set the CE codepoint in the IP header.

An environment where all end nodes were ECN-Capable could allow new criteria to be developed for setting the CE codepoint, 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 (i.e., a packet that has the CE codepoint set) is received by a router, the CE codepoint is left unchanged, and the packet is 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 ECN-Capable environment that is adequately-provisioned, packet losses should occur primarily during transients or in the presence of non-cooperating sources.

The above discussion of when CE may be set instead of dropping a packet applies by default to all Differentiated Services Per-Hop Behaviors (PHBs) [RFC 2475]. Specifications for PHBs MAY provide more specifics on how a compliant implementation is to choose between setting CE and dropping a packet, but this is NOT REQUIRED. A router MUST NOT set CE instead of dropping a packet when the drop that would occur is caused by reasons other than congestion or the desire to indicate incipient congestion to end nodes (e.g., a diffserv edge node may be configured to unconditionally drop certain classes of traffic to prevent them from entering its diffserv domain).

We expect that routers will set the CE codepoint 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 the need to have a reasonable active queue management mechanism at the router, the reverse does not hold; active queue management mechanisms have been developed and deployed independent of ECN, using packet drops as indications of congestion in the absence of ECN in the IP architecture.

5.1. ECN as an Indication of Persistent Congestion

We emphasize that a *single* packet with the CE codepoint set in an IP packet causes the transport layer to respond, in terms of congestion control, as it would to a packet drop. The instantaneous queue size is likely to see considerable variations even when the router does not experience persistent congestion. As such, it is important that transient congestion at a router, reflected by the instantaneous queue size reaching a threshold much smaller than the capacity of the queue, not trigger a reaction at the transport layer. Therefore, the CE codepoint should not be set by a router based on the instantaneous queue size.

For example, since the ATM and Frame Relay mechanisms for congestion indication have typically been defined without an associated 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 document for ECN is NOT the appropriate reaction for such a noisy signal of congestion notification. 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 continue to encourage experiments in techniques at layer 2 (e.g., in ATM switches or Frame Relay switches) to take advantage of ECN. 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 codepoint (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 codepoint into the CE codepoint 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.

5.2. Dropped or Corrupted Packets

For the proposed use for ECN in this document (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. To ensure the reliable delivery of the congestion indication of the CE codepoint, an ECT codepoint MUST NOT be set in a packet unless the loss of that packet in the network would be detected by the end nodes and interpreted as an indication of congestion.

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 issues such as the case of an ACK packet that was marked with the CE codepoint but was later dropped in the network. We believe that this aspect is still the subject of research, so this document specifies that at this time, "pure" ACK packets MUST NOT indicate ECN-Capability.

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. In particular, the ubiquitous deployment of ECN would not, in and of itself, be a sufficient development to allow end-nodes to interpret packet drops as indications of corruption rather than congestion.

5.3. Fragmentation

ECN-capable packets MAY have the DF (Don't Fragment) bit set. Reassembly of a fragmented packet MUST NOT lose indications of congestion. In other words, if any fragment of an IP packet to be reassembled has the CE codepoint set, then one of two actions MUST be taken:

* Set the CE codepoint on the reassembled packet. However, this MUST NOT occur if any of the other fragments contributing to this reassembly carries the Not-ECT codepoint.

* The packet is dropped, instead of being reassembled, for any other reason.

If both actions are applicable, either MAY be chosen. Reassembly of a fragmented packet MUST NOT change the ECN codepoint when all of the fragments carry the same codepoint.

We would note that because RFC 2481 did not specify reassembly behavior, older ECN implementations conformant with that Experimental RFC do not necessarily perform reassembly correctly, in terms of preserving the CE codepoint in a fragment. The sender could avoid the consequences of this behavior by setting the DF bit in ECN-Capable packets.

Situations may arise in which the above reassembly specification is insufficiently precise. For example, if there is a malicious or broken entity in the path at or after the fragmentation point, packet fragments could carry a mixture of ECT(0), ECT(1), and/or Not-ECT codepoints. The reassembly specification above does not place requirements on reassembly of fragments in this case. In situations where more precise reassembly behavior would be required, protocol specifications SHOULD instead specify that DF MUST be set in all ECN-capable packets sent by the protocol.

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 codepoint 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 on that congested link. CE packets indicate persistent rather than transient congestion (see Section 5.1), and hence reactions to the receipt of CE packets should be those appropriate for persistent congestion.

This document only addresses the addition of ECN Capability to TCP, leaving issues of ECN in other transport protocols to further research. For TCP, ECN requires three new pieces of functionality:

negotiation between the endpoints during connection setup to determine if they are both ECN-capable; an ECN-Echo (ECE) 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, particularly for unreliable or reliable multicast transport protocols, and will have to be determined as other transport protocols are brought to the IETF for standardization.

In a mild abuse of terminology, in this document we refer to 'TCP packets' instead of 'TCP segments'.

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 [RFC2581].

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 (ECE) flag in the TCP header. 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 4 of RFC 793 [RFC793] (and is reproduced below for completeness). This specification of the ECN Field leaves the Reserved field as a 4-bit field using bits 4-7.

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 CWR flag. The CWR flag is assigned to Bit 8 in the Reserved field of the TCP header.

```
  0   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|               |                       | U | A | P | R | S | F |
| Header Length |        Reserved       | R | C | S | S | Y | I |
|               |                       | G | K | H | T | N | N |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
```

Figure 3: The old definition of bytes 13 and 14 of the TCP header.

```
  0   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|               |               | C | E | U | A | P | R | S | F |
| Header Length |    Reserved   | W | C | R | C | S | S | Y | I |
|               |               | R | E | G | K | H | T | N | N |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
```

Figure 4: The new definition of bytes 13 and 14 of the TCP Header.

Thus, ECN uses the ECT and CE flags in the IP header (as shown in Figure 1) for signaling between routers and connection endpoints, and uses the ECN-Echo and CWR flags in the TCP header (as shown in Figure 4) for TCP-endpoint to TCP-endpoint signaling. For a TCP connection, a typical sequence of events in an ECN-based reaction to congestion is as follows:

* An ECT codepoint is set in packets transmitted by the sender to indicate that ECN is supported by the transport entities for these packets.

* An ECN-capable router detects impending congestion and detects that an ECT codepoint is set in the packet it is about to drop. Instead of dropping the packet, the router chooses to set the CE codepoint in the IP header and forwards the packet.

* The receiver receives the packet with the CE codepoint set, and sets the ECN-Echo flag in its next TCP ACK sent to the sender.

* The sender receives the TCP ACK with ECN-Echo set, and reacts to the congestion as if a packet had been dropped.

* The sender sets the CWR flag in the TCP header of the next packet sent to the receiver to acknowledge its receipt of and reaction to the ECN-Echo flag.

The negotiation for using ECN by the TCP transport entities and the use of the ECN-Echo and CWR flags is described in more detail in the sections below.

6.1.1 TCP Initialization

In the TCP connection setup phase, the source and destination TCPs exchange information about their willingness to use ECN. Subsequent to the completion of this negotiation, the TCP sender sets an ECT codepoint 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 indicates to the routers that they may mark this packet with the CE codepoint, 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 ECN codepoint to not-ECT, and the TCP receiver ignores the CE codepoint in the received packet.

For this discussion, we designate the initiating host as Host A and the responding host as Host B. We call a SYN packet with the ECE and CWR flags set an "ECN-setup SYN packet", and we call a SYN packet with at least one of the ECE and CWR flags not set a "non-ECN-setup SYN packet". Similarly, we call a SYN-ACK packet with only the ECE flag set but the CWR flag not set an "ECN-setup SYN-ACK packet", and we call a SYN-ACK packet with any other configuration of the ECE and CWR flags a "non-ECN-setup SYN-ACK packet".

Before a TCP connection can use ECN, Host A sends an ECN-setup SYN packet, and Host B sends an ECN-setup SYN-ACK packet. For a SYN packet, the setting of both ECE and CWR in the ECN-setup SYN packet is 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, an ECN-setup SYN packet indicates that the TCP implementation transmitting the SYN packet will participate in ECN as both a sender and receiver. Specifically, as a receiver, it will respond to incoming data packets that have the CE codepoint set in the IP header by setting ECE in outgoing TCP Acknowledgement (ACK) packets. As a sender, it will respond to incoming packets that have ECE set by reducing the congestion window and setting CWR when appropriate. An ECN-setup SYN packet does not commit the TCP sender to setting the ECT codepoint in any or all of the packets it may transmit. However, the commitment to respond appropriately to incoming packets with the CE codepoint set remains even if the TCP sender in a later transmission, within this TCP connection, sends a SYN packet without ECE and CWR set.

When Host B sends an ECN-setup SYN-ACK packet, it sets the ECE flag but not the CWR flag. An ECN-setup SYN-ACK packet is defined as an indication that the TCP transmitting the SYN-ACK packet is ECN-Capable. As with the SYN packet, an ECN-setup SYN-ACK packet does not commit the TCP host to setting the ECT codepoint in transmitted packets.

The following rules apply to the sending of ECN-setup packets within a TCP connection, where a TCP connection is defined by the standard rules for TCP connection establishment and termination.

* If a host has received an ECN-setup SYN packet, then it MAY send an ECN-setup SYN-ACK packet. Otherwise, it MUST NOT send an ECN-setup SYN-ACK packet.

* A host MUST NOT set ECT on data packets unless it has sent at least one ECN-setup SYN or ECN-setup SYN-ACK packet, and has received at least one ECN-setup SYN or ECN-setup SYN-ACK packet, and has sent no non-ECN-setup SYN or non-ECN-setup SYN-ACK packet. If a host has received at least one non-ECN-setup SYN or non-ECN-setup SYN-ACK packet, then it SHOULD NOT set ECT on data packets.

* If a host ever sets the ECT codepoint on a data packet, then that host MUST correctly set/clear the CWR TCP bit on all subsequent packets in the connection.

* If a host has sent at least one ECN-setup SYN or ECN-setup SYN-ACK packet, and has received no non-ECN-setup SYN or non-ECN-setup SYN-ACK packet, then if that host receives TCP data packets with ECT and CE codepoints set in the IP header, then that host MUST process these packets as specified for an ECN-capable connection.

* A host that is not willing to use ECN on a TCP connection SHOULD clear both the ECE and CWR flags in all non-ECN-setup SYN and/or SYN-ACK packets that it sends to indicate this unwillingness. Receivers MUST correctly handle all forms of the non-ECN-setup SYN and SYN-ACK packets.

* A host MUST NOT set ECT on SYN or SYN-ACK packets.

A TCP client enters TIME-WAIT state after receiving a FIN-ACK, and transitions to CLOSED state after a timeout. Many TCP implementations create a new TCP connection if they receive an in-window SYN packet during TIME-WAIT state. When a TCP host enters TIME-WAIT or CLOSED state, it should ignore any previous state about the negotiation of ECN for that connection.

6.1.1.1. Middlebox Issues

ECN introduces the use of the ECN-Echo and CWR flags in the TCP header (as shown in Figure 3) for initialization. There exist some faulty firewalls, load balancers, and intrusion detection systems in the Internet that either drop an ECN-setup SYN packet or respond with a RST, in the belief that such a packet (with these bits set) is a signature for a port-scanning tool that could be used in a denial-of-service attack. Some of the offending equipment has been identified, and a web page [FIXES] contains a list of non-compliant products and the fixes posted by the vendors, where these are available. The TBIT web page [TBIT] lists some of the web servers affected by this faulty equipment. We mention this in this document as a warning to the community of this problem.

To provide robust connectivity even in the presence of such faulty equipment, a host that receives a RST in response to the transmission of an ECN-setup SYN packet MAY resend a SYN with CWR and ECE cleared. This could result in a TCP connection being established without using ECN.

A host that receives no reply to an ECN-setup SYN within the normal SYN retransmission timeout interval MAY resend the SYN and any subsequent SYN retransmissions with CWR and ECE cleared. To overcome normal packet loss that results in the original SYN being lost, the originating host may retransmit one or more ECN-setup SYN packets before giving up and retransmitting the SYN with the CWR and ECE bits cleared.

We note that in this case, the following example scenario is possible:

(1) Host A: Sends an ECN-setup SYN.
(2) Host B: Sends an ECN-setup SYN/ACK, packet is dropped or delayed.
(3) Host A: Sends a non-ECN-setup SYN.
(4) Host B: Sends a non-ECN-setup SYN/ACK.

We note that in this case, following the procedures above, neither Host A nor Host B may set the ECT bit on data packets. Further, an important consequence of the rules for ECN setup and usage in Section 6.1.1 is that a host is forbidden from using the reception of ECT data packets as an implicit signal that the other host is ECN-capable.

6.1.1.2. Robust TCP Initialization with an Echoed Reserved Field

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 some deployed TCP implementations. There exists at least one faulty 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. The sending TCP is not mislead by a faulty TCP implementation sending a SYN-ACK packet that simply reflects the Reserved field of the incoming SYN packet.

6.1.2. The TCP Sender

For a TCP connection using ECN, new data packets are transmitted with an ECT codepoint set in the IP header. When only one ECT codepoint is needed by a sender for all packets sent on a TCP connection, ECT(0) SHOULD be used. If the sender receives an ECN-Echo (ECE) 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 SHOULD NOT increase the congestion window in response to the receipt of an ECN-Echo ACK packet.

TCP should 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, 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 acknowledgments 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. It is necessary 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 of reducing the rate further in this circumstance. Therefore, the sending TCP MUST 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 only when the retransmit timer expires.

When an ECN-Capable TCP sender reduces its congestion window for any reason (because of a retransmit timeout, a Fast Retransmit, or in response to an ECN Notification), the TCP sender sets the CWR flag in the TCP header of the first new 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.

We ensure that the "Congestion Window Reduced" information is reliably delivered to the TCP receiver. This comes about from the fact that if the new data packet carrying the CWR flag is dropped, then the TCP sender will have to again reduce its congestion window, and send another new data packet with the CWR flag set. Thus, the CWR bit in the TCP header SHOULD NOT be set on retransmitted packets.

When the TCP data sender is ready to set the CWR bit after reducing the congestion window, it SHOULD set the CWR bit only on the first new data packet that it transmits.

[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 acknowledgments, or retransmit timeouts [RFC2581]. TCP also follows the normal procedures for increasing the congestion window when it receives ACK packets without the ECN-Echo bit set [RFC2581].

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 '1' if the CE codepoint is set in any 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 sets the ECN-Echo flag in a series of ACK packets sent subsequently. The TCP receiver uses the CWR flag received from the TCP sender to determine when to stop setting the ECN-Echo flag.

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 all the ACK packets it sends (whether they acknowledge CE data packets or non-CE data packets) until it receives a CWR packet (a packet with the CWR flag set). After the receipt of the CWR packet, acknowledgments 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 suggest that the data sender reduced its congestion window at some point *after* it sent the data packet for which the CE codepoint was set.

We have already specified that a TCP sender is not required to reduce its congestion window more than once per window of data. Some care is required if the TCP sender is to avoid unnecessary reductions of the congestion window when a window of data includes both dropped packets and (marked) CE packets. This is illustrated 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) MUST be sent with the not-ECT codepoint. 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 codepoint set). For current TCP implementations, a single dropped ACK generally has only a very small effect on the TCP's sending rate.

6.1.5. Retransmitted TCP packets

This document specifies ECN-capable TCP implementations MUST NOT set either ECT codepoint (ECT(0) or ECT(1)) in the IP header for retransmitted data packets, and that the TCP data receiver SHOULD ignore the ECN field on arriving data packets that are outside of the receiver's current window. This is for greater security against denial-of-service attacks, as well as for robustness of the ECN congestion indication with packets that are dropped later in the network.

First, we note that if the TCP sender were to set an ECT codepoint on a retransmitted packet, then if an unnecessarily-retransmitted packet was later dropped in the network, the end nodes would never receive the indication of congestion from the router setting the CE codepoint. Thus, setting an ECT codepoint on retransmitted data packets is not consistent with the robust delivery of the congestion indication even for packets that are later dropped in the network.

In addition, an attacker capable of spoofing the IP source address of the TCP sender could send data packets with arbitrary sequence numbers, with the CE codepoint set in the IP header. On receiving this spoofed data packet, the TCP data receiver would determine that the data does not lie in the current receive window, and return a duplicate acknowledgement. We define an out-of-window packet at the TCP data receiver as a data packet that lies outside the receiver's current window. On receiving an out-of-window packet, the TCP data receiver has to decide whether or not to treat the CE codepoint in the packet header as a valid indication of congestion, and therefore whether to return ECN-Echo indications to the TCP data sender. If the TCP data receiver ignored the CE codepoint in an out-of-window packet, then the TCP data sender would not receive this possibly-legitimate indication of congestion from the network, resulting in a violation of end-to-end congestion control. On the other hand, if the TCP data receiver honors the CE indication in the out-of-window packet, and reports the indication of congestion to the TCP data sender, then the malicious node that created the spoofed, out-of-window packet has successfully "attacked" the TCP connection by forcing the data sender to unnecessarily reduce (halve) its congestion window. To prevent such a denial-of-service attack, we specify that a legitimate TCP data sender MUST NOT set an ECT codepoint on retransmitted data packets, and that the TCP data receiver SHOULD ignore the CE codepoint on out-of-window packets.

One drawback of not setting ECT(0) or ECT(1) on retransmitted packets is that it denies ECN protection for retransmitted packets. However, for an ECN-capable TCP connection in a fully-ECN-capable environment with mild congestion, packets should rarely be dropped due to congestion in the first place, and so instances of retransmitted packets should rarely arise. If packets are being retransmitted, then there are already packet losses (from corruption or from congestion) that ECN has been unable to prevent.

We note that if the router sets the CE codepoint for an ECN-capable data packet within a TCP connection, then the TCP connection is guaranteed to receive that indication of congestion, or to receive some other indication of congestion within the same window of data, even if this packet is dropped or reordered in the network. We consider two cases, when the packet is later retransmitted, and when the packet is not later retransmitted.

In the first case, if the packet is either dropped or delayed, and at some point retransmitted by the data sender, then the retransmission is a result of a Fast Retransmit or a Retransmit Timeout for either that packet or for some prior packet in the same window of data. In this case, because the data sender already has retransmitted this packet, we know that the data sender has already responded to an indication of congestion for some packet within the same window of data as the original packet. Thus, even if the first transmission of the packet is dropped in the network, or is delayed, if it had the CE codepoint set, and is later ignored by the data receiver as an out-of-window packet, this is not a problem, because the sender has already responded to an indication of congestion for that window of data.

In the second case, if the packet is never retransmitted by the data sender, then this data packet is the only copy of this data received by the data receiver, and therefore arrives at the data receiver as an in-window packet, regardless of how much the packet might be delayed or reordered. In this case, if the CE codepoint is set on the packet within the network, this will be treated by the data receiver as a valid indication of congestion.

6.1.6. TCP Window Probes.

When the TCP data receiver advertises a zero window, the TCP data sender sends window probes to determine if the receiver's window has increased. Window probe packets do not contain any user data except for the sequence number, which is a byte. If a window probe packet is dropped in the network, this loss is not detected by the receiver. Therefore, the TCP data sender MUST NOT set either an ECT codepoint or the CWR bit on window probe packets.

However, because window probes use exact sequence numbers, they cannot be easily spoofed in denial-of-service attacks. Therefore, if a window probe arrives with the CE codepoint set, then the receiver SHOULD respond to the ECN indications.

7. 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 codepoint in transmitted packets but do not respond to received CE packets). We argue that the addition of ECN to the IP architecture will 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 [RFC2309,FF99]. It has also been suggested 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 might seem that dropping packets in itself is 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 codepoint 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, when all flows are sharing the same packet drop rate.

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 codepoint 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.

8. Non-compliance in the Network

This section considers the issues when a router is operating, possibly maliciously, to modify either of the bits in the ECN field. We note that in IPv4, the IP header is protected from bit errors by a header checksum; this is not the case in IPv6. Thus for IPv6 the ECN field can be accidentally modified by bit errors on links or in routers without being detected by an IP header checksum.

By tampering with the bits in the ECN field, an adversary (or a broken router) could do one or more of the following: falsely report congestion, disable ECN-Capability for an individual packet, erase the ECN congestion indication, or falsely indicate ECN-Capability. Section 18 systematically examines the various cases by which the ECN field could be modified. The important criterion considered in determining the consequences of such modifications is whether it is likely to lead to poorer behavior in any dimension (throughput, delay, fairness or functionality) than if a router were to drop a packet.

The first two possible changes, falsely reporting congestion or disabling ECN-Capability for an individual packet, are no worse than if the router were to simply drop the packet. From a congestion control point of view, setting the CE codepoint in the absence of congestion by a non-compliant router would be no worse than a router dropping a packet unnecessarily. By "erasing" an ECT codepoint 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.

However, as discussed in Section 18, a router that erases the ECN congestion indication or falsely indicates ECN-Capability could potentially do more damage to the flow that if it has simply dropped the packet. A rogue or broken router that "erased" the CE codepoint 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.

Section 19 considers the potential repercussions of subverting end-to-end congestion control by either falsely indicating ECN-Capability, or by erasing the congestion indication in ECN (the CE-codepoint). We observe in Section 19 that the consequence of subverting ECN-based congestion control may lead to potential unfairness, but this is likely to be no worse than the subversion of either ECN-based or packet-based congestion control by the end nodes.

8.1. Complications Introduced by Split Paths

If a router or other network element has access to all of the packets of a flow, then that router could do no more damage to a flow by altering the ECN field than it could by simply dropping all of the packets from that flow. However, in some cases, a malicious or broken router might have access to only a subset of the packets from a flow. The question is as follows: can this router, by altering the ECN field in this subset of the packets, do more damage to that flow than if it has simply dropped that set of the packets?

This is also discussed in detail in Section 18, which concludes as follows: It is true that the adversary that has access only to a subset of packets in an aggregate might, by subverting ECN-based congestion control, be able to deny the benefits of ECN to the other packets in the aggregate. While this is undesirable, this is not a sufficient concern to result in disabling ECN.

9. Encapsulated Packets

9.1. IP packets encapsulated in IP

The encapsulation of IP packet headers in tunnels is used in many places, including IPsec and IP in IP [RFC2003]. This section considers issues related to interactions between ECN and IP tunnels, and specifies two alternative solutions. This discussion is complemented by RFC 2983's discussion of interactions between Differentiated Services and IP tunnels of various forms [RFC 2983], as Differentiated Services uses the remaining six bits of the IP header octet that is used by ECN (see Figure 2 in Section 5).

Some IP tunnel modes are based on adding a new "outer" IP header that encapsulates the original, or "inner" IP header and its associated packet. In many cases, the new "outer" IP header may be added and removed at intermediate points along a connection, enabling the network to establish a tunnel without requiring endpoint participation. We denote tunnels that specify that the outer header be discarded at tunnel egress as "simple tunnels".

ECN uses the ECN field in the IP header for signaling between routers and connection endpoints. ECN interacts with IP tunnels based on the treatment of the ECN field in the IP header. In simple IP tunnels the octet containing the ECN field is copied or mapped from the inner IP header to the outer IP header at IP tunnel ingress, and the outer header's copy of this field is discarded at IP tunnel egress. If the outer header were to be simply discarded without taking care to deal with the ECN field, and an ECN-capable router were to set the CE

(Congestion Experienced) codepoint within a packet in a simple IP tunnel, this indication would be discarded at tunnel egress, losing the indication of congestion.

Thus, the use of ECN over simple IP tunnels would result in routers attempting to use the outer IP header to signal congestion to endpoints, but those congestion warnings never arriving because the outer header is discarded at the tunnel egress point. This problem was encountered with ECN and IPsec in tunnel mode, and RFC 2481 recommended that ECN not be used with the older simple IPsec tunnels in order to avoid this behavior and its consequences. When ECN becomes widely deployed, then simple tunnels likely to carry ECN-capable traffic will have to be changed. If ECN-capable traffic is carried by a simple tunnel through a congested, ECN-capable router, this could result in subsequent packets being dropped for this flow as the average queue size increases at the congested router, as discussed in Section 8 above.

From a security point of view, the use of ECN in the outer header of an IP tunnel might raise security concerns because an adversary could tamper with the ECN information that propagates beyond the tunnel endpoint. Based on an analysis in Sections 18 and 19 of these concerns and the resultant risks, our overall approach is to make support for ECN an option for IP tunnels, so that an IP tunnel can be specified or configured either to use ECN or not to use ECN in the outer header of the tunnel. Thus, in environments or tunneling protocols where the risks of using ECN are judged to outweigh its benefits, the tunnel can simply not use ECN in the outer header. Then the only indication of congestion experienced at routers within the tunnel would be through packet loss.

The result is that there are two viable options for the behavior of ECN-capable connections over an IP tunnel, including IPsec tunnels:

* A limited-functionality option in which ECN is preserved in the inner header, but disabled in the outer header. The only mechanism available for signaling congestion occurring within the tunnel in this case is dropped packets.

* A full-functionality option that supports ECN in both the inner and outer headers, and propagates congestion warnings from nodes within the tunnel to endpoints.

Support for these options requires varying amounts of changes to IP header processing at tunnel ingress and egress. A small subset of these changes sufficient to support only the limited-functionality option would be sufficient to eliminate any incompatibility between ECN and IP tunnels.

One goal of this document is to give guidance about the tradeoffs between the limited-functionality and full-functionality options. A full discussion of the potential effects of an adversary's modifications of the ECN field is given in Sections 18 and 19.

9.1.1. The Limited-functionality and Full-functionality Options

The limited-functionality option for ECN encapsulation in IP tunnels is for the not-ECT codepoint to be set in the outside (encapsulating) header regardless of the value of the ECN field in the inside (encapsulated) header. With this option, the ECN field in the inner header is not altered upon de-capsulation. The disadvantage of this approach is that the flow does not have ECN support for that part of the path that is using IP tunneling, even if the encapsulated packet (from the original TCP sender) is ECN-Capable. That is, if the encapsulated packet arrives at a congested router that is ECN-capable, and the router can decide to drop or mark the packet as an indication of congestion to the end nodes, the router will not be permitted to set the CE codepoint in the packet header, but instead will have to drop the packet.

The full-functionality option for ECN encapsulation is to copy the ECN codepoint of the inside header to the outside header on encapsulation if the inside header is not-ECT or ECT, and to set the ECN codepoint of the outside header to ECT(0) if the ECN codepoint of the inside header is CE. On decapsulation, if the CE codepoint is set on the outside header, then the CE codepoint is also set in the inner header. Otherwise, the ECN codepoint on the inner header is left unchanged. That is, for full ECN support the encapsulation and decapsulation processing involves the following: At tunnel ingress, the full-functionality option sets the ECN codepoint in the outer header. If the ECN codepoint in the inner header is not-ECT or ECT, then it is copied to the ECN codepoint in the outer header. If the ECN codepoint in the inner header is CE, then the ECN codepoint in the outer header is set to ECT(0). Upon decapsulation at the tunnel egress, the full-functionality option sets the CE codepoint in the inner header if the CE codepoint is set in the outer header. Otherwise, no change is made to this field of the inner header.

With the full-functionality option, a flow can take advantage of ECN in those parts of the path that might use IP tunneling. The disadvantage of the full-functionality option from a security perspective is that the IP tunnel cannot protect the flow from certain modifications to the ECN bits in the IP header within the tunnel. The potential dangers from modifications to the ECN bits in the IP header are described in detail in Sections 18 and 19.

(1) An IP tunnel MUST modify the handling of the DS field octet at IP tunnel endpoints by implementing either the limited-functionality or the full-functionality option.

(2) Optionally, an IP tunnel MAY enable the endpoints of an IP tunnel to negotiate the choice between the limited-functionality and the full-functionality option for ECN in the tunnel.

The minimum required to make ECN usable with IP tunnels is the limited-functionality option, which prevents ECN from being enabled in the outer header of the tunnel. Full support for ECN requires the use of the full-functionality option. If there are no optional mechanisms for the tunnel endpoints to negotiate a choice between the limited-functionality or full-functionality option, there can be a pre-existing agreement between the tunnel endpoints about whether to support the limited-functionality or the full-functionality ECN option.

All IP tunnels MUST implement the limited-functionality option, and SHOULD support the full-functionality option.

In addition, it is RECOMMENDED that packets with the CE codepoint in the outer header be dropped if they arrive at the tunnel egress point for a tunnel that uses the limited-functionality option, or for a tunnel that uses the full-functionality option but for which the not-ECT codepoint is set in the inner header. This is motivated by backwards compatibility and to ensure that no unauthorized modifications of the ECN field take place, and is discussed further in the next Section (9.1.2).

9.1.2. Changes to the ECN Field within an IP Tunnel.

The presence of a copy of the ECN field in the inner header of an IP tunnel mode packet provides an opportunity for detection of unauthorized modifications to the ECN field in the outer header. Comparison of the ECT fields in the inner and outer headers falls into two categories for implementations that conform to this document:

* If the IP tunnel uses the full-functionality option, then the not-ECT codepoint should be set in the outer header if and only if it is also set in the inner header.

* If the tunnel uses the limited-functionality option, then the not-ECT codepoint should be set in the outer header.

Receipt of a packet not satisfying the appropriate condition could be a cause of concern.

Consider the case of an IP tunnel where the tunnel ingress point has not been updated to this document's requirements, while the tunnel egress point has been updated to support ECN. In this case, the IP tunnel is not explicitly configured to support the full-functionality ECN option. However, the tunnel ingress point is behaving identically to a tunnel ingress point that supports the full-functionality option. If packets from an ECN-capable connection use this tunnel, the ECT codepoint will be set in the outer header at the tunnel ingress point. Congestion within the tunnel may then result in ECN-capable routers setting CE in the outer header. Because the tunnel has not been explicitly configured to support the full-functionality option, the tunnel egress point expects the not-ECT codepoint to be set in the outer header. When an ECN-capable tunnel egress point receives a packet with the ECT or CE codepoint in the outer header, in a tunnel that has not been configured to support the full-functionality option, that packet should be processed, according to whether the CE codepoint was set, as follows. It is RECOMMENDED that on a tunnel that has not been configured to support the full-functionality option, packets should be dropped at the egress point if the CE codepoint is set in the outer header but not in the inner header, and should be forwarded otherwise.

An IP tunnel cannot provide protection against erasure of congestion indications based on changing the ECN codepoint from CE to ECT. The erasure of congestion indications may impact the network and other flows in ways that would not be possible in the absence of ECN. It is important to note that erasure of congestion indications can only be performed to congestion indications placed by nodes within the tunnel; the copy of the ECN field in the inner header preserves congestion notifications from nodes upstream of the tunnel ingress (unless the inner header is also erased). If erasure of congestion notifications is judged to be a security risk that exceeds the congestion management benefits of ECN, then tunnels could be specified or configured to use the limited-functionality option.

9.2. IPsec Tunnels

IPsec supports secure communication over potentially insecure network components such as intermediate routers. IPsec protocols support two operating modes, transport mode and tunnel mode, that span a wide range of security requirements and operating environments. Transport mode security protocol header(s) are inserted between the IP (IPv4 or IPv6) header and higher layer protocol headers (e.g., TCP), and hence transport mode can only be used for end-to-end security on a connection. IPsec tunnel mode is based on adding a new "outer" IP header that encapsulates the original, or "inner" IP header and its associated packet. Tunnel mode security headers are inserted between these two IP headers. In contrast to transport mode, the new "outer"

IP header and tunnel mode security headers can be added and removed at intermediate points along a connection, enabling security gateways to secure vulnerable portions of a connection without requiring endpoint participation in the security protocols. An important aspect of tunnel mode security is that in the original specification, the outer header is discarded at tunnel egress, ensuring that security threats based on modifying the IP header do not propagate beyond that tunnel endpoint. Further discussion of IPsec can be found in [RFC2401].

The IPsec protocol as originally defined in [ESP, AH] required that the inner header's ECN field not be changed by IPsec decapsulation processing at a tunnel egress node; this would have ruled out the possibility of full-functionality mode for ECN. At the same time, this would ensure 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.

In principle, permitting the use of ECN functionality in the outer header of an IPsec tunnel raises security concerns because an adversary could tamper with the information that propagates beyond the tunnel endpoint. Based on an analysis (included in Sections 18 and 19) of these concerns and the associated risks, our overall approach has been to provide configuration support for IPsec changes to remove the conflict with ECN.

In particular, in tunnel mode the IPsec tunnel MUST support the limited-functionality option outlined in Section 9.1.1, and SHOULD support the full-functionality option outlined in Section 9.1.1.

This makes permission to use ECN functionality in the outer header of an IPsec tunnel a configurable part of the corresponding IPsec Security Association (SA), so that it can be disabled in situations where the risks are judged to outweigh the benefits. The result is that an IPsec security administrator is presented with two alternatives for the behavior of ECN-capable connections within an IPsec tunnel, the limited-functionality alternative and full-functionality alternative described earlier.

In addition, this document specifies how the endpoints of an IPsec tunnel could negotiate enabling ECN functionality in the outer headers of that tunnel based on security policy. The ability to negotiate ECN usage between tunnel endpoints would enable a security administrator to disable ECN in situations where she believes the risks (e.g., of lost congestion notifications) outweigh the benefits of ECN.

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 IPsec tunnel should be run in limited-functionality mode.

9.2.1. Negotiation between Tunnel Endpoints

This section describes the detailed changes to enable usage of ECN over IPsec tunnels, including the negotiation of ECN support between tunnel endpoints. This is supported by three changes to IPsec:

* An optional Security Association Database (SAD) field indicating whether tunnel encapsulation and decapsulation processing allows or forbids ECN usage in the outer IP header.

* An optional Security Association Attribute that enables negotiation of this SAD field between the two endpoints of an SA that supports tunnel mode.

* Changes to tunnel mode encapsulation and decapsulation processing to allow or forbid ECN usage in the outer IP header based on the value of the SAD field. When ECN usage is allowed in the outer IP header, the ECT codepoint is set in the outer header for ECN-capable connections and congestion notifications (indicated by the CE codepoint) from such connections are propagated to the inner header at tunnel egress.

If negotiation of ECN usage is implemented, then the SAD field SHOULD also be implemented. On the other hand, negotiation of ECN usage is OPTIONAL in all cases, even for implementations that support the SAD field. The encapsulation and decapsulation processing changes are REQUIRED, but MAY be implemented without the other two changes by assuming that ECN usage is always forbidden. The full-functionality alternative for ECN usage over IPsec tunnels consists of the SAD field and the full version of encapsulation and decapsulation processing changes, with or without the OPTIONAL negotiation support. The limited-functionality alternative consists of a subset of the encapsulation and decapsulation changes that always forbids ECN usage.

These changes are covered further in the following three subsections.

9.2.1.1. ECN Tunnel Security Association Database Field

Full ECN functionality adds a new field to the SAD (see [RFC2401]):

ECN Tunnel: allowed or forbidden.

Indicates whether ECN-capable connections using this SA in tunnel mode are permitted to receive ECN congestion notifications for congestion occurring within the tunnel. The allowed value enables ECN congestion notifications. The forbidden value disables such notifications, causing all congestion to be indicated via dropped packets.

[OPTIONAL. The value of this field SHOULD be assumed to be "forbidden" in implementations that do not support it.]

If this attribute is implemented, then the SA specification in a Security Policy Database (SPD) entry MUST support a corresponding attribute, and this SPD attribute MUST be covered by the SPD administrative interface (currently described in Section 4.4.1 of [RFC2401]).

9.2.1.2. ECN Tunnel Security Association Attribute

A new IPsec Security Association Attribute is defined to enable the support for ECN congestion notifications based on the outer IP header to be negotiated for IPsec tunnels (see [RFC2407]). This attribute is OPTIONAL, although implementations that support it SHOULD also support the SAD field defined in Section 9.2.1.1.

Attribute Type

class	value	type
ECN Tunnel	10	Basic

The IPsec SA Attribute value 10 has been allocated by IANA to indicate that the ECN Tunnel SA Attribute is being negotiated; the type of this attribute is Basic (see Section 4.5 of [RFC2407]). The Class Values are used to conduct the negotiation. See [RFC2407, RFC2408, RFC2409] for further information including encoding formats and requirements for negotiating this SA attribute.

Class Values

ECN Tunnel

Specifies whether ECN functionality is allowed to be used with Tunnel Encapsulation Mode. This affects tunnel encapsulation and decapsulation processing - see Section 9.2.1.3.

```
RESERVED          0
Allowed           1
Forbidden         2
```

Values 3-61439 are reserved to IANA. Values 61440-65535 are for private use.

If unspecified, the default shall be assumed to be Forbidden.

ECN Tunnel is a new SA attribute, and hence initiators that use it can expect to encounter responders that do not understand it, and therefore reject proposals containing it. For backwards compatibility with such implementations initiators SHOULD always also include a proposal without the ECN Tunnel attribute to enable such a responder to select a transform or proposal that does not contain the ECN Tunnel attribute. RFC 2407 currently requires responders to reject all proposals if any proposal contains an unknown attribute; this requirement is expected to be changed to require a responder not to select proposals or transforms containing unknown attributes.

9.2.1.3. Changes to IPsec Tunnel Header Processing

For full ECN support, the encapsulation and decapsulation processing for the IPv4 TOS field and the IPv6 Traffic Class field are changed from that specified in [RFC2401] to the following:

```
                     <-- How Outer Hdr Relates to Inner Hdr -->
                     Outer Hdr at                 Inner Hdr at
IPv4                 Encapsulator                 Decapsulator
  Header fields:     --------------------         ------------
    DS Field         copied from inner hdr (5)    no change
    ECN Field        constructed (7)              constructed (8)

IPv6
  Header fields:
    DS Field         copied from inner hdr (6)    no change
    ECN Field        constructed (7)              constructed (8)
```

(5)(6) If the packet will immediately enter a domain for which the DSCP value in the outer header is not appropriate, that value MUST be mapped to an appropriate value for the domain [RFC 2474]. Also see [RFC 2475] for further information.

(7) If the value of the ECN Tunnel field in the SAD entry for this SA is "allowed" and the ECN field in the inner header is set to any value other than CE, copy this ECN field to the outer header. If the ECN field in the inner header is set to CE, then set the ECN field in the outer header to ECT(0).

(8) If the value of the ECN tunnel field in the SAD entry for this SA is "allowed" and the ECN field in the inner header is set to ECT(0) or ECT(1) and the ECN field in the outer header is set to CE, then copy the ECN field from the outer header to the inner header. Otherwise, make no change to the ECN field in the inner header.

(5) and (6) are identical to match usage in [RFC2401], although they are different in [RFC2401].

The above description applies to implementations that support the ECN Tunnel field in the SAD; such implementations MUST implement this processing instead of the processing of the IPv4 TOS octet and IPv6 Traffic Class octet defined in [RFC2401]. This constitutes the full-functionality alternative for ECN usage with IPsec tunnels.

An implementation that does not support the ECN Tunnel field in the SAD MUST implement this processing by assuming that the value of the ECN Tunnel field of the SAD is "forbidden" for every SA. In this case, the processing of the ECN field reduces to:

(7) Set the ECN field to not-ECT in the outer header.
(8) Make no change to the ECN field in the inner header.

This constitutes the limited functionality alternative for ECN usage with IPsec tunnels.

For backwards compatibility, packets with the CE codepoint set in the outer header SHOULD be dropped if they arrive on an SA that is using the limited-functionality option, or that is using the full-functionality option with the not-ECN codepoint set in the inner header.

9.2.2. Changes to the ECN Field within an IPsec Tunnel.

If the ECN Field is changed inappropriately within an IPsec tunnel, and this change is detected at the tunnel egress, then the receipt of a packet not satisfying the appropriate condition for its SA is an auditable event. An implementation MAY create audit records with per-SA counts of incorrect packets over some time period rather than creating an audit record for each erroneous packet. Any such audit record SHOULD contain the headers from at least one erroneous packet, but need not contain the headers from every packet represented by the entry.

9.2.3. Comments for IPsec Support

Substantial comments were received on two areas of this document during review by the IPsec working group. This section describes these comments and explains why the proposed changes were not incorporated.

The first comment indicated that per-node configuration is easier to implement than per-SA configuration. After serious thought and despite some initial encouragement of per-node configuration, it no longer seems to be a good idea. The concern is that as ECN-awareness is progressively deployed in IPsec, many ECN-aware IPsec implementations will find themselves communicating with a mixture of ECN-aware and ECN-unaware IPsec tunnel endpoints. In such an environment with per-node configuration, the only reasonable thing to do is forbid ECN usage for all IPsec tunnels, which is not the desired outcome.

In the second area, several reviewers noted that SA negotiation is complex, and adding to it is non-trivial. One reviewer suggested using ICMP after tunnel setup as a possible alternative. The addition to SA negotiation in this document is OPTIONAL and will remain so; implementers are free to ignore it. The authors believe that the assurance it provides can be useful in a number of situations. In practice, if this is not implemented, it can be deleted at a subsequent stage in the standards process. Extending ICMP to negotiate ECN after tunnel setup is more complex than extending SA attribute negotiation. Some tunnels do not permit traffic to be addressed to the tunnel egress endpoint, hence the ICMP packet would have to be addressed to somewhere else, scanned for by the egress endpoint, and discarded there or at its actual destination. In addition, ICMP delivery is unreliable, and hence there is a possibility of an ICMP packet being dropped, entailing the invention of yet another ack/retransmit mechanism. It seems better simply to specify an OPTIONAL extension to the existing SA negotiation mechanism.

9.3. IP packets encapsulated in non-IP Packet Headers.

A different set of issues are raised, relative to ECN, when IP packets are encapsulated in tunnels with non-IP packet headers. This occurs with MPLS [MPLS], GRE [GRE], L2TP [L2TP], and PPTP [PPTP]. For these protocols, there is no conflict with ECN; it is just that ECN cannot be used within the tunnel unless an ECN codepoint can be specified for the header of the encapsulating protocol. Earlier work considered a preliminary proposal for incorporating ECN into MPLS, and proposals for incorporating ECN into GRE, L2TP, or PPTP will be considered as the need arises.

10. Issues Raised by Monitoring and Policing Devices

One possibility is that monitoring and policing devices (or more informally, "penalty boxes") will be installed in the network to monitor whether best-effort flows are appropriately responding to congestion, and to preferentially drop packets from flows determined not to be using adequate end-to-end congestion control procedures.

We recommend that any "penalty box" that detects a flow or an aggregate of flows that is not responding to end-to-end congestion control first change from marking to dropping packets from that flow, before taking any additional action to restrict the bandwidth available to that flow. Thus, initially, the router may drop packets in which the router would otherwise would have set the CE codepoint. This could include dropping those arriving packets for that flow that are ECN-Capable and that already have the CE codepoint set. In this way, any congestion indications seen by that router for that flow will be guaranteed to also be seen by the end nodes, even in the presence of malicious or broken routers elsewhere in the path. If we assume that the first action taken at any "penalty box" for an ECN-capable flow will be to drop packets instead of marking them, then there is no way that an adversary that subverts ECN-based end-to-end congestion control can cause a flow to be characterized as being non-cooperative and placed into a more severe action within the "penalty box".

The monitoring and policing devices that are actually deployed could fall short of the 'ideal' monitoring device described above, in that the monitoring is applied not to a single flow, but to an aggregate of flows (e.g., those sharing a single IPsec tunnel). In this case, the switch from marking to dropping would apply to all of the flows in that aggregate, denying the benefits of ECN to the other flows in the aggregate also. At the highest level of aggregation, another form of the disabling of ECN happens even in the absence of monitoring and policing devices, when ECN-Capable RED queues switch from marking to dropping packets as an indication of congestion when the average queue size has exceeded some threshold.

11. Evaluations of ECN

11.1. Related Work Evaluating ECN

This section discusses some of the related work evaluating the use of ECN. The ECN Web Page [ECN] has pointers to other papers, as well as to implementations of ECN.

[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 the appendix of this document, and the second is addressed by the addition of the CWR flag in the TCP header.

Experimental evaluations of ECN include [RFC2884,K98]. The conclusions of [K98] and [RFC2884] 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 (with 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.

11.2. A Discussion of the ECN nonce.

The use of two ECT codepoints, ECT(0) and ECT(1), can provide a one-bit ECN nonce in packet headers [SCWA99]. The primary motivation for this is the desire to allow mechanisms for the data sender to verify that network elements are not erasing the CE codepoint, and that data receivers are properly reporting to the sender the receipt of packets with the CE codepoint set, as required by the transport protocol. This section discusses issues of backwards compatibility with IP ECN implementations in routers conformant with RFC 2481, in which only one ECT codepoint was defined. We do not believe that the incremental deployment of ECN implementations that understand the ECT(1) codepoint will cause significant operational problems. This is particularly likely to be the case when the deployment of the ECT(1) codepoint begins with routers, before the ECT(1) codepoint starts to be used by end-nodes.

11.2.1. The Incremental Deployment of ECT(1) in Routers.

ECN has been an Experimental standard since January 1999, and there are already implementations of ECN in routers that do not understand the ECT(1) codepoint. When the use of the ECT(1) codepoint is standardized for TCP or for other transport protocols, this could mean that a data sender is using the ECT(1) codepoint, but that this codepoint is not understood by a congested router on the path.

If allowed by the transport protocol, a data sender would be free not to make use of ECT(1) at all, and to send all ECN-capable packets with the codepoint ECT(0). However, if an ECN-capable sender is using ECT(1), and the congested router on the path did not understand the ECT(1) codepoint, then the router would end up marking some of the ECT(0) packets, and dropping some of the ECT(1) packets, as indications of congestion. Since TCP is required to react to both marked and dropped packets, this behavior of dropping packets that could have been marked poses no significant threat to the network, and is consistent with the overall approach to ECN that allows routers to determine when and whether to mark packets as they see fit (see Section 5).

12. Summary of changes required in IP and TCP

This document specified two bits in the IP header to be used for ECN. The not-ECT codepoint indicates that the transport protocol will ignore the CE codepoint. This is the default value for the ECN codepoint. The ECT codepoints indicate that the transport protocol is willing and able to participate in ECN.

The router sets the CE codepoint to indicate congestion to the end nodes. The CE codepoint in a packet header MUST NOT be reset by a router.

TCP requires three changes for ECN, a setup phase and two new flags in the TCP header. The ECN-Echo flag is used by the data receiver to inform the data sender of a received CE packet. The Congestion Window Reduced (CWR) flag is used by the data sender to inform the data receiver that the congestion window has been reduced.

When ECN (Explicit Congestion Notification) is used, it is required that congestion indications generated within an IP tunnel not be lost at the tunnel egress. We specified a minor modification to the IP protocol's handling of the ECN field during encapsulation and de-capsulation to allow flows that will undergo IP tunneling to use ECN.

Two options for ECN in tunnels were specified:

1) A limited-functionality option that does not use ECN inside the IP tunnel, by setting the ECN field in the outer header to not-ECT, and not altering the inner header at the time of decapsulation.

2) The full-functionality option, which sets the ECN field in the outer header to either not-ECT or to one of the ECT codepoints, depending on the ECN field in the inner header. At decapsulation, if the CE codepoint is set in the outer header, and the inner header is set to one of the ECT codepoints, then the CE codepoint is copied to the inner header.

For IPsec tunnels, this document also defines an optional IPsec Security Association (SA) attribute that enables negotiation of ECN usage within IPsec tunnels and an optional field in the Security Association Database to indicate whether ECN is permitted in tunnel mode on a SA. The required changes to IPsec tunnels for ECN usage modify RFC 2401 [RFC2401], which defines the IPsec architecture and specifies some aspects of its implementation. The new IPsec SA attribute is in addition to those already defined in Section 4.5 of [RFC2407].

This document obsoletes RFC 2481, "A Proposal to add Explicit Congestion Notification (ECN) to IP", which defined ECN as an Experimental Protocol for the Internet Community. The rest of this section describes the relationship between this document and its predecessor.

RFC 2481 included a brief discussion of the use of ECN with encapsulated packets, and noted that for the IPsec specifications at the time (January 1999), flows could not safely use ECN if they were to traverse IPsec tunnels. RFC 2481 also described the changes that could be made to IPsec tunnel specifications to made them compatible with ECN.

This document also incorporates work that was done after RFC 2481. First was to describe the changes to IPsec tunnels in detail, and extensively discuss the security implications of ECN (now included as Sections 18 and 19 of this document). Second was to extend the discussion of IPsec tunnels to include all IP tunnels. Because older IP tunnels are not compatible with a flow's use of ECN, the deployment of ECN in the Internet will create strong pressure for older IP tunnels to be updated to an ECN-compatible version, using either the limited-functionality or the full-functionality option.

This document does not address the issue of including ECN in non-IP tunnels such as MPLS, GRE, L2TP, or PPTP. An earlier preliminary document about adding ECN support to MPLS was not advanced.

A third new piece of work after RFC2481 was to describe the ECN procedure with retransmitted data packets, that an ECT codepoint should not be set on retransmitted data packets. The motivation for this additional specification is to eliminate a possible avenue for denial-of-service attacks on an existing TCP connection. Some prior deployments of ECN-capable TCP might not conform to the (new) requirement not to set an ECT codepoint on retransmitted packets; we do not believe this will cause significant problems in practice.

This document also expands slightly on the specification of the use of SYN packets for the negotiation of ECN. While some prior deployments of ECN-capable TCP might not conform to the requirements specified in this document, we do not believe that this will lead to any performance or compatibility problems for TCP connections with a combination of TCP implementations at the endpoints.

This document also includes the specification of the ECT(1) codepoint, which may be used by TCP as part of the implementation of an ECN nonce.

13. Conclusions

Given the current effort to implement AQM, we believe this is the right time to deploy 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).

We examined the consequence of modifications of the ECN field within the network, analyzing all the opportunities for an adversary to change the ECN field. In many cases, the change to the ECN field is no worse than dropping a packet. However, we noted that some changes have the more serious consequence of subverting end-to-end congestion control. However, we point out that even then the potential damage is limited, and is similar to the threat posed by end-systems intentionally failing to cooperate with end-to-end congestion control.

14. Acknowledgements

Many people have made contributions to this work and this document, including many that we have not managed to directly acknowledge in this document. In addition, 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.

Email discussions with a number of people, including Dax Kelson, Alexey Kuznetsov, Jamal Hadi-Salim, and Venkat Venkatsubra, have addressed the issues raised by non-conformant equipment in the Internet that does not respond to TCP SYN packets with the ECE and CWR flags set. We thank Mark Handley, Jitentra Padhye, and others for discussions on the TCP initialization procedures.

The discussion of ECN and IP tunnel considerations draws heavily on related discussions and documents from the Differentiated Services Working Group. We thank Tabassum Bint Haque from Dhaka, Bangladesh, for feedback on IP tunnels. We thank Derrell Piper and Kero Tivinen for proposing modifications to RFC 2407 that improve the usability of negotiating the ECN Tunnel SA attribute.

We thank David Wetherall, David Ely, and Neil Spring for the proposal for the ECN nonce. We also thank Stefan Savage for discussions on this issue. We thank Bob Briscoe and Jon Crowcroft for raising the issue of fragmentation in IP, on alternate semantics for the fourth ECN codepoint, and several other topics. We thank Richard Wendland for feedback on several issues in the document.

We also thank the IESG, and in particular the Transport Area Directors over the years, for their feedback and their work towards the standardization of ECN.

15. References

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

[ECN] "The ECN Web Page", URL "http://www.aciri.org/floyd/ecn.html". Reference for informational purposes only.

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

[FIXES] ECN-under-Linux Unofficial Vendor Support Page, URL "http://gtf.org/garzik/ecn/". Reference for informational purposes only.

[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.

[Floyd94] Floyd, S., "TCP and Explicit Congestion Notification", ACM Computer Communication Review, V. 24 N. 5, October 1994, p. 10-23.

[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. Reference for informational purposes only.

[FF99] Floyd, S., and Fall, K., "Promoting the Use of End-to-End Congestion Control in the Internet", IEEE/ACM Transactions on Networking, August 1999.

[FRED] Lin, D., and Morris, R., "Dynamics of Random Early Detection", SIGCOMM '97, September 1997.

[GRE] Hanks, S., Li, T., Farinacci, D. and P. Traina, "Generic Routing Encapsulation (GRE)", RFC 1701, October 1994.

[Jacobson88] V. Jacobson, "Congestion Avoidance and Control", Proc. ACM SIGCOMM '88, pp. 314-329.

[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".

[K98] Krishnan, H., "Analyzing Explicit Congestion Notification (ECN) benefits for TCP", Master's thesis, UCLA, 1998. Citation for acknowledgement purposes only.

[L2TP] Townsley, W., Valencia, A., Rubens, A., Pall, G., Zorn, G. and B. Palter, "Layer Two Tunneling Protocol "L2TP"", RFC 2661, August 1999.

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

[MPLS] Awduche, D., Malcolm, J., Agogbua, J., O'Dell, M. and J. McManus, Requirements for Traffic Engineering Over MPLS, RFC 2702, September 1999.

[PPTP] Hamzeh, K., Pall, G., Verthein, W., Taarud, J., Little, W. and G. Zorn, "Point-to-Point Tunneling Protocol (PPTP)", RFC 2637, July 1999.

[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.

[RFC1701] Hanks, S., Li, T., Farinacci, D. and P. Traina, "Generic Routing Encapsulation (GRE)", RFC 1701, October 1994.

[RFC1702] Hanks, S., Li, T., Farinacci, D. and P. Traina, "Generic Routing Encapsulation over IPv4 networks", RFC 1702, October 1994.

[RFC2003] Perkins, C., "IP Encapsulation within IP", RFC 2003, October 1996.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC2309] Braden, B., et al., "Recommendations on Queue Management and Congestion Avoidance in the Internet", RFC 2309, April 1998.

[RFC2401] Kent, S. and R. Atkinson, Security Architecture for the Internet Protocol, RFC 2401, November 1998.

[RFC2407] Piper, D., "The Internet IP Security Domain of Interpretation for ISAKMP", RFC 2407, November 1998.

[RFC2408] Maughan, D., Schertler, M., Schneider, M. and J. Turner, "Internet Security Association and Key Management Protocol (ISAKMP)", RFC 2409, November 1998.

[RFC2409] Harkins D. and D. Carrel, "The Internet Key Exchange (IKE)", RFC 2409, November 1998.

[RFC2474] 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.

[RFC2475] Blake, S., Black, D., Carlson, M., Davies, E., Wang, Z. and W. Weiss, "An Architecture for Differentiated Services", RFC 2475, December 1998.

[RFC2481] Ramakrishnan K. and S. Floyd, "A Proposal to add Explicit Congestion Notification (ECN) to IP", RFC 2481, January 1999.

[RFC2581] Alman, M., Paxson, V. and W. Stevens, "TCP Congestion Control", RFC 2581, April 1999.

[RFC2884] Hadi Salim, J. and U. Ahmed, "Performance Evaluation of Explicit Congestion Notification (ECN) in IP Networks", RFC 2884, July 2000.

[RFC2983] Black, D., "Differentiated Services and Tunnels", RFC2983, October 2000.

[RFC2780] Bradner S. and V. Paxson, "IANA Allocation Guidelines For Values In the Internet Protocol and Related Headers", BCP 37, RFC 2780, March 2000.

[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.

[SCWA99] Stefan Savage, Neal Cardwell, David Wetherall, and Tom Anderson, TCP Congestion Control with a Misbehaving Receiver, ACM Computer Communications Review, October 1999.

[TBIT] Jitendra Padhye and Sally Floyd, "Identifying the TCP Behavior of Web Servers", ICSI TR-01-002, February 2001. URL "http://www.aciri.org/tbit/".

16. Security Considerations

Security considerations have been discussed in Sections 7, 8, 18, and 19.

17. 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 codepoint was set would work as follows: Let HC be the original header checksum for an ECT(0) packet, and let HC' be the new header checksum after the CE bit has been set. That is, the ECN field has changed from '10' to '11'. 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
```

A similar incremental updating of the IPv4 checksum can be carried out when the ECN field is changed from ECT(1) to CE, that is, from ' 01' to '11'.

18. Possible Changes to the ECN Field in the Network

This section discusses in detail possible changes to the ECN field in the network, such as falsely reporting congestion, disabling ECN-Capability for an individual packet, erasing the ECN congestion indication, or falsely indicating ECN-Capability.

18.1. Possible Changes to the IP Header

18.1.1. Erasing the Congestion Indication

First, we consider the changes that a router could make that would result in effectively erasing the congestion indication after it had been set by a router upstream. The convention followed is: ECN codepoint of received packet -> ECN codepoint of packet transmitted.

Replacing the CE codepoint with the ECT(0) or ECT(1) codepoint effectively erases the congestion indication. However, with the use of two ECT codepoints, a router erasing the CE codepoint has no way to know whether the original ECT codepoint was ECT(0) or ECT(1). Thus, it is possible for the transport protocol to deploy mechanisms to detect such erasures of the CE codepoint.

The consequence of the erasure of the CE codepoint for the upstream router is that there is a potential for congestion to build for a time, because the congestion indication does not reach the source. However, the packet would be received and acknowledged.

The potential effect of erasing the congestion indication is complex, and is discussed in depth in Section 19 below. Note that the effect of erasing the congestion indication is different from dropping a packet in the network. When a data packet is dropped, the drop is detected by the TCP sender, and interpreted as an indication of congestion. Similarly, if a sufficient number of consecutive acknowledgement packets are dropped, causing the cumulative acknowledgement field not to be advanced at the sender, the sender is limited by the congestion window from sending additional packets, and ultimately the retransmit timer expires.

In contrast, a systematic erasure of the CE bit by a downstream router can have the effect of causing a queue buildup at an upstream router, including the possible loss of packets due to buffer overflow. There is a potential of unfairness in that another flow that goes through the congested router could react to the CE bit set while the flow that has the CE bit erased could see better performance. The limitations on this potential unfairness are discussed in more detail in Section 19 below.

The last of the three changes is to replace the CE codepoint with the not-ECT codepoint, thus erasing the congestion indication and disabling ECN-Capability at the same time.

The 'erasure' of the congestion indication is only effective if the packet does not end up being marked or dropped again by a downstream router. If the CE codepoint is replaced by an ECT codepoint, the packet remains ECN-Capable, and could be either marked or dropped by a downstream router as an indication of congestion. If the CE codepoint is replaced by the not-ECT codepoint, the packet is no longer ECN-capable, and can therefore be dropped but not marked by a downstream router as an indication of congestion.

18.1.2. Falsely Reporting Congestion

This change is to set the CE codepoint when an ECT codepoint was already set, even though there was no congestion. This change does not affect the treatment of that packet along the rest of the path. In particular, a router does not examine the CE codepoint in deciding whether to drop or mark an arriving packet.

However, this could result in the application unnecessarily invoking end-to-end congestion control, and reducing its arrival rate. By itself, this is no worse (for the application or for the network) than if the tampering router had actually dropped the packet.

18.1.3. Disabling ECN-Capability

This change is to turn off the ECT codepoint of a packet. This means that if the packet later encounters congestion (e.g., by arriving to a RED queue with a moderate average queue size), it will be dropped instead of being marked. By itself, this is no worse (for the application) than if the tampering router had actually dropped the packet. The saving grace in this particular case is that there is no congested router upstream expecting a reaction from setting the CE bit.

18.1.4. Falsely Indicating ECN-Capability

This change would incorrectly label a packet as ECN-Capable. The packet may have been sent either by an ECN-Capable transport or a transport that is not ECN-Capable.

If the packet later encounters moderate congestion at an ECN-Capable router, the router could set the CE codepoint instead of dropping the packet. If the transport protocol in fact is not ECN-Capable, then the transport will never receive this indication of congestion, and will not reduce its sending rate in response. The potential consequences of falsely indicating ECN-capability are discussed further in Section 19 below.

If the packet never later encounters congestion at an ECN-Capable router, then the first of these two changes would have no effect, other than possibly interfering with the use of the ECN nonce by the transport protocol. The last change, however, would have the effect of giving false reports of congestion to a monitoring device along the path. If the transport protocol is ECN-Capable, then this change could also have an effect at the transport level, by combining falsely indicating ECN-Capability with falsely reporting congestion. For an ECN-capable transport, this would cause the transport to unnecessarily react to congestion. In this particular case, the router that is incorrectly changing the ECN field could have dropped the packet. Thus for this case of an ECN-capable transport, the consequence of this change to the ECN field is no worse than dropping the packet.

18.2. Information carried in the Transport Header

For TCP, an ECN-capable TCP receiver informs its TCP peer that it is ECN-capable at the TCP level, conveying this information in the TCP header at the time the connection is setup. This document does not consider potential dangers introduced by changes in the transport header within the network. We note that when IPsec is used, the transport header is protected both in tunnel and transport modes [ESP, AH].

Another issue concerns TCP packets with a spoofed IP source address carrying invalid ECN information in the transport header. For completeness, we examine here some possible ways that a node spoofing the IP source address of another node could use the two ECN flags in the TCP header to launch a denial-of-service attack. However, these attacks would require an ability for the attacker to use valid TCP sequence numbers, and any attacker with this ability and with the ability to spoof IP source addresses could damage the TCP connection without using the ECN flags. Therefore, ECN does not add any new vulnerabilities in this respect.

An acknowledgement packet with a spoofed IP source address of the TCP data receiver could include the ECE bit set. If accepted by the TCP data sender as a valid packet, this spoofed acknowledgement packet could result in the TCP data sender unnecessarily halving its congestion window. However, to be accepted by the data sender, such a spoofed acknowledgement packet would have to have the correct 32-bit sequence number as well as a valid acknowledgement number. An attacker that could successfully send such a spoofed acknowledgement packet could also send a spoofed RST packet, or do other equally damaging operations to the TCP connection.

Packets with a spoofed IP source address of the TCP data sender could include the CWR bit set. Again, to be accepted, such a packet would have to have a valid sequence number. In addition, such a spoofed packet would have a limited performance impact. Spoofing a data packet with the CWR bit set could result in the TCP data receiver sending fewer ECE packets than it would otherwise, if the data receiver was sending ECE packets when it received the spoofed CWR packet.

18.3. Split Paths

In some cases, a malicious or broken router might have access to only a subset of the packets from a flow. The question is as follows: can this router, by altering the ECN field in this subset of the packets, do more damage to that flow than if it had simply dropped that set of packets?

We will classify the packets in the flow as A packets and B packets, and assume that the adversary only has access to A packets. Assume that the adversary is subverting end-to-end congestion control along the path traveled by A packets only, by either falsely indicating ECN-Capability upstream of the point where congestion occurs, or erasing the congestion indication downstream. Consider also that there exists a monitoring device that sees both the A and B packets, and will "punish" both the A and B packets if the total flow is determined not to be properly responding to indications of congestion. Another key characteristic that we believe is likely to be true is that the monitoring device, before 'punishing' the A&B flow, will first drop packets instead of setting the CE codepoint, and will drop arriving packets of that flow that already have the CE codepoint set. If the end nodes are in fact using end-to-end congestion control, they will see all of the indications of congestion seen by the monitoring device, and will begin to respond to these indications of congestion. Thus, the monitoring device is successful in providing the indications to the flow at an early stage.

It is true that the adversary that has access only to the A packets might, by subverting ECN-based congestion control, be able to deny the benefits of ECN to the other packets in the A&B aggregate. While this is unfortunate, this is not a reason to disable ECN.

A variant of falsely reporting congestion occurs when there are two adversaries along a path, where the first adversary falsely reports congestion, and the second adversary 'erases' those reports. (Unlike packet drops, ECN congestion reports can be 'reversed' later in the network by a malicious or broken router. However, the use of the ECN nonce could help the transport to detect this behavior.) While this would be transparent to the end node, it is possible that a monitoring device between the first and second adversaries would see the false indications of congestion. Keep in mind our recommendation in this document, that before 'punishing' a flow for not responding appropriately to congestion, the router will first switch to dropping rather than marking as an indication of congestion, for that flow. When this includes dropping arriving packets from that flow that have the CE codepoint set, this ensures that these indications of congestion are being seen by the end nodes. Thus, there is no additional harm that we are able to postulate as a result of multiple conflicting adversaries.

19. Implications of Subverting End-to-End Congestion Control

This section focuses on the potential repercussions of subverting end-to-end congestion control by either falsely indicating ECN-Capability, or by erasing the congestion indication in ECN (the CE codepoint). Subverting end-to-end congestion control by either of these two methods can have consequences both for the application and for the network. We discuss these separately below.

The first method to subvert end-to-end congestion control, that of falsely indicating ECN-Capability, effectively subverts end-to-end congestion control only if the packet later encounters congestion that results in the setting of the CE codepoint. In this case, the transport protocol (which may not be ECN-capable) does not receive the indication of congestion from these downstream congested routers.

The second method to subvert end-to-end congestion control, 'erasing' the CE codepoint in a packet, effectively subverts end-to-end congestion control only when the CE codepoint in the packet was set earlier by a congested router. In this case, the transport protocol does not receive the indication of congestion from the upstream congested routers.

Either of these two methods of subverting end-to-end congestion control can potentially introduce more damage to the network (and possibly to the flow itself) than if the adversary had simply dropped packets from that flow. However, as we discuss later in this section and in Section 7, this potential damage is limited.

19.1. Implications for the Network and for Competing Flows

The CE codepoint of the ECN field is only used by routers as an indication of congestion during periods of *moderate* congestion. ECN-capable routers should drop rather than mark packets during heavy congestion even if the router's queue is not yet full. For example, for routers using active queue management based on RED, the router should drop rather than mark packets that arrive while the average queue sizes exceed the RED queue's maximum threshold.

One consequence for the network of subverting end-to-end congestion control is that flows that do not receive the congestion indications from the network might increase their sending rate until they drive the network into heavier congestion. Then, the congested router could begin to drop rather than mark arriving packets. For flows that are not isolated by some form of per-flow scheduling or other per-flow mechanisms, but are instead aggregated with other flows in a single queue in an undifferentiated fashion, this packet-dropping at the congested router would apply to all flows that share that queue. Thus, the consequences would be to increase the level of congestion in the network.

In some cases, the increase in the level of congestion will lead to a substantial buffer buildup at the congested queue that will be sufficient to drive the congested queue from the packet-marking to the packet-dropping regime. This transition could occur either because of buffer overflow, or because of the active queue management policy described above that drops packets when the average queue is above RED's maximum threshold. At this point, all flows, including the subverted flow, will begin to see packet drops instead of packet marks, and a malicious or broken router will no longer be able to ' erase' these indications of congestion in the network. If the end nodes are deploying appropriate end-to-end congestion control, then the subverted flow will reduce its arrival rate in response to congestion. When the level of congestion is sufficiently reduced, the congested queue can return from the packet-dropping regime to the packet-marking regime. The steady-state pattern could be one of the congested queue oscillating between these two regimes.

In other cases, the consequences of subverting end-to-end congestion control will not be severe enough to drive the congested link into sufficiently-heavy congestion that packets are dropped instead of being marked. In this case, the implications for competing flows in the network will be a slightly-increased rate of packet marking or dropping, and a corresponding decrease in the bandwidth available to those flows. This can be a stable state if the arrival rate of the subverted flow is sufficiently small, relative to the link bandwidth, that the average queue size at the congested router remains under control. In particular, the subverted flow could have a limited bandwidth demand on the link at this router, while still getting more than its "fair" share of the link. This limited demand could be due to a limited demand from the data source; a limitation from the TCP advertised window; a lower-bandwidth access pipe; or other factors. Thus the subversion of ECN-based congestion control can still lead to unfairness, which we believe is appropriate to note here.

The threat to the network posed by the subversion of ECN-based congestion control in the network is essentially the same as the threat posed by an end-system that intentionally fails to cooperate with end-to-end congestion control. The deployment of mechanisms in routers to address this threat is an open research question, and is discussed further in Section 10.

Let us take the example described in Section 18.1.1, where the CE codepoint that was set in a packet is erased: {'11' -> '10' or '11' -> '01'}. The consequence for the congested upstream router that set the CE codepoint is that this congestion indication does not reach the end nodes for that flow. The source (even one which is completely cooperative and not malicious) is thus allowed to continue to increase its sending rate (if it is a TCP flow, by increasing its congestion window). The flow potentially achieves better throughput than the other flows that also share the congested router, especially if there are no policing mechanisms or per-flow queuing mechanisms at that router. Consider the behavior of the other flows, especially if they are cooperative: that is, the flows that do not experience subverted end-to-end congestion control. They are likely to reduce their load (e.g., by reducing their window size) on the congested router, thus benefiting our subverted flow. This results in unfairness. As we discussed above, this unfairness could either be transient (because the congested queue is driven into the packet-marking regime), oscillatory (because the congested queue oscillates between the packet marking and the packet dropping regime), or more moderate but a persistent stable state (because the congested queue is never driven to the packet dropping regime).

The results would be similar if the subverted flow was intentionally avoiding end-to-end congestion control. One difference is that a flow that is intentionally avoiding end-to-end congestion control at the end nodes can avoid end-to-end congestion control even when the congested queue is in packet-dropping mode, by refusing to reduce its sending rate in response to packet drops in the network. Thus the problems for the network from the subversion of ECN-based congestion control are less severe than the problems caused by the intentional avoidance of end-to-end congestion control in the end nodes. It is also the case that it is considerably more difficult to control the behavior of the end nodes than it is to control the behavior of the infrastructure itself. This is not to say that the problems for the network posed by the network's subversion of ECN-based congestion control are small; just that they are dwarfed by the problems for the network posed by the subversion of either ECN-based or other currently known packet-based congestion control mechanisms by the end nodes.

19.2. Implications for the Subverted Flow

When a source indicates that it is ECN-capable, there is an expectation that the routers in the network that are capable of participating in ECN will use the CE codepoint for indication of congestion. There is the potential benefit of using ECN in reducing the amount of packet loss (in addition to the reduced queuing delays because of active queue management policies). When the packet flows through an IPsec tunnel where the nodes that the tunneled packets traverse are untrusted in some way, the expectation is that IPsec will protect the flow from subversion that results in undesirable consequences.

In many cases, a subverted flow will benefit from the subversion of end-to-end congestion control for that flow in the network, by receiving more bandwidth than it would have otherwise, relative to competing non-subverted flows. If the congested queue reaches the packet-dropping stage, then the subversion of end-to-end congestion control might or might not be of overall benefit to the subverted flow, depending on that flow's relative tradeoffs between throughput, loss, and delay.

One form of subverting end-to-end congestion control is to falsely indicate ECN-capability by setting the ECT codepoint. This has the consequence of downstream congested routers setting the CE codepoint in vain. However, as described in Section 9.1.2, if an ECT codepoint is changed in an IP tunnel, this can be detected at the egress point of the tunnel, as long as the inner header was not changed within the tunnel.

The second form of subverting end-to-end congestion control is to erase the congestion indication by erasing the CE codepoint. In this case, it is the upstream congested routers that set the CE codepoint in vain.

If an ECT codepoint is erased within an IP tunnel, then this can be detected at the egress point of the tunnel, as long as the inner header was not changed within the tunnel. If the CE codepoint is set upstream of the IP tunnel, then any erasure of the outer header's CE codepoint within the tunnel will have no effect because the inner header preserves the set value of the CE codepoint. However, if the CE codepoint is set within the tunnel, and erased either within or downstream of the tunnel, this is not necessarily detected at the egress point of the tunnel.

With this subversion of end-to-end congestion control, an end-system transport does not respond to the congestion indication. Along with the increased unfairness for the non-subverted flows described in the previous section, the congested router's queue could continue to build, resulting in packet loss at the congested router - which is a means for indicating congestion to the transport in any case. In the interim, the flow might experience higher queuing delays, possibly along with an increased bandwidth relative to other non-subverted flows. But transports do not inherently make assumptions of consistently experiencing carefully managed queuing in the path. We believe that these forms of subverting end-to-end congestion control are no worse for the subverted flow than if the adversary had simply dropped the packets of that flow itself.

19.3. Non-ECN-Based Methods of Subverting End-to-end Congestion Control

We have shown that, in many cases, a malicious or broken router that is able to change the bits in the ECN field can do no more damage than if it had simply dropped the packet in question. However, this is not true in all cases, in particular in the cases where the broken router subverted end-to-end congestion control by either falsely indicating ECN-Capability or by erasing the ECN congestion indication (in the CE codepoint). While there are many ways that a router can harm a flow by dropping packets, a router cannot subvert end-to-end congestion control by dropping packets. As an example, a router cannot subvert TCP congestion control by dropping data packets, acknowledgement packets, or control packets.

Even though packet-dropping cannot be used to subvert end-to-end congestion control, there *are* non-ECN-based methods for subverting end-to-end congestion control that a broken or malicious router could use. For example, a broken router could duplicate data packets, thus effectively negating the effects of end-to-end congestion control along some portion of the path. (For a router that duplicated packets within an IPsec tunnel, the security administrator can cause the duplicate packets to be discarded by configuring anti-replay protection for the tunnel.) This duplication of packets within the network would have similar implications for the network and for the subverted flow as those described in Sections 18.1.1 and 18.1.4 above.

20. The Motivation for the ECT Codepoints.

20.1. The Motivation for an ECT Codepoint.

The need for an ECT codepoint 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 an ECT codepoint, the router can drop packets from flows that are not ECN-capable, but can *instead* set the CE codepoint in packets that *are*

ECN-capable. Because an ECT codepoint allows an end node to have the CE codepoint 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 codepoint, then the router would have to set the CE codepoint 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 codepoints, 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 codepoint 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 codepoints is functionally equivalent to a flow that turns off congestion control, as discussed earlier in this document.

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.

20.2. The Motivation for two ECT Codepoints.

The primary motivation for the two ECT codepoints is to provide a one-bit ECN nonce. The ECN nonce allows the development of mechanisms for the sender to probabilistically verify that network elements are not erasing the CE codepoint, and that data receivers are properly reporting to the sender the receipt of packets with the CE codepoint set.

Another possibility for senders to detect misbehaving network elements or receivers would be for the data sender to occasionally send a data packet with the CE codepoint set, to see if the receiver reports receiving the CE codepoint. Of course, if these packets encountered congestion in the network, the router might make no change in the packets, because the CE codepoint would already be set. Thus, for packets sent with the CE codepoint set, the TCP end-nodes could not determine if some router intended to set the CE codepoint in these packets. For this reason, sending packets with the CE codepoint would have to be done sparingly, and would be a less effective check against misbehaving network elements and receivers than would be the ECN nonce.

The assignment of the fourth ECN codepoint to ECT(1) precludes the use of this codepoint for some other purposes. For clarity, we briefly list other possible purposes here.

One possibility might have been for the data sender to use the fourth ECN codepoint to indicate an alternate semantics for ECN. However, this seems to us more appropriate to be signaled using a differentiated services codepoint in the DS field.

A second possible use for the fourth ECN codepoint would have been to give the router two separate codepoints for the indication of congestion, CE(0) and CE(1), for mild and severe congestion respectively. While this could be useful in some cases, this certainly does not seem a compelling requirement at this point. If there was judged to be a compelling need for this, the complications of incremental deployment would most likely necessitate more that just one codepoint for this function.

A third use that has been informally proposed for the ECN codepoint is for use in some forms of multicast congestion control, based on randomized procedures for duplicating marked packets at routers. Some proposed multicast packet duplication procedures are based on a new ECN codepoint that (1) conveys the fact that congestion occurred upstream of the duplication point that marked the packet with this codepoint and (2) can detect congestion downstream of that duplication point. ECT(1) can serve this purpose because it is both distinct from ECT(0) and is replaced by CE when ECN marking occurs in response to congestion or incipient congestion. Explanation of how this enhanced version of ECN would be used by multicast congestion control is beyond the scope of this document, as are ECN-aware multicast packet duplication procedures and the processing of the ECN field at multicast receivers in all cases (i.e., irrespective of the multicast packet duplication procedure(s) used).

The specification of IP tunnel modifications for ECN in this document assumes that the only change made to the outer IP header's ECN field between tunnel endpoints is to set the CE codepoint to indicate congestion. This is not consistent with some of the proposed uses of ECT(1) by the multicast duplication procedures in the previous paragraph, and such procedures SHOULD NOT be deployed unless this inconsistency between multicast duplication procedures and IP tunnels with full ECN functionality is resolved. Limited ECN functionality may be used instead, although in practice many tunnel protocols (including IPsec) will not work correctly if multicast traffic duplication occurs within the tunnel

21. 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) codepoints should have been overloaded on a single bit. This overloaded-one-bit alternative, explored in [Floyd94], would have involved 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 codepoint 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 concern that was described earlier (and recommended in this document) is that transports (particularly TCP) should not mark pure ACK packets or retransmitted packets as being ECN-Capable. A pure ACK packet from a non-ECN-capable transport could be dropped, without necessarily having an impact on the transport from a congestion control perspective (because subsequent ACKs are cumulative). An ECN-capable transport reacting to the CE codepoint in a pure ACK packet by reducing the window would be at a disadvantage in comparison to a non-ECN-capable transport. For this reason (and for reasons described earlier in relation to retransmitted packets), it is desirable to have the ECT codepoint set on a per-packet basis.

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 not-ECT codepoint should be the 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.

22. 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 and RFC 1455 have been obsoleted by "Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers" [RFC2474] in which bits 6 and 7 of the DS field are listed as Currently Unused (CU). RFC 2780 [RFC2780] specified ECN as an experimental use of the two-bit CU field. RFC 2780 updated the definition of the DS Field to only encompass the first six bits of this octet rather than all eight bits; these first six bits are defined as the Differentiated Services CodePoint (DSCP):

```
    0     1     2     3     4     5     6     7
 +-----+-----+-----+-----+-----+-----+-----+-----+
 |               DSCP                |    CU     |  RFCs 2474,
 +-----+-----+-----+-----+-----+-----+-----+-----+    2780
```

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.

Prior to RFC 2474, routers were not permitted to modify bits in either the DSCP or ECN field of packets forwarded through them, and hence routers that comply only with RFCs prior to 2474 should have no effect on ECN. For end nodes, bit 7 (the second ECN bit) must be transmitted as zero for any implementation compliant only with RFCs prior to 2474. Such nodes may transmit bit 6 (the first ECN bit) as one for the "Minimize Monetary Cost" provision of RFC 1349 or the experiment authorized by RFC 1455; neither this aspect of RFC 1349 nor the experiment in RFC 1455 were widely implemented or used. The damage that could be done by a broken, non-conformant router would include "erasing" the CE codepoint for an ECN-capable packet that arrived at the router with the CE codepoint set, or setting the CE codepoint even in the absence of congestion. This has been discussed in the section 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 codepoint set has been discussed in the section on "Non-compliance by the End Nodes".

23. IANA Considerations

This section contains the namespaces that have either been created in this specification, or the values assigned in existing namespaces managed by IANA.

23.1. IPv4 TOS Byte and IPv6 Traffic Class Octet

The codepoints for the ECN Field of the IP header are specified by the Standards Action of this RFC, as is required by RFC 2780.

When this document is published as an RFC, IANA should create a new registry, "IPv4 TOS Byte and IPv6 Traffic Class Octet", with the namespace as follows:

IPv4 TOS Byte and IPv6 Traffic Class Octet

Description: The registrations are identical for IPv4 and IPv6.

Bits 0-5: see Differentiated Services Field Codepoints Registry
(http://www.iana.org/assignments/dscp-registry)

Bits 6-7, ECN Field:

Binary	Keyword	References
00	Not-ECT (Not ECN-Capable Transport)	[RFC 3168]
01	ECT(1) (ECN-Capable Transport(1))	[RFC 3168]
10	ECT(0) (ECN-Capable Transport(0))	[RFC 3168]
11	CE (Congestion Experienced)	[RFC 3168]

23.2. TCP Header Flags

The codepoints for the CWR and ECE flags in the TCP header are specified by the Standards Action of this RFC, as is required by RFC 2780.

When this document is published as an RFC, IANA should create a new registry, "TCP Header Flags", with the namespace as follows:

TCP Header Flags

The Transmission Control Protocol (TCP) included a 6-bit Reserved field defined in RFC 793, reserved for future use, in bytes 13 and 14 of the TCP header, as illustrated below. The other six Control bits are defined separately by RFC 793.

```
  0   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|               |                       | U | A | P | R | S | F |
| Header Length |        Reserved       | R | C | S | S | Y | I |
|               |                       | G | K | H | T | N | N |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
```

RFC 3168 defines two of the six bits from the Reserved field to be used for ECN, as follows:

```
  0   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|               |               | C | E | U | A | P | R | S | F |
| Header Length |    Reserved   | W | C | R | C | S | S | Y | I |
|               |               | R | E | G | K | H | T | N | N |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
```

TCP Header Flags

Bit	Name	Reference
8	CWR (Congestion Window Reduced)	[RFC 3168]
9	ECE (ECN-Echo)	[RFC 3168]

23.3. IPSEC Security Association Attributes

IANA allocated the IPSEC Security Association Attribute value 10 for the ECN Tunnel use described in Section 9.2.1.2 above at the request of David Black in November 1999. The IANA has changed the Reference for this allocation from David Black's request to this RFC.

24. Authors' Addresses

K. K. Ramakrishnan
TeraOptic Networks, Inc.

Phone: +1 (408) 666-8650
EMail: kk@teraoptic.com

Sally Floyd
ACIRI

Phone: +1 (510) 666-2989
EMail: floyd@aciri.org
URL: http://www.aciri.org/floyd/

David L. Black
EMC Corporation
42 South St.
Hopkinton, MA 01748

Phone: +1 (508) 435-1000 x75140
EMail: black_david@emc.com

25. Full Copyright Statement

Acknowledgement

Funding for the RFC Editor function is currently provided by the Internet Society.