Internet Engineering Task Force
INTERNET-DRAFT
Intended status: Proposed Standard
Expires: 18 May 2008

A. Kuzmanovic
A. Mondal
Northwestern University
S. Floyd
ICIR
K.K. Ramakrishnan
AT&T

18 November 2007

Adding Explicit Congestion Notification (ECN) Capability to TCP's SYN/ACK Packets

draft-ietf-tcpm-ecnsyn-03.txt

Status of this Memo

Copyright Notice

Abstract

This draft specifies a modification to RFC 3168 to allow TCP SYN/ACK packets to be ECN-Capable. For TCP, RFC 3168 only specifies setting an ECN-Capable codepoint on data packets, and not on SYN and SYN/ACK packets. However, because of the high cost to the TCP transfer of having a SYN/ACK packet dropped, with the resulting retransmit timeout, this document specifies the use of ECN for the SYN/ACK packet itself, when sent in response to a SYN packet with the two ECN flags set in the TCP header, indicating a willingness to use ECN. Setting TCP SYN/ACK packets as ECN-Capable can be of great benefit to the TCP connection, avoiding the severe penalty of a retransmit timeout for a connection that has not yet started placing a load on the network. The sender of the SYN/ACK packet must respond to a report of an ECN-marked SYN/ACK packet by reducing its initial congestion window from two, three, or four segments to one segment, thereby reducing the subsequent load from that connection on the network.

Table of Contents

NOTE TO RFC EDITOR: PLEASE DELETE THIS NOTE UPON PUBLICATION.

Changes from draft-ietf-tcpm-ecnsyn-02:

* Added to the discussion in the Security section of whether

ECN-Capable TCP SYN packets have problems with firewalls, over and above the known problems of TCP data packets (e.g., as in the Microsoft report). From a question raised at the TCPM meeting at the July 2007 IETF.

* Added a sentence to the discussion of routers or middleboxes that *might* drop TCP SYN packets on the basis of IP header fields. Feedback from Remi Denis-Courmont.

* General editing. Feedback from Alfred Henes.

Changes from draft-ietf-tcpm-ecnsyn-01:

* Changes in response to feedback from Anil Agarwal.

* Added a look at the costs of adding ECN-Capability to SYN/ACKs in a highly-congested scenario. From feedback from Mark Allman and Janardhan Iyengar.

* Added a comparative evaluation of two possible responses to an ECN-marked SYN/ACK packet. From Mark Allman.

Changes from draft-ietf-tcpm-ecnsyn-00:

* Only updating the revision number.

Changes from draft-ietf-twvsg-ecnsyn-00:

* Changed name of draft to draft-ietf-tcpm-ecnsyn.

* Added a discussion in Section 3 of "Response to ECN-marking of SYN/ACK packets". Based on suggestions from Mark Allman.

* Added a discussion to the Conclusions about adding ECN-capability to relevant set-up packets in other protocols. From a suggestion from Wesley Eddy.

* Added a description of SYN exchanges with SYN cookies. From a suggestion from Wesley Eddy.

* Added a discussion of one-way data transfers, where the host sending the SYN/ACK packet sends no data packets.

* Minor editing, from feedback from Mark Allman and Janardhan Iyengar.

* Future work: a look at the costs of adding

ECN-Capability in a worst-case scenario.
From feedback from Mark Allman and Janardhan Iyengar.

* Future work: a comparative evaluation of two possible responses to an ECN-marked SYN/ACK packet.

Changes from draft-kuzmanovic-ecn-syn-00.txt:

* Changed name of draft to draft-ietf-twvsg-ecnsyn.

END OF NOTE TO RFC EDITOR.

1. Introduction

TCP's congestion control mechanism has primarily used packet loss as the congestion indication, with packets dropped when buffers overflow. With such tail-drop mechanisms, the packet delay can be high, as the queue at bottleneck routers can be fairly large. Dropping packets only when the queue overflows, and having TCP react only to such losses, results in:
1) significantly higher packet delay;
2) unnecessarily many packet losses; and
3) unfairness due to synchronization effects.

The adoption of Active Queue Management (AQM) mechanisms allows better control of bottleneck queues [RFC2309]. This use of AQM has the following potential benefits:
1) better control of the queue, with reduced queueing delay;
2) fewer packet drops; and
3) better fairness because of fewer synchronization effects.

With the adoption of ECN, performance may be further improved. When the router detects congestion before buffer overflow, the router can provide a congestion indication either by dropping a packet, or by setting the Congestion Experienced (CE) codepoint in the Explicit Congestion Notification (ECN) field in the IP header [RFC3168]. The IETF has standardized the use of the Congestion Experienced (CE) codepoint in the IP header for routers to indicate congestion. For incremental deployment and backwards compatibility, the RFC on ECN [RFC3168] specifies that routers may mark ECN-capable packets that would otherwise have been dropped, using the Congestion Experienced codepoint in the ECN field. The use of ECN allows TCP to react to congestion while avoiding unnecessary retransmissions and, in some cases, unnecessary retransmit timeouts. Thus, using ECN has several benefits:

1) For short transfers, a TCP connection's congestion window may be small. For example, if the current window contains only one packet,

and that packet is dropped, TCP will have to wait for a retransmit timeout to recover, reducing its overall throughput. Similarly, if the current window contains only a few packets and one of those packets is dropped, there might not be enough duplicate acknowledgements for a fast retransmission, and the sender might have to wait for a delay of several round-trip times using Limited Transmit [RFC3042]. With the use of ECN, short flows are less likely to have packets dropped, sometimes avoiding unnecessary delays or costly retransit timeouts.

2) While longer flows may not see substantially improved throughput with the use of ECN, they experience lower loss. This may benefit TCP applications that are latency- and loss-sensitive, because of the avoidance of retransmissions.

RFC 3168 only specifies marking the Congestion Experienced codepoint on TCP's data packets, and not on SYN and SYN/ACK packets. RFC 3168 specifies the negotiation of the use of ECN between the two TCP end-points in the TCP SYN and SYN-ACK exchange, using flags in the TCP header. Erring on the side of being conservative, RFC 3168 does not specify the use of ECN for the SYN/ACK packet itself. However, because of the high cost to the TCP transfer of having a SYN/ACK packet dropped, with the resulting retransmit timeout, this document specifies the use of ECN for the SYN/ACK packet itself. This can be of great benefit to the TCP connection, avoiding the severe penalty of a retransmit timeout for a connection that has not yet started placing a load on the network. The sender of the SYN/ACK packet must respond to a report of an ECN-marked SYN/ACK packet by reducing its initial congestion window from two, three, or four segments to one segment, reducing the subsequent load from that connection on the network.

The use of ECN for SYN/ACK packets has the following potential benefits:
1) Avoidance of a retransmit timeout;
2) Improvement in the throughput of short connections.

This draft specifies ECN+, a modification to RFC 3168 to allow TCP SYN/ACK packets to be ECN-Capable. Section 3 contains the specification of the change, while Section 4 discusses some of the issues, and Section 5 discusses related work. Section 6 contains an evaluation of the proposed change.

2. Conventions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC 2119].

3. Proposal

This section specifies the modification to RFC 3168 to allow TCP SYN/ACK packets to be ECN-Capable. We use the following terminology from RFC 3168:

The ECN field in the IP header:
- o CE: the Congestion Experienced codepoint; and
- o ECT: either one of the two ECN-Capable Transport codepoints.

The ECN flags in the TCP header:
- o CWR: the Congestion Window Reduced flag; and
- o ECE: the ECN-Echo flag.

ECN-setup packets:
- o ECN-setup SYN packet: a SYN packet with the ECE and CWR flags;
- o ECN-setup SYN-ACK packet: a SYN-ACK packet with ECE but not CWR.

RFC 3168 in Section 6.1.1. states that "A host MUST NOT set ECT on SYN or SYN-ACK packets." In this section, we specify that a TCP node MAY respond to an ECN-setup SYN packet by setting ECT in the responding ECN-setup SYN/ACK packet, indicating to routers that the SYN/ACK packet is ECN-Capable. This allows a congested router along the path to mark the packet instead of dropping the packet as an indication of congestion.

Assume that TCP node A transmits to TCP node B an ECN-setup SYN packet, indicating willingness to use ECN for this connection. As specified by RFC 3168, if TCP node B is willing to use ECN, node B responds with an ECN-setup SYN-ACK packet.

Figure 1 shows an interchange with the SYN/ACK packet dropped by a congested router. Node B waits for a retransmit timeout, and then retransmits the SYN/ACK packet.

```
---------------------------------------------------------------
   TCP Node A             Router                   TCP Node B
   ----------             ------                   ----------

   ECN-setup SYN packet --->
                                    ECN-setup SYN packet --->

                          <--- ECN-setup SYN/ACK, possibly ECT
                                           3-second timer set
                        SYN/ACK dropped              .
                                                     .
                                                     .
                                        3-second timer expires
                               <--- ECN-setup SYN/ACK, not ECT
   <--- ECN-setup SYN/ACK
   Data/ACK --->
                                                 Data/ACK --->
                              <--- Data (one to four segments)
---------------------------------------------------------------
```

Figure 1: SYN exchange with the SYN/ACK packet dropped.

If the SYN/ACK packet is dropped in the network, the TCP host (node B) responds by waiting three seconds for the retransmit timer to expire [RFC2988]. If a SYN/ACK packet with the ECT codepoint is dropped, the TCP node SHOULD resend the SYN/ACK packet without the ECN-Capable codepoint. (Although we are not aware of any middleboxes that drop SYN/ACK packets that contain an ECN-Capable codepoint in the IP header, we have learned to design our protocols defensively in this regard [RFC3360].)

We note that if syn-cookies were used by Node B in the exchange in Figure 1, TCP Node B wouldn’t set a timer upon transmission of the SYN/ACK packet [SYN-COOK]. In this case, if the SYN/ACK packet was lost, the initiator (Node A) would have to timeout and retransmit the SYN packet in order to trigger another SYN-ACK.

Figure 2 shows an interchange with the SYN/ACK packet sent as ECN-Capable, and ECN-marked instead of dropped at the congested router.

```
 ---------------------------------------------------------------
    TCP Node A             Router                  TCP Node B
    ----------             ------                  ----------

    ECN-setup SYN packet --->
                                ECN-setup SYN packet --->

                              <--- ECN-setup SYN/ACK, ECT
                   <--- Sets CE on SYN/ACK
    <--- ECN-setup SYN/ACK, CE

    Data/ACK, ECN-Echo --->
                                  Data/ACK, ECN-Echo --->
                             Window reduced to one segment.
                        <--- Data, CWR (one segment only)
 ---------------------------------------------------------------
```

Figure 2: SYN exchange with the SYN/ACK packet marked.

If the receiving node (node A) receives a SYN/ACK packet that has been marked by the congested router, with the CE codepoint set, the receiving node MUST respond by setting the ECN-Echo flag in the TCP header of the responding ACK packet. As specified in RFC 3168, the receiving node continues to set the ECN-Echo flag in packets until it receives a packet with the CWR flag set.

When the sending node (node B) receives the ECN-Echo packet reporting the Congestion Experienced indication in the SYN/ACK packet, the node MUST set the initial congestion window to one segment, instead of two segments as allowed by [RFC2581], or three or four segments allowed by [RFC3390]. If the sending node (node B) was going to use an initial window of one segment, and receives an ECN-Echo packet informing it of a Congestion Experienced indication on its SYN/ACK packet, the sending node MAY continue to send with an initial window of one segment, without waiting for a retransmit timeout. We note that this updates RFC 3168, which specifies that "the sending TCP MUST reset the retransmit timer on receiving the ECN-Echo packet when the congestion window is one." As specified by RFC 3168, the sending node (node B) also sets the CWR flag in the TCP header of the next data packet sent, to acknowledge its receipt of and reaction to the ECN-Echo flag.

If the data transfer in Figure 2 is entirely from Node A to Node B, then data packets from Node A continue to set the ECN-Echo flag in data packets, waiting for the CWR flag from Node B acknowledging a response to the ECN-Echo flag.

4. Discussion

Motivation:
The rationale for the proposed change is the following. When node B receives a TCP SYN packet with ECN-Echo bit set in the TCP header, this indicates that node A is ECN-capable. If node B is also ECN-capable, there are no obstacles to immediately setting one of the ECN-Capable codepoints in the IP header in the responding TCP SYN/ACK packet.

There can be a great benefit in setting an ECN-capable codepoint in SYN/ACK packets, as is discussed further in [ECN+], and reported briefly in Section 5 below. Congestion is most likely to occur in the server-to-client direction. As a result, setting an ECN-capable codepoint in SYN/ACK packets can reduce the occurrence of three-second retransmit timeouts resulting from the drop of SYN/ACK packets.

Flooding attacks:
Setting an ECN-Capable codepoint in the responding TCP SYN/ACK packets does not raise any novel security vulnerabilities. For example, provoking servers or hosts to send SYN/ACK packets to a third party in order to perform a "SYN/ACK flood" attack would be highly inefficient. Third parties would immediately drop such packets, since they would know that they didn't generate the TCP SYN packets in the first place. Moreover, such SYN/ACK attacks would have the same signatures as the existing TCP SYN attacks. Provoking servers or hosts to reply with SYN/ACK packets in order to congest a certain link would also be highly inefficient because SYN/ACK packets are small in size.

However, the addition of ECN-Capability to SYN/ACK packets could allow SYN/ACK packets to persist for more hops along a network path before being dropped, thus adding somewhat to the ability of a SYN/ACK attack to flood a network link.

The TCP SYN packet:
There are several reasons why an ECN-Capable codepoint MUST NOT be set in the IP header of the initiating TCP SYN packet. First, when the TCP SYN packet is sent, there are no guarantees that the other TCP endpoint (node B in Figure 2) is ECN-capable, or that it would be able to understand and react if the ECN CE codepoint was set by a congested router.

Second, the ECN-Capable codepoint in TCP SYN packets could be misused by malicious clients to 'improve' the well-known TCP SYN attack. By setting an ECN-Capable codepoint in TCP SYN packets, a malicious host might be able to inject a large number of TCP SYN packets through a potentially congested ECN-enabled router, congesting it even further.

For both these reasons, we continue the restriction that the TCP SYN packet MUST NOT have the ECN-Capable codepoint in the IP header set.

Backwards compatibility:
In order for TCP node B to send a SYN/ACK packet as ECN-Capable, node B must have received an ECN-setup SYN packet from node A. However, it is possible that node A supports ECN, but either ignores the CE codepoint on received SYN/ACK packets, or ignores SYN/ACK packets with the ECT or CE codepoint set. If the TCP sender ignores the CE codepoint on received SYN/ACK packets, this would mean that the TCP connection would not respond to this congestion indication. However, this seems to us an acceptable cost to pay in the incremental deployment of ECN-Capability for TCP's SYN/ACK packets. It would mean that the sender of the SYN/ACK packet would not reduce the initial congestion window from two, three, or four segments down to one segment, as it should. However, the TCP sender would still respond correctly to any subsequent CE indications on data packets later on in the connection. Thus, to be explicit, when a TCP connection includes a sender that supports ECN but *does not* support ECN-Capability for SYN/ACK packets, in combination with a receiver that *does* support ECN-Capabililty for SYN/ACK packets, it is quite possible that the ECN-Capable SYN/ACK packets will be marked rather than dropped in the network, and that the sender will not respond to the ECN mark on the SYN/ACK packet.

It is also possible that in some older TCP implementation, the TCP sender would ignore arriving SYN/ACK packets that had the ECT or CE codepoint set. This would result in a delay in connection set-up for that TCP connection, with the TCP sender re-sending the SYN packet after a retransmit timeout. We are not aware of any TCP implementations with this behavior.

SYN/ACK packets and packet size:
There are a number of router buffer architectures that have smaller dropping rates for small (SYN) packets than for large (data) packets. For example, for a Drop Tail queue in units of packets, where each packet takes a single slot in the buffer regardless of packet size, small and large packets are equally likely to be dropped. However, for a Drop Tail queue in units of bytes, small packets are less likely to be dropped than are large ones. Similarly, for RED in packet mode, small and large packets are equally likely to be dropped or marked, while for RED in byte mode, a packet's chance of being dropped or marked is proportional to the packet size in bytes.

For a congested router with an AQM mechanism in byte mode, where a packet's chance of being dropped or marked is proportional to the packet size in bytes, the drop or marking rate for TCP SYN/ACK packets should generally be low. In this case, the benefit of making SYN/ACK packets ECN-Capable should be similarly moderate. However, for a congested router with a Drop Tail queue in units of packets or with an AQM mechanism in packet mode, and with no priority queueing for smaller packets, small and large packets should have the same probability of being dropped or marked. In such a case, making SYN/ACK packets ECN-Capable should be of significant benefit.

We believe that there are a wide range of behaviors in the real world in terms of the drop or mark behavior at routers as a function of packet size [Tools] (Section 10). We note that all of these alternatives listed above are available in the NS simulator (Drop Tail queues are by default in units of packets, while the default for RED queue management has been changed from packet mode to byte mode).

Response to ECN-marking of SYN/ACK packets:
One question is why TCP SYN/ACK packets should be treated differently from other packets in terms of the packet sender's response to an ECN-marked packet. Section 5 of RFC 3168 specifies the following:

"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."

In particular, Section 6.1.2 of RFC 3168 specifies that when the TCP congestion window consists of a single packet and that packet is ECN-marked in the network, then the sender must reduce the sending rate below one packet per round-trip time, by waiting for one RTO before sending another packet. If the RTO was set to the average round-trip time, this would result in halving the sending rate; because the RTO is in fact larger than the average round-trip time, the sending rate is reduced to less than half of its previous value.

TCP's congestion control response to the *dropping* of a SYN/ACK packet is to wait a default time before sending another packet. This document argues that ECN gives end-systems a wider range of possible responses to the *marking* of a SYN/ACK packet, and that waiting a default time before sending a data packet is not the desired response.

On the conservative end, one could assume an effective congestion window of one packet for the SYN/ACK packet, and respond to an ECN-marked SYN/ACK packet by reducing the sending rate to one packet every two round-trip times. As an approximation, the TCP end-node could measure the round-trip time T between the sending of the SYN/ACK packet and the receipt of the acknowledgement, and reply to the acknowledgement of the ECN-marked SYN/ACK packet by waiting T seconds before sending a data packet.

However, we note that for an ECN-marked SYN/ACK packet, halving the *congestion window* is not the same as halving the *sending rate*; there is no 'sending rate' associated with an ECN-Capable SYN/ACK packet, as such packets are only sent as the first packet in a connection from that host. Further, a router's marking of a SYN/ACK packet is not affected by any past history of that connection.

Adding ECN-Capability to SYN/ACK packets allows the simple response of setting the initial congestion window to one packet, instead of its allowed default value of two, three, or four packets, with the host proceeding with a cautious sending rate of one packet per round-trip time. If that packet is ECN-marked or dropped, then the sender will wait an RTO before sending another packet. This document argues that this approach is useful to users, with no dangers of congestion collapse or of starvation of competing traffic. This is discussed in more detail below in Section 6.2.

We note that if the data transfer is entirely from Node A to Node B, then there is no effective difference between the two possible responses to an ECN-marked SYN/ACK packet outlined above. In either case, Node B sends no data packets, only sending acknowledgement packets in response to received data packets.

5. Related Work

The addition of ECN-capability to TCP's SYN/ACK packets was proposed in [ECN+]. The paper includes an extensive set of simulation and testbed experiments to evaluate the effects of the proposal, using several Active Queue Management (AQM) mechanisms, including Random Early Detection (RED) [RED], Random Exponential Marking (REM) [REM], and Proportional Integrator (PI) [PI]. The performance measures were the end-to-end response times for each request/response pair, and the aggregate throughput on the bottleneck link. The end-to-end response time was computed as the time from the moment when the request for the file is sent to the server, until that file is successfully downloaded by the client.

The measurements from [ECN+] show that setting an ECN-Capable codepoint in the IP packet header in TCP SYN/ACK packets systematically improves performance with all evaluated AQM schemes. When SYN/ACK packets at a congested router are ECN-marked instead of dropped, this can avoid a long initial retransmit timeout, improving the response time for the affected flow dramatically.

[ECN+] shows that the impact on aggregate throughput can also be quite significant, because marking SYN ACK packets can prevent larger flows from suffering long timeouts before being "admitted" into the network. In addition, the testbed measurements from [ECN+] show that web servers setting the ECN-Capable codepoint in TCP SYN/ACK packets could serve more requests.

As a final step, [ECN+] explores the co-existence of flows that do and don't set the ECN-capable codepoint in TCP SYN/ACK packets. The results in [ECN+] show that both types of flows can coexist, with some performance degradation for flows that don't use ECN+. Flows that do use ECN+ improve their end-to-end performance. At the same time, the performance degradation for flows that don't use ECN+, as a result of the flows that do use ECN+, increases as a greater fraction of flows use ECN+.

6. Performance Evaluation

6.1. The Costs and Benefit of Adding ECN-Capability

[ECN+] explores the costs and benefits of adding ECN-Capability to SYN/ACK packets with both simulations and experiments. The addition of ECN-capability to SYN/ACK packets could be of significant benefit for those ECN connections that would have had the SYN/ACK packet dropped in the network, and for which the ECN-Capability would allow the SYN/ACK to be marked rather than dropped.

The percent of SYN/ACK packets on a link can be quite high. In particular, measurements on links dominated by web traffic indicate that 15-20% of the packets can be SYN/ACK packets [SCJO01].

The benefit of adding ECN-capability to SYN/ACK packets depends in part on the size of the data transfer. The drop of a SYN/ACK packet can increase the download time of a short file by an order of magnitude, by requiring a three-second retransmit timeout. For longer-lived flows, the effect of a dropped SYN/ACK packet on file download time is less dramatic. However, even for longer-lived flows, the addition of ECN-capability to SYN/ACK packets can improve the fairness among long-lived flows, as newly-arriving flows would be less likely to have to wait for retransmit timeouts.

One question that arises is what fraction of connections would see the benefit from making SYN/ACK packets ECN-capable, in a particular scenario. Specifically:

(1) What fraction of arriving SYN/ACK packets are dropped at the congested router when the SYN/ACK packets are not ECN-capable?

(2) Of those SYN/ACK packets that are dropped, what fraction would have been ECN-marked instead of dropped if the SYN/ACK packets had been ECN-capable?

To answer (1), it is necessary to consider not only the level of congestion but also the queue architecture at the congested link. As described in Section 4 above, for some queue architectures small packets are less likely to be dropped than large ones. In such an environment, SYN/ACK packets would have lower packet drop rates; question (1) could not necessarily be inferred from the overall packet drop rate, but could be answered by measuring the drop rate for SYN/ACK packets directly. In such an environment, adding ECN-capability to SYN/ACK packets would be of less dramatic benefit than in environments where all packets are equally likely to be dropped regardless of packet size.

As question (2) implies, even if all of the SYN/ACK packets were ECN-capable, there could still be some SYN/ACK packets dropped instead of marked at the congested link; the full answer to question (2) depends on the details of the queue management mechanism at the router. If congestion is sufficiently bad, and the queue management mechanism cannot prevent the buffer from overflowing, then SYN/ACK packets will be dropped rather than marked upon buffer overflow whether or not they are ECN-capable.

For some AQM mechanisms, ECN-capable packets are marked instead of dropped any time this is possible, that is, any time the buffer is not yet full. For other AQM mechanisms however, such as the RED mechanism as recommended in [RED], packets are dropped rather than marked when the packet drop/mark rate exceeds a certain threshold, e.g., 10%, even if the packets are ECN-capable. For a router with such an AQM mechanism, when congestion is sufficiently severe to cause a high drop/mark rate, some SYN/ACK packets would be dropped instead of marked whether or not they were ECN-capable.

Thus, the degree of benefit of adding ECN-Capability to SYN/ACK packets depends not only on the overall packet drop rate in the network, but also on the queue management architecture at the congested link.

6.2. An Evaluation of Different Responses to ECN-Marked SYN/ACK Packets

This document specifies that the end-node responds to the report of an ECN-marked SYN/ACK packet by setting the initial congestion window to one segment, instead of its possible default value of two to four segments. We call this ECN+ with NoWaiting. However, in Section 4 discussed another possible response to an ECN-marked SYN/ACK packet, of the end-node waiting an RTT before sending a data packet. We call this approach ECN+ with Waiting.

Simulations comparing the performance with Standard ECN (without ECN-marked SYN/ACK packets), ECN+ with NoWaiting, and ECN+ with Waiting show little difference, in terms of aggregate congestion, between ECN+ with NoWaiting and ECN+ with Waiting. The details are given in Appendix A below. Our conclusions are that ECN+ with NoWaiting is perfectly safe, and there are no congestion-related reasons for preferring ECN+ with Waiting over ECN+ with NoWaiting. That is, there is no need for the TCP end-node to wait a round-trip time before sending a data packet after receiving an acknowledgement of an ECN-marked SYN/ACK packet.

7. Security Considerations

TCP packets carrying the ECT codepoint in IP headers can be marked rather than dropped by ECN-capable routers. This raises several security concerns that we discuss below.

"Bad" routers or middleboxes:
There are a number of known deployment problems from using ECN with TCP traffic in the Internet. The first reported problem, dating back to 2000, is of a small but decreasing number of routers or middleboxes that reset a TCP connection in response to TCP SYN packets using flags in the TCP header to negotiate ECN-capability [Kelson00] [RFC3360] [MAF05]. Dave Thaler reported at the March 2007 IETF of new two problems encountered by TCP connections using ECN; the first of the two problems concerns routers that crash when a TCP data packet arrives with the ECN field in the IP header with the codepoint ECT(0) or ECT(1), indicating that an ECN-Capable connection has been established [SBT07].

While there is no evidence that any routers or middleboxes drop SYN/ACK packets that contain an ECN-Capable or CE codepoint in the IP header, such behavior cannot be excluded. (There seems to be a number of routers or middleboxes that drop TCP SYN packets that contain known or unknown IP options [MAF05] (Figure 1).) Thus, as specified in Section 3, if a SYN/ACK packet with the ECT or CE codepoint is dropped, the TCP node SHOULD resend the SYN/ACK packet without the ECN-Capable codepoint. There is also no evidence that any routers or middleboxes crash when a SYN/ACK arrives with an ECN-Capable or CE codepoint in the IP header (over and above the routers already known to crash when a data packet arrives with either ECT(0) or ECT(1)), but we have not conducted any measurement studies of this [F07].

Congestion collapse:
Because TCP SYN/ACK packets carrying an ECT codepoint could be ECN-marked instead of dropped at an ECN-capable router, the concern is whether this can either invoke congestion, or worsen performance in highly congested scenarios. However, after learning that a SYN/ACK packet was ECN-marked, the sender of that packet will only send one data packet; if this data packet is ECN-marked, the sender will then wait for a retransmission timeout. In addition, routers are free to drop rather than mark arriving packets in times of high congestion, regardless of whether the packets are ECN-capable. When congestion is very high and a router's buffer is full, the router has no choice but to drop rather than to mark an arriving packet.

The simulations reported in Appendix A show that even with demanding traffic mixes dominated by short flows and high levels of congestion, the aggregate packet dropping rates are not significantly different with Standard ECN, ECN+ with NoWaiting, or ECN+ with Waiting. In particular, the simulations show that in periods of very high congestion the packet-marking rate is low with or without ECN+, and the use of ECN+ does not significantly increase the number of dropped or marked packets.

The simulations show that ECN+ is most effective in times of moderate congestion. In these moderate-congested scenarios, the use of ECN+ increases the number of ECN-marked packets, because ECN+ allows SYN/ACK packets to be ECN-marked. At the same time, in these times of moderate congestion, the use of ECN+ instead of Standard ECN does not significantly affect the overall levels of congestion.

The simulations show that the use of ECN+ is less effective in times of high congestion; the simulations show that in times of high congestion more packets are dropped instead of marked, both with Standard ECN and with ECN+. In times of high congestion, the buffer can overflow, even with Active Queue Management and ECN; when the buffer is full arriving packets are dropped rather than marked, whether the packets are ECN-capable or not. Thus while ECN+ is less effective in times of high congestion, it still doesn't result in a significant increase in the level of congestion. More details are given in the appendix.

8. Conclusions

This draft specifies a modification to RFC 3168 to allow TCP nodes to send SYN/ACK packets as being ECN-Capable. Making the SYN/ACK packet ECN-Capable avoids the high cost to a TCP transfer when a SYN/ACK packet is dropped by a congested router, by avoiding the resulting retransmit timeout. This improves the throughput of short connections. The sender of the SYN/ACK packet responds to an ECN mark by reducing its initial congestion window from two, three, or four segments to one segment, reducing the subsequent load from that connection on the network. The addition of ECN-capability to SYN/ACK packets is particularly beneficial in the server-to-client direction, where congestion is more likely to occur. In this case, the initial information provided by the ECN marking in the SYN/ACK packet enables the server to more appropriately adjust the initial load it places on the network.

Future work will address the more general question of adding ECN-Capability to relevant handshake packets in other protocols that use retransmission-based reliability in their setup phase (e.g., SCTP, DCCP, HIP, and the like).

9. Acknowledgements

We thank Anil Agarwal, Mark Allman, Wesley Eddy, Janardhan Iyengar, and Pasi Sarolahti for feedback on earlier versions of this draft.

A. Report on Simulations

This section reports on simulations showing the costs of adding ECN+ in highly-congested scenarios. This section also reports on simulations for a comparative evaluation between ECN+ with NoWaiting and ECN+ with Waiting.

The simulations are run with a range of file-size distributions. As a baseline, they use the empirical heavy-tailed distribution reported in [SCJO01], with a mean file size of around 7 KBytes. This flow-size distribution is manipulated by skewing the flow sizes towards lower and higher values to get distributions with mean file sizes of 3 KBytes, 5 KBytes, 14 KBytes and 17 KBytes. The congested link is 100 Mbps. RED is run in gentle mode, and arriving ECN-Capable packets are only dropped instead of marked if the buffer is full (and the router has no choice).

We explore two alternatives for a TCP node's response to a report of an ECN-marked SYN/ACK packet. With ECN+ with NoWaiting, the TCP node sends a data packet immediately (with an initial congestion window of one segment). With the alternative ECN+ with Waiting, the TCP node waits a round-trip time before sending a data packet; the sender already has one measurement of the round-trip time when the acknowledgement for the SYN/ACK packet is received.

In the tables below, ECN+ refers to ECN+ with NoWaiting, where the sender starts transmitting immediately, and ECN+/wait refers to ECN+ with Waiting, where the sender waits a round-trip time before sending a data packet into the network.

The simulation scripts are available on [ECN-SYN], along with graphs showing the distribution of response times for the TCP connections.

A.1. Simulations with RED in Packet Mode

The simulations with RED in packet mode and with the queue in packets show that ECN+ is useful in times of moderate congestion, though it adds little benefit in times of high congestion. The simulations show a minimal increase in levels of congestion with either ECN+ with Waiting or ECN+ with NoWaiting, either in terms of packet dropping or marking rates or in terms of the distribution of responses times. Thus, the simulations show no problems with ECN+ in times of high congestion, and no reason to use ECN+ with Waiting instead of ECN+ with NoWaiting.

Table 1 shows the congestion levels for simulations with RED in packet mode, with a queue in packets. To explore a worst-case scenario, these simulations use a traffic mix with an unrealistically small flow size distribution, with a mean flow size of 3 Kbytes. For each table showing a particular traffic load, the three rows show the number of packets dropped, the number of packets ECN-marked, and the aggregate packet drop rate, and the three columns show the simulations with Standard ECN, ECN+ (NoWaiting) and ECN+/wait.

The usefulness of ECN+: The first thing to observe is that for the simulations with the somewhat moderate load of 95%, with packet drop rates of 5-6%, the use of ECN+ or ECN+/wait more than doubled the number of packets marked. This indicates that with ECN+ or ECN+/wait, many SYN/ACK packets are marked instead of dropped.

No increase in congestion: The second thing to observe is that in all of the simulations, the use of ECN+ or ECN+/wait does not significantly increase the aggregate packet drop rate.

Comparing ECN+ and ECN+/wait: The third thing to observe is that there is little difference between ECN+ and ECN+/wait in terms of the aggregate packet drop rate. Thus, there is no congestion-related reason to prefer ECN+/wait over ECN+.

Traffic Load = 95%:

	ECN	ECN+	ECN+/wait
Dropped	74,645	64,034	64,983
Marked	7,639	17,681	16,914
Loss rate	6.05%	5.26%	5.33%

Traffic Load = 110%:

	ECN	ECN+	ECN+/wait
Dropped	161,644	163,620	165,196
Marked	4,375	6,653	6,144
Loss rate	10.38%	10.45%	10.53%

Traffic Load = 125%:

	ECN	ECN+	ECN+/wait
Dropped	257,671	268,161	264,437
Marked	2,885	3,712	3,359
Loss rate	14.52%	15.00%	14.83%

Traffic Load = 150%:

	ECN	ECN+	ECN+/wait
Loss rate	24.36%	24.61%	24.46%

Traffic Load = 200%:

	ECN	ECN+	ECN+/wait
Loss rate	29.99%	30.22%	30.23%

Table 1: Simulations with an average flow size of 3 Kbytes, RED in packet mode, queue in packets.

A.2. Simulations with RED in Byte Mode

Table 3 below shows simulations with RED in byte mode and the queue in bytes. Like the simulations with RED in packet mode, there is no significant increase in aggregate congestion with the use of ECN+ or ECN+/wait, and no congestion-related reason to prefer ECN+/wait over ECN+.

However, unlike the simulations with RED in packet mode, the simulations with RED in byte mode show little benefit from the use of

ECN+ or ECN+/wait, in that the packet marking rate with ECN+ or ECN+/wait is not much different than the packet marking rate with Standard ECN. This is because with RED in byte mode, small packets like SYN/ACK packets are rarely dropped or marked - that is, there is no drawback from the use of ECN+ in these scenarios, but not much need for ECN+ either, in a scenario where small packets are unlikely to be dropped or marked.

Traffic Load = 95%:

	ECN	ECN+	ECN+/wait
Dropped	13,044	13,323	14,855
Marked	18,880	19,175	19,049
Loss rate	1.13%	1.16%	1.29%

Traffic Load = 110%:

	ECN	ECN+	ECN+/wait
Dropped	84,809	83,013	83,564
Marked	4,086	4,644	4,826
Loss rate	5.90%	5.78%	5.81%

Traffic Load = 125%:

	ECN	ECN+	ECN+/wait
Dropped	157,305	157,435	158,368
Marked	2,183	2,363	2,663
Loss rate	9.89%	9.87%	9.93%

Table 3: Simulations with an average flow size of 3 Kbytes, RED in byte mode, queue in bytes.

Normative References

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

[RFC3168] K.K. Ramakrishnan, S. Floyd, and D. Black, The Addition of Explicit Congestion Notification (ECN) to IP, RFC 3168, Proposed Standard, September 2001.

Informative References

[ECN+] A. Kuzmanovic, The Power of Explicit Congestion Notification, SIGCOMM 2005.

[ECN-SYN] ECN-SYN web page with simulation scripts, URL to be added.

[F07] S. Floyd, "[BEHAVE] Response of firewalls and middleboxes to TCP SYN packets that are ECN-Capable?", August 2, 2007, email sent to the BEHAVE mailing list, URL "http://www1.ietf.org/mail-archive/web/behave/current/msg02644.html".`

[Kelson00] Dax Kelson, note sent to the Linux kernel mailing list, September 10, 2000.

[MAF05] A. Medina, M. Allman, and S. Floyd. Measuring the Evolution of Transport Protocols in the Internet, ACM CCR, April 2005.

[PI] C. Hollot, V. Misra, W. Gong, and D. Towsley, On Designing Improved Controllers for AQM Routers Supporting TCP Flows, April 1998.

[RED] Floyd, S., and Jacobson, V. Random Early Detection gateways for Congestion Avoidance . IEEE/ACM Transactions on Networking, V.1 N.4, August 1993.

[REM] S. Athuraliya, V. H. Li, S. H. Low and Q. Yin, REM: Active Queue Management, IEEE Network, May 2001.

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

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

[RFC2988] V. Paxson and M. Allman, Computing TCP's Retransmission Timer, RFC 2988, November 2000.

[RFC3042] M. Allman, H. Balakrishnan, and S. Floyd, Enhancing TCP's Loss Recovery Using Limited Transmit, RFC 3042, Proposed Standard, January 2001.

[RFC3360] S. Floyd, Inappropriate TCP Resets Considered Harmful, RFC 3360, August 2002.

[RFC3390] M. Allman, S. Floyd, and C. Partridge, Increasing TCP's Initial Window, RFC 3390, October 2002.

[SCJO01] F. Smith, F. Campos, K. Jeffay, D. Ott, What {TCP/IP} Protocol Headers Can Tell us about the Web, SIGMETRICS, June 2001.

[SYN-COOK] Dan J. Bernstein, SYN cookies, 1997, see also <http://cr.yp.to/syncookies.html>

[SBT07] M. Sridharan, D. Bansal, and D. Thaler, Implementation Report on Experiences with Various TCP RFCs, Presentation in the TSVAREA, IETF 68, March 2007. URL "http://www3.ietf.org/proceedings/07mar/slides/tsvarea-3/sld6.htm".

[Tools] S. Floyd and E. Kohler, Tools for the Evaluation of Simulation and Testbed Scenarios, Internet-draft draft-irtf-tmrg-tools-04, work in progress, July 2007.

IANA Considerations

There are no IANA considerations regarding this document.

Authors' Addresses

Aleksandar Kuzmanovic
Phone: +1 (847) 467-5519
Northwestern University
Email: akuzma at northwestern.edu
URL: http://cs.northwestern.edu/˜a

Amit Mondal
Northwestern University
Email: a-mondal at northwestern.edu

Sally Floyd
Phone: +1 (510) 666-2989
ICIR (ICSI Center for Internet Research)
Email: floyd@icir.org
URL: http://www.icir.org/floyd/

K. K. Ramakrishnan
Phone: +1 (973) 360-8764
AT&T Labs Research
Email: kkrama at research.att.com
URL: http://www.research.att.com/info/kkrama

Full Copyright Statement