Network Working Group   A. Kuzmanovic
Request for Comments: 5562   A. Mondal
Category: Experimental   Northwestern University
                                                              S. Floyd
                                                                  ICSI
                                                       K. Ramakrishnan
                                                     AT&T Labs Research
                                                             June 2009

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

Status of This Memo

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

Abstract

   The proposal in this document is Experimental. While it may be
   deployed in the current Internet, it does not represent a consensus
   that this is the best possible mechanism for the use of Explicit
   Congestion Notification (ECN) in TCP SYN/ACK packets.

   This document describes an optional, experimental modification to RFC
   3168 to allow TCP SYN/ACK packets to be ECN-Capable. For TCP, RFC
   3168 specifies setting an ECN-Capable codepoint on data packets, but
   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 retransmission timeout, this document describes 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 the initial TCP SYN/ACK packet as
   ECN-Capable can be of great benefit to the TCP connection, avoiding
   the severe penalty of a retransmission timeout for a connection that
   has not yet started placing a load on the network. The TCP responder
   (the sender of the SYN/ACK packet) must reply to a report of an ECN-
   marked SYN/ACK packet by resending a SYN/ACK packet that is not ECN-
   Capable. If the resent SYN/ACK packet is acknowledged, then the TCP
   responder reduces its initial congestion window from two, three, or
   four segments to one segment, thereby reducing the subsequent load
   from that connection on the network. If instead the SYN/ACK packet
   is dropped, or for some other reason the TCP responder does not
   receive an acknowledgement in the specified time, the TCP responder
   follows TCP standards for a dropped SYN/ACK packet (setting the
   retransmission timer).

Table of Contents

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 queuing 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 retransmission 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
   retransmission 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 of the data packet might have to wait for a delay of
   several round-trip times (RTT) using Limited Transmit [RFC3042].
   With the use of ECN, short flows are less likely to have packets
   dropped, sometimes avoiding unnecessary delays or costly
   retransmission timeouts.

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

RFC 3168 [RFC3168] specifies setting the ECN-Capable codepoint on TCP
data packets, but not on TCP SYN and SYN/ACK packets. RFC 3168
[RFC3168] specifies the negotiation of the use of ECN between the two
TCP endpoints in the TCP SYN and SYN-ACK exchange, using flags in the
TCP header. Erring on the side of being conservative, RFC 3168
[RFC3168] does not specify the use of ECN for the first SYN/ACK

packet itself. However, because of the high cost to the TCP transfer
of having a SYN/ACK packet dropped, with the resulting retransmission
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 retransmission 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 (a SYN/ACK packet with the CE codepoint set in
the ECN field in the IP header) by sending a non-ECN-Capable SYN/ACK
packet, and 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 retransmission timeout;

2) Improvement in the throughput of short connections.

This document specifies a modification to RFC 3168 [RFC3168] 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 specified change.

2. Conventions and Terminology

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 [RFC2119].

We use the following terminology from RFC 3168 [RFC3168]:

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.

In this document, we use the terms "initiator" and "responder" to
refer to the sender of the SYN packet and of the SYN-ACK packet,
respectively.

3. Specification

This section specifies the modification to RFC 3168 [RFC3168] to
allow TCP SYN/ACK packets to be ECN-Capable.

Section 6.1.1 of RFC 3168 [RFC3168] 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 initial 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 [RFC3168], if TCP node B is willing to use ECN,
node B responds with an ECN-setup SYN-ACK packet.

3.1. SYN/ACK Packets Dropped in the Network

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

```
      ----------------------------------------------------------------
          TCP Node A              Router                 TCP Node B
          (initiator)                                    (responder)
          ----------              ------                 ----------

          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)
      ----------------------------------------------------------------
```
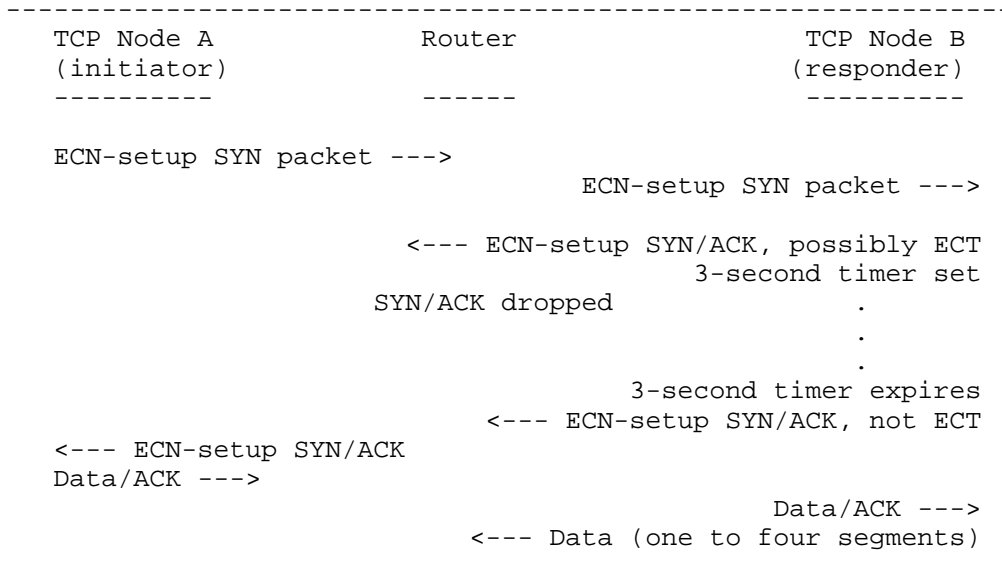
         Figure 1: SYN exchange with the SYN/ACK packet dropped

   If the SYN/ACK packet is dropped in the network, the responder (node
   B) responds by waiting three seconds for the retransmission timer to
   expire [RFC2988]. If a SYN/ACK packet with the ECT codepoint is
   dropped, the responder 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 the responder (node B) in
   the exchange in Figure 1, the responder wouldn't set a timer upon
   transmission of the SYN/ACK packet [SYN-COOK] [RFC4987]. 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.

3.2. SYN/ACK Packets ECN-Marked in the Network

   Figure 2 shows an interchange with the SYN/ACK packet sent as ECN-
   Capable, and ECN-marked instead of dropped at the congested router.
   This document specifies ECN+/TryOnce, which differs from the original
   proposal for ECN+ in [ECN+]; with ECN+/TryOnce, if the TCP responder
   is informed that the SYN/ACK was ECN-marked, the TCP responder
   immediately sends a SYN/ACK packet that is not ECN-Capable. The TCP
   responder is only allowed to send data packets after the TCP
   initiator reports the receipt of a SYN/ACK packet that is not ECN-
   marked.

```
       ---------------------------------------------------------------
           TCP Node A              Router               TCP Node B
           (initiator)                                  (responder)
           ----------             ------                ----------

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

                                         <--- ECN-setup SYN/ACK, ECT
                                                  3-second timer set
                             <--- Sets CE on SYN/ACK
           <--- ECN-setup SYN/ACK, CE

           ACK, ECN-Echo --->
                                              ACK, ECN-Echo --->
                                 Window reduced to one segment.
                                 <--- ECN-setup SYN/ACK, not ECT
           <--- ECN-setup SYN/ACK

           Data/ACK, ECT --->
                                              Data/ACK, ECT --->
                                 <--- Data, ECT (one segment only)
       ---------------------------------------------------------------
```
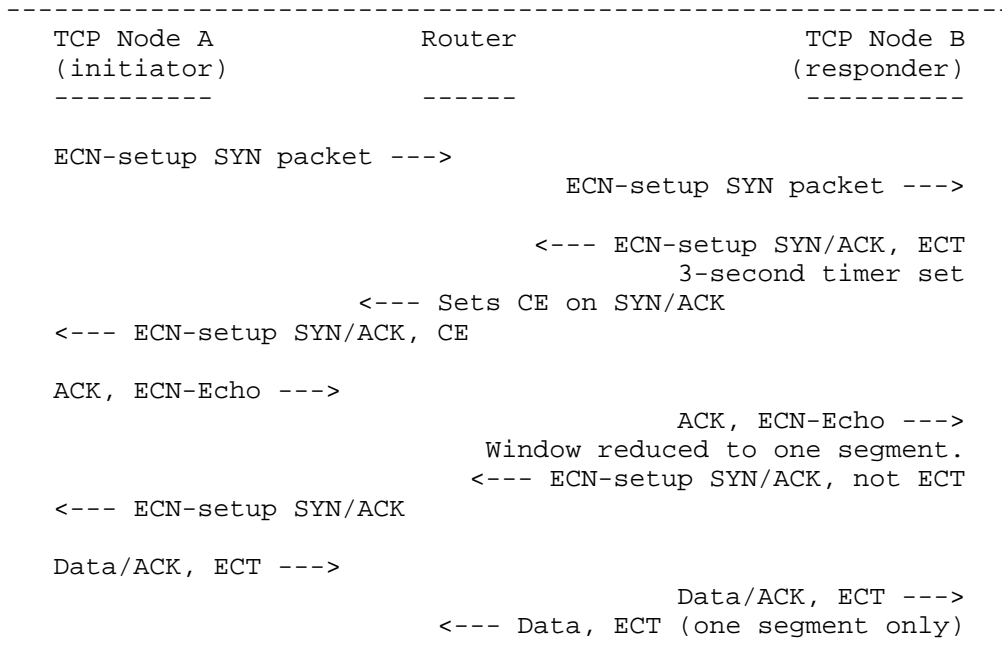
           Figure 2: SYN exchange with the SYN/ACK packet marked -
                               ECN+/TryOnce

   If the initiator (node A) receives a SYN/ACK packet that has been
   ECN-marked by the congested router, with the CE codepoint set, the
   initiator restarts the retransmission timer. The initiator responds
   to the ECN-marked SYN/ACK packet by setting the ECN-Echo flag in the
   TCP header of the responding ACK packet. The initiator uses the
   standard rules in setting the cumulative acknowledgement field in the
   responding ACK packet.

The initiator does not advance from the "SYN-Sent" to the
"Established" state until it receives a SYN/ACK packet that is not
ECN-marked.

When the responder (node B) receives the ECN-Echo packet reporting
the Congestion Experienced indication in the SYN/ACK packet, the
responder sets the initial congestion window to one segment, instead
of two segments as allowed by [RFC2581], or three or four segments
allowed by [RFC3390]. As illustrated in Figure 2, if the responder
(node B) receives an ECN-Echo packet informing it of a Congestion
Experienced indication on its SYN/ACK packet, the responder sends a
SYN/ACK packet that is not ECN-Capable, in addition to setting the
initial window to one segment. The responder does not advance the
send sequence number. The responder also sets the retransmission
timer. The responder follows RFC 2988 [RFC2988] in setting the RTO
(retransmission timeout).

The TCP hosts follow the standard specification for the response to
duplicate SYN/ACK packets (e.g., Section 3.4 of RFC 793 [RFC793]).

We note that the mechanism in this document differs from RFC 3168
[RFC3168], which specifies that "the sending TCP MUST restart the
retransmission timer on receiving the ECN-Echo packet when the
congestion window is one". RFC 3168 [RFC3168] does not allow SYN/ACK
packets to be ECN-Capable. RFC 3168 [RFC3168] specifies that in
response to an ECN-Echo packet, the TCP responder 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. In contrast, in
response to an ECN-Echo packet acknowledging the receipt of an ECN-
Capable SYN/ACK packet, the TCP responder doesn't set the CWR flag,
but simply sends a SYN/ACK packet that is not ECN-Capable. On
receiving the non-ECN-Capable SYN/ACK packet, the TCP initiator
clears the ECN-Echo flag on replying packets.

```
       ----------------------------------------------------------------
           TCP Node A              Router                  TCP Node B
           (initiator)                                     (responder)
           ----------             ------                   ----------

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

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

           ACK, ECN-Echo --->
                                                   ACK, ECN-Echo --->
                                       Window reduced to one segment.

                                       <--- ECN-setup SYN/ACK, not ECT
                                                   3-second timer set
                               SYN/ACK dropped              .
                                                            .
                                                            .
                                               3-second timer expires
                                       <--- ECN-setup SYN/ACK, not ECT
           <--- ECN-setup SYN/ACK, not ECT
           Data/ACK, ECT --->
                                                   Data/ACK, ECT --->
                               <--- Data, ECT (one segment only)
       ----------------------------------------------------------------
```
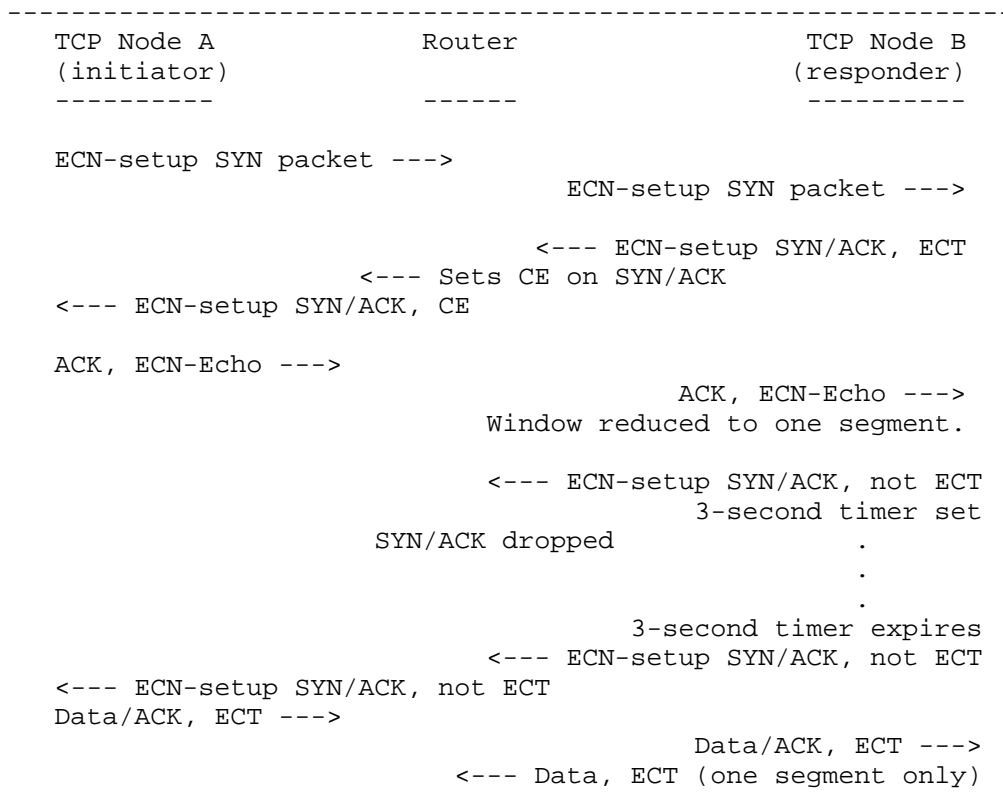
         Figure 3: SYN exchange with the first SYN/ACK packet marked
             and the second SYN/ACK packet dropped - ECN+/TryOnce

   In contrast to Figure 2, Figure 3 shows an interchange where the
   first SYN/ACK packet is ECN-marked and the second SYN/ACK packet is
   dropped in the network. As in Figure 2, the TCP responder sets a
   timer when the second SYN/ACK packet is sent. Figure 3 shows that if
   the timer expires before the TCP responder receives an
   acknowledgement for the other end, the TCP responder resends the
   SYN/ACK packet, following the TCP standards.

3.3. Management Interface

   The TCP implementation using ECN-Capable SYN/ACK packets should
   include a management interface to allow the use of ECN to be turned
   off for SYN/ACK packets. This is to deal with possible backwards
   compatibility problems such as those discussed in Appendix B.

4. Discussion

   The rationale for the specification in this document 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 retransmission timeouts resulting from the drop of SYN/ACK
   packets.

4.1. Flooding Attacks

   Setting an ECN-Capable codepoint in the responding TCP SYN/ACK
   packets does not raise any new or additional 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.

4.2. 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.

4.3. 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 Random
Early Detection (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 queuing
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 (see Section 10 of [Tools]). 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).

4.4. 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 end node's response to an ECN-
marked packet. Section 5 of RFC 3168 [RFC3168] 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 [RFC3168] specifies that
when the TCP congestion window consists of a single packet and that
packet is ECN-marked in the network, then the data 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 another 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 response of the
responder setting the initial congestion window to one packet,
instead of its allowed default value of two, three, or four packets.
The responder sends a non-ECN-Capable SYN/ACK packet, and proceeds

with a cautious sending rate of one data packet per round-trip time
after that SYN/ACK packet is acknowledged. 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,
there is still a difference in performance between the original
mechanism ECN+ and the mechanism ECN+/TryOnce specified in this
document. In particular, with ECN+/TryOnce, the TCP originator does
not send data packets until it has received a non-ECN-marked SYN/ACK
packet from the other end.

5. Related Work

The addition of ECN-Capability to TCP's SYN/ACK packets was initially
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 retransmission 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 coexistence 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 Benefits 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 retransmission 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 retransmission 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, and resending a SYN/ACK packet that is not ECN-Capable. We
   call this ECN+/TryOnce.

   However, Section 4 discussed two other possible responses to an ECN-
   marked SYN/ACK packet. In ECN+, the original proposal from [ECN+],
   the end node responds to the report of an ECN-marked SYN/ACK packet
   by setting the initial congestion window to one segment and
   immediately sending a data packet, if it has one to send. In
   ECN+/Wait, the end node responds to the report of an ECN-marked
   SYN/ACK packet by setting the initial congestion window to one
   segment and waiting an RTT before sending a data packet.

Simulations comparing the performance with Standard ECN (without
ECN-marked SYN/ACK packets), ECN+, ECN+/Wait, and ECN/TryOnce show
little difference, in terms of aggregate congestion, between ECN+ and
ECN+/Wait. However, for some scenarios with queues that are packet-
based rather than byte-based, and with packet drop rates above 25%
without ECN+, the use of ECN+ or of ECN+/Wait can more than double
the packet drop rates to greater than 50%. The details are given in
Tables 1 and 3 of Appendix A below. ECN+/TryOnce does not increase
the packet drop rate in scenarios of high congestion. Therefore,
ECN+/TryOnce is superior to ECN+ or to ECN+/Wait, which both
significantly increase the packet drop rate in scenarios of high
congestion. At the same time, ECN+/TryOnce gives a performance
improvement similar to that of ECN+ or ECN+/Wait (Tables 2 and 4 of
Appendix A).

Our conclusions are that ECN+/TryOnce is safe, and has significant
benefits to the user, and avoids the problems of ECN+ or ECN+/Wait
under extreme levels of congestion. As a consequence, this document
specifies the use of ECN+/TryOnce.

Note: We only discovered the occasional congestion-related problems
of ECN+ and of ECN+/Wait when re-running the simulations with an
updated version of the ns-2 simulator, after the document had almost
completed the standardization process.

6.3. Experiments

This section discusses experiments that would be useful before a
widespread deployment of ECN-Capability for TCP SYN/ACK packets.

Section 7.1 below discusses some of the known deployment problems of
ECN, in terms of routers or middleboxes that react inappropriately to
packets that use ECN codepoints in the IP or TCP packet headers. One
goal of a measurement study of ECN-Capability for TCP SYN/ACK packets
would be to determine if there were any routers or middleboxes that
react inappropriately to TCP SYN/ACK packets containing an ECN-
Capable or CE codepoint in the IP header. A second goal of a
measurement study would be to check the deployment status of older
TCP implementations that are ECN-Capable, but that don't respond to
ECN-Capability for SYN/ACK packets. (This is discussed in more
detail in Appendix B below.)

Following the discussion in Section 6.2, an experimental study could
explore the use of ECN-Capability for TCP SYN/ACK packets in highly
congested environments with ECN-Capable routers.

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.

7.1. "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 two new 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 (see figure 1 of [MAF05].) 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].

7.2. 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 responder sends a SYN/ACK packet that is
   not ECN-Capable; if this SYN/ACK packet is dropped, the responder
   then waits for a retransmission timeout, as specified in the TCP
   standards. 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 or with ECN+/TryOnce. However, in our simulations,
we have one scenario where ECN+ or ECN+/Wait results in a
significantly higher packet drop rate than ECN or ECN+/TryOnce
(Tables 1 and 3 in Appendix A below).

8. Conclusions

This document specifies a modification to RFC 3168 [RFC3168] 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 retransmission timeout. This improves the throughput
of short connections. This document specifies the ECN+/TryOnce
mechanism for ECN-Capability for SYN/ACK packets, where 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,
and sending a SYN/ACK packet that is not ECN-Capable. 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 appropriately
adjust the initial load it places on the network, while avoiding the
delay of a retransmission timeout.

9. Acknowledgements

We thank Anil Agarwal, Mark Allman, Remi Denis-Courmont, Wesley Eddy,
Lars Eggert, Alfred Hoenes, Janardhan Iyengar, and Pasi Sarolahti for
feedback on earlier working drafts of this document. We thank Adam
Langley [L08] for contributing a patch for ECN+/TryOnce for the Linux
development tree.

Appendix 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, ECN+,
   ECN+/Wait, and ECN+/TryOnce.

   The simulations are run with a range of file-size distributions,
   using the PackMime traffic generator in the ns-2 simulator. They all
   use a heavy-tailed distribution of file sizes. The simulations
   reported in the tables below use a mean file size of 3 Kbytes, to
   show the results with a traffic mix with a large number of small
   transfers. Other simulations were run with mean file sizes of 5
   Kbytes, 7 Kbytes, 14 Kbytes, and 17 Kbytes. The title of each chart
   gives the targeted average load from the traffic generator. Because
   the simulations use a heavy-tailed distribution of file sizes, and
   run for only 85 seconds (including ten seconds of warm-up time), the
   actual load is often much smaller than the targeted load. 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 three possible mechanisms for a TCP node's response to a
   report of an ECN-marked SYN/ACK packet. With ECN+, the TCP node
   sends a data packet immediately (with an initial congestion window of
   one segment). With ECN+/Wait, the TCP node waits a round-trip time
   before sending a data packet; the responder already has one
   measurement of the round-trip time when the acknowledgement for the
   SYN/ACK packet is received. With ECN+/TryOnce, the mechanism
   standardized in this document, the TCP responder replies to a report
   of an ECN-marked SYN/ACK packet by sending a SYN/ACK packet that is
   not ECN-Capable, and reducing the initial congestion window to one
   segment.

   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 or high congestion.
   However, for the simulations with a target load of 125%, with a
   packet loss rate of over 25% for ECN, ECN+ and ECN+/Wait both result
   in a packet loss rate of over 50%. (In contrast, the packet loss
   rate with ECN+/TryOnce is less than that of ECN alone.) For the
   distribution of response times, the simulations show that ECN+,
   ECN+/Wait, and ECN+/TryOnce all significantly improve the response
   times, when compared to the response times with Standard ECN.

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 four rows show the
number of packets dropped, the number of packets ECN-marked, the
aggregate packet drop rate, and the aggregate throughput. The four
columns show the simulations with Standard ECN, ECN+, ECN+/Wait, and
ECN+/TryOnce.

These simulations were run with RED set to mark instead of drop
packets any time that the queue is not full. This is a worst-case
scenario for ECN+ and its variants. For the default implementation
of RED in the ns-2 simulator, when the average queue size exceeds a
configured threshold, the router drops all arriving packets. For
scenarios with this RED mechanism, it is less likely that ECN+ or one
of its variants would increase the average queue size above the
configured threshold.

The usefulness of ECN+: The first thing to observe is that for all of
the simulations, the use of ECN+ or ECN+/Wait significantly increases
the number of packets marked. In contrast, the use of ECN+/TryOnce
significantly increases the number of packets marked in the
simulations with moderate congestion, and gives a more moderate
increase in the number of packets marked for the simulations with
higher levels of congestion. However, the cumulative distribution
function (CDF) in Table 2 shows that ECN+, ECN+/Wait, and
ECN+/TryOnce all improve response times for all of the simulations,
with moderate or with larger levels of congestion.

Little increase in congestion, sometimes: The second thing to observe
is that for the simulations with low or moderate levels of congestion
(that is, with packet drop rates less than 10%), the use of ECN+,
ECN+/Wait, and ECN+/TryOnce all decrease the aggregate packet drop
rate relative to the simulations with ECN. This makes sense, since
with low or moderate levels of congestion, ECN+ allows SYN/ACK
packets to be marked instead of dropped, and the use of ECN+ doesn't
add to the aggregate congestion. However, for the simulations with
packet drop rates of 15% or higher with ECN, the use of ECN+ or
ECN+/Wait increases the aggregate packet drop rate, sometimes even
doubling it.

Comparing ECN+, ECN+/Wait, and ECN+/TryOnce: The aggregate packet
drop rate is generally higher with ECN+/Wait than with ECN+. Thus,
there is no congestion-related reason to prefer ECN+/Wait over ECN+.
In contrast, the aggregate packet drop rate with ECN+/TryOnce is
often significantly lower than the aggregate packet drop rate with
either ECN, ECN+, or ECN+/Wait.

```
     Target Load = 95%:
               ECN         ECN+      ECN+/Wait    ECN+/TryOnce
               -------     -------    -------      ----------
     Dropped   20,516      11,226     11,735        16,755`
     Marked    30,586      37,741     37,425        40,764
     Loss rate  1.41%       0.78%      0.81%         1.02%
     Throughput  81%         81%        81%           81%


     Target Load = 110%:
               ECN         ECN+      ECN+/Wait    ECN+/TryOnce
               -------     -------    -------      ----------
     Dropped   165,566     106,083    147,180       208,422
     Marked    179,735     281,306    308,473       235,483
     Loss rate  9.01%       6.12%      8.02%         6.89%
     Throughput  92%         92%        92%           94%


     Target Load = 125%:
               ECN         ECN+      ECN+/Wait    ECN+/TryOnce
               -------     -------    -------      ----------
     Dropped   600,628   1,746,768  2,176,530       625,552
     Marked    418,433   1,166,450  1,164,932       439,847
     Loss rate  25.45%      51.73%     56.87%        18.31%
     Throughput  94%         98%        97%           95%


     Target Load =  150%
               ECN         ECN+      ECN+/Wait    ECN+/TryOnce
               -------     -------    -------      ----------
     Dropped  1,449,945  1,565,0517 1,563,0801    1,351,637
     Marked    669,840    583,378    591,315       684,715
     Loss rate  46.7%       59.0%      59.0%         32.7%
     Throughput  88%         94%        94%           92%
```

   Table 1: Simulations with an average flow size of 3 Kbytes, a 100
             Mbps link, RED in packet mode, queue in packets

```
   Target Load = 95%:
   TIME:    10  100  200  300  400  500 1000 2000 3000 4000 5000
           -------------------------------------------------------
   ECN:   0.00 0.07 0.26 0.51 0.82 0.96 0.97 0.97 0.97 1.00 1.00
   ECN+:  0.00 0.07 0.27 0.53 0.85 0.99 1.00 1.00 1.00 1.00 1.00
   Wait:  0.00 0.07 0.26 0.51 0.83 0.97 1.00 1.00 1.00 1.00 1.00
   Once:  0.00 0.07 0.24 0.49 0.83 0.97 1.00 1.00 1.00 1.00 1.00


   Target Load = 110%:
   TIME:    10  100  200  300  400  500 1000 2000 3000 4000 5000
           -------------------------------------------------------
   ECN:   0.00 0.05 0.19 0.41 0.67 0.79 0.80 0.80 0.80 0.96 0.96
   ECN+:  0.00 0.07 0.22 0.48 0.81 0.96 1.00 1.00 1.00 1.00 1.00
   Wait:  0.00 0.05 0.18 0.38 0.64 0.77 0.95 1.00 1.00 1.00 1.00
   Once:  0.00 0.06 0.19 0.42 0.70 0.86 0.95 0.96 0.96 0.99 0.99


   Target Load = 125%:
   TIME:    10  100  200  300  400  500 1000 2000 3000 4000 5000
           -------------------------------------------------------
   ECN:   0.00 0.04 0.13 0.27 0.46 0.56 0.58 0.59 0.59 0.82 0.82
   ECN+:  0.00 0.06 0.18 0.33 0.58 0.76 0.97 0.99 0.99 1.00 1.00
   Wait:  0.00 0.01 0.06 0.13 0.21 0.27 0.68 0.98 0.99 1.00 1.00
   Once:  0.00 0.05 0.16 0.34 0.58 0.73 0.85 0.87 0.87 0.95 0.96


   Target Load = 150%:
   TIME:    10  100  200  300  400  500 1000 2000 3000 4000 5000
           -------------------------------------------------------
   ECN:   0.00 0.03 0.08 0.18 0.31 0.39 0.42 0.42 0.43 0.68 0.68
   ECN+:  0.00 0.06 0.18 0.39 0.67 0.81 0.83 0.84 0.84 0.93 0.93
   Wait:  0.00 0.06 0.18 0.39 0.67 0.81 0.83 0.84 0.84 0.93 0.94
   Once:  0.00 0.04 0.13 0.27 0.46 0.59 0.72 0.75 0.75 0.88 0.88
```

   Table 2: The cumulative distribution function (CDF) for transfer
             times, for simulations with an average flow size of 3
             Kbytes, a 100 Mbps link, RED in packet mode, queue in
             packets (the graphs are available from
             "http://www.icir.org/floyd/ecn-syn/")

```
     Target Load =  95%
               ECN        ECN+      ECN+/Wait    ECN+/TryOnce
             -------    -------     -------      ----------
     Dropped   8,448      6,362       7,740       14,107
     Marked    9,891     16,787      17,456       16,132
     Loss rate  5.5%       4.3%        5.0%         5.0%
     Throughput  78%        78%         78%          81%

     Target Load =  110%
               ECN        ECN+      ECN+/Wait    ECN+/TryOnce
             -------    -------     -------      ----------
     Dropped  31,284     29,773      49,297       45,277
     Marked   28,429     54,729      60,383       34,622
     Loss rate 15.3%      15.2%       21.9%        13.6%
     Throughput  97%        96%         96%          94%

     Target Load =  125%
               ECN        ECN+      ECN+/Wait    ECN+/TryOnce
             -------    -------     -------      ----------
     Dropped  61,433    176,682     214,096       75,612
     Marked   44,408    119,728     117,301       49,442
     Loss rate 25.4%      51.9%       56.0%        22.3%
     Throughput  97%        98%         98%          96%

     Target Load =  150%
               ECN        ECN+      ECN+/Wait    ECN+/TryOnce
             -------    -------     -------      ----------
     Dropped 130,007    251,856     326,845      133,603
     Marked   63,066    146,757     147,239       66,444
     Loss rate 42.5%      61.3%       67.3%        31.7%
     Throughput  93%        99%         99%          94%
```

Table 3: Simulations with an average flow size of 3 Kbytes, a 10
          Mbps link, RED in packet mode, queue in packets

```
   Target Load = 95%:
   TIME:    10  100  200  300  400  500 1000 2000 3000 4000 5000
           ----------------------------------------------------------
   ECN:    0.00 0.05 0.18 0.42 0.70 0.86 0.88 0.88 0.88 0.98 0.98
   ECN+:   0.00 0.06 0.20 0.45 0.78 0.96 1.00 1.00 1.00 1.00 1.00
   Wait:   0.00 0.05 0.18 0.40 0.68 0.84 0.96 1.00 1.00 1.00 1.00
   Once:   0.00 0.05 0.18 0.40 0.71 0.88 0.96 0.97 0.97 0.99 0.99

   Target Load = 110%:
   TIME:    10  100  200  300  400  500 1000 2000 3000 4000 5000
           ----------------------------------------------------------
   ECN:    0.00 0.03 0.13 0.29 0.52 0.66 0.69 0.69 0.69 0.91 0.91
   ECN+:   0.00 0.05 0.17 0.36 0.66 0.88 0.98 0.99 1.00 1.00 1.00
   Wait:   0.00 0.02 0.08 0.20 0.35 0.47 0.76 0.98 1.00 1.00 1.00
   Once:   0.00 0.05 0.15 0.32 0.58 0.75 0.88 0.90 0.90 0.97 0.97


   Target Load = 125%:
   TIME:    10  100  200  300  400  500 1000 2000 3000 4000 5000
           ----------------------------------------------------------
   ECN:    0.00 0.03 0.10 0.22 0.40 0.52 0.56 0.56 0.57 0.82 0.82
   ECN+:   0.00 0.03 0.14 0.27 0.49 0.70 0.96 0.99 0.99 0.99 1.00
   Wait:   0.00 0.00 0.03 0.07 0.12 0.18 0.50 0.94 0.99 0.99 1.00
   Once:   0.00 0.04 0.13 0.28 0.51 0.66 0.81 0.84 0.84 0.94 0.94

   Target Load = 150%:
   TIME:    10  100  200  300  400  500 1000 2000 3000 4000 5000
           ----------------------------------------------------------
   ECN:    0.00 0.02 0.07 0.15 0.28 0.38 0.42 0.42 0.43 0.67 0.68
   ECN+:   0.00 0.00 0.00 0.00 0.01 0.05 0.68 0.83 0.95 0.97 0.98
   Wait:   0.00 0.00 0.00 0.00 0.00 0.00 0.10 0.62 0.83 0.93 0.97
   Once:   0.00 0.03 0.11 0.24 0.42 0.56 0.71 0.75 0.75 0.88 0.88
```

   Table 4: The cumulative distribution function (CDF) for transfer
             times, for simulations with an average flow size of 3
             Kbytes, a 10 Mbps link, RED in packet mode, queue in
             packets (the graphs are available from
             "http://www.icir.org/floyd/ecn-syn/")

A.2. Simulations with RED in Byte Mode

   Table 5 below shows simulations with RED in byte mode and the queue
   in bytes. There is no significant increase in aggregate congestion
   with the use of ECN+, ECN+/Wait, or ECN+/TryOnce.

   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.

```
Target Load = 95%
            ECN        ECN+     ECN+/Wait    ECN+/TryOnce
          -------    -------    -------      ----------
Dropped       766        446        427             408
Marked     32,683     34,289     33,412          31,892
Loss rate   0.05%      0.03%      0.03%           0.03%
Throughput    81%        81%        81%             81%


Target Load = 110%
            ECN        ECN+     ECN+/Wait    ECN+/TryOnce
          -------    -------    -------      ----------
Dropped     2,496      2,110      1,733           2,020
Marked    220,573    258,696    230,955         214,604
Loss rate   0.15%      0.13%      0.11%           0.11%
Throughput    92%        91%        92%             92%


Target Load = 125%
            ECN        ECN+     ECN+/Wait    ECN+/TryOnce
          -------    -------    -------      ----------
Dropped    20,032     13,555     13,979          16,918
Marked    725,165    726,992    726,823         615,235
Loss rate   1.11%      0.76%      0.78%           0.66%
Throughput    95%        95%        95%             96%


Target Load = 150%
            ECN        ECN+     ECN+/Wait    ECN+/TryOnce
          -------    -------    -------      ----------
Dropped   484,251    483,847    507,727         600,737
Marked    865,905    872,254    873,317         818,451
Loss rate  19.09%     19.13%     19.71%          12.66%
Throughput    99%        98%        99%             99%
```

Table 5: Simulations with an average flow size of 3 Kbytes, a 100
          Mbps link, RED in byte mode, queue in bytes

```
     Target Load =  95%
               ECN        ECN+      ECN+/Wait   ECN+/TryOnce
             -------    -------    -------     ----------
     Dropped      142         77        103          99
     Marked    11,694     11,387     11,604      12,129
     Loss rate    0.1%       0.1%       0.1%        0.1%
     Throughput    78%        78%        78%         78%


     Target Load =  110%
               ECN        ECN+      ECN+/Wait   ECN+/TryOnce
             -------    -------    -------     ----------
     Dropped      338        210        247         274
     Marked    41,676     40,412     44,173      36,265
     Loss rate    0.2%       0.1%       0.1%        0.1%
     Throughput    94%        94%        94%         96%


     Target Load =  125%
               ECN        ECN+      ECN+/Wait   ECN+/TryOnce
             -------    -------    -------     ----------
     Dropped    1,559        951        978       1,723
     Marked    74,933     75,499     75,481      59,670
     Loss rate    0.8%       0.5%       0.5%        0.6%
     Throughput    99%        99%        99%         96%


     Target Load =  150%
               ECN        ECN+      ECN+/Wait   ECN+/TryOnce
             -------    -------    -------     ----------
     Dropped    2,374      1,528      1,515       4,848
     Marked    85,739     86,428     86,144      81,350
     Loss rate    1.2%       0.8%       0.8%        1.4%
     Throughput    99%        98%        98%         98%
```

   Table 6: Simulations with an average flow size of 3 Kbytes, a 10
             Mbps link, RED in byte mode, queue in bytes

Appendix B. Issues of Incremental Deployment

   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 initiator ignores the
   CE codepoint on received SYN/ACK packets, this would mean that the
   TCP responder 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 responder would not reduce the initial
   congestion window from two, three, or four segments down to one
   segment, as it should, and would not sent a non-ECN-Capable SYN/ACK
   packet to complete the SYN exchange. However, the TCP end nodes
   would still respond correctly to any subsequent CE indications on
   data packets later on in the connection.

   Figure 4 shows an interchange with the SYN/ACK packet ECN-marked, but
   with the ECN mark ignored by the TCP originator.

```
      ----------------------------------------------------------------
         TCP Node A                Router              TCP Node B
         (initiator)                                   (responder)
         ----------                ------              ----------


         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, No ECN-Echo --->
                                              Data/ACK --->
                          <--- Data (up to four packets)
      ----------------------------------------------------------------
```
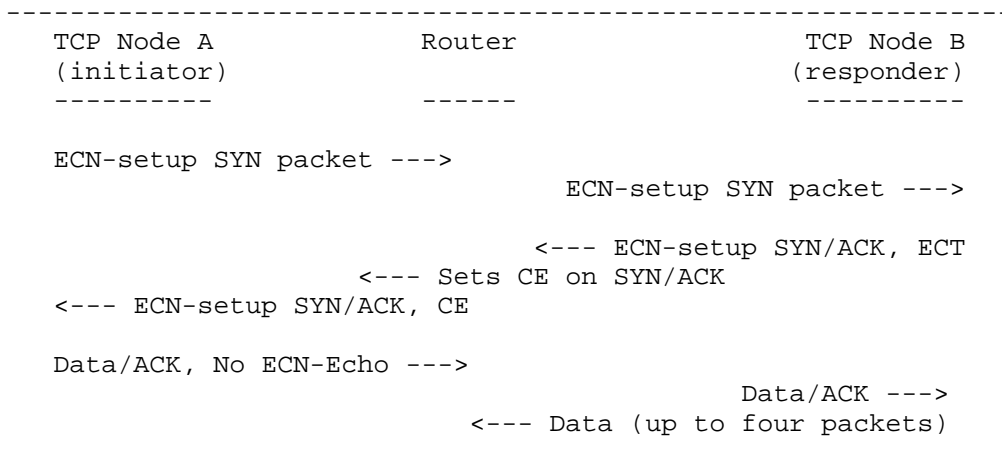
            Figure 4: SYN exchange with the SYN/ACK packet marked,
             but with the ECN mark ignored by the TCP initiator

   Thus, to be explicit, when a TCP connection includes an initiator
   that supports ECN but *does not* support ECN-Capability for SYN/ACK
   packets, in combination with a responder that *does* support ECN-
   Capability for SYN/ACK packets, it is possible that the ECN-Capable
   SYN/ACK packets will be marked rather than dropped in the network,
   and that the responder will not learn about the ECN mark on the
   SYN/ACK packet. This would not be a problem if most packets from the

responder supporting ECN for SYN/ACK packets were in long-lived TCP
connections, but it would be more problematic if most of the packets
were from TCP connections consisting of four data packets, and the
TCP responder for these connections was ready to send its data
packets immediately after the SYN/ACK exchange. Of course, with
severe congestion, the SYN/ACK packets would likely be dropped
rather than ECN-marked at the congested router, preventing the TCP
responder from adding to the congestion by sending its initial window
of four data packets.

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

One possibility for coping with problems of backwards compatibility
would be for TCP initiators to use a TCP flag that means "I
understand ECN-Capable SYN/ACK packets". If this document were to
standardize the use of such an "ECN-SYN" flag, then the TCP responder
would only send a SYN/ACK packet as ECN-Capable if the incoming SYN
packet had the "ECN-SYN" flag set. An ECN-SYN flag would prevent the
backwards compatibility problems described in the paragraphs above.

One drawback to the use of an ECN-SYN flag is that it would use one
of the four remaining reserved bits in the TCP header for a transient
backwards compatibility problem. This drawback is limited by the
fact that the "ECN-SYN" flag would be defined only for use with ECN-
setup SYN packets; that bit in the TCP header could be defined to
have other uses for other kinds of TCP packets.

Factors in deciding not to use an ECN-SYN flag include the following:

(1) The limited installed base: At the time that this document was
    written, the TCP implementations in Microsoft Vista and Mac OS X
    included ECN, but ECN was not enabled by default [SBT07]. Thus,
    there was not a large deployed base of ECN-Capable TCP
    implementations. This limits the scope of any backwards
    compatibility problems.

(2) Limits to the scope of the problem: The backwards compatibility
    problem would not be serious enough to cause congestion collapse;
    with severe congestion, the buffer at the congested router will
    overflow, and the congested router will drop rather than ECN-mark

arriving SYN packets. Some active queue management mechanisms
might switch from packet-marking to packet-dropping in times of
high congestion before buffer overflow, as recommended in Section
19.1 of RFC 3168 [RFC3168]. This helps to prevent congestion
collapse problems with the use of ECN.

(3) Detection of and response to backwards-compatibility problems: A
    TCP responder such as a web server can't differentiate between a
    SYN/ACK packet that is not ECN-marked in the network, and a
    SYN/ACK packet that is ECN-marked, but where the ECN mark is
    ignored by the TCP initiator. However, a TCP responder *can*
    detect if a SYN/ACK packet is sent as ECN-capable and not
    reported as ECN-marked, but data packets are dropped or marked
    from the initial window of data. We will call this scenario
    "initial-window-congestion". If a web server frequently
    experienced initial-window-congestion (without SYN/ACK
    congestion), then the web server *might* be experiencing
    backwards compatibility problems with ECN-Capable SYN/ACK
    packets, and could respond by not sending SYN/ACK packets as
    ECN-Capable.

Normative References

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

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

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

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

Informative References

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

   [ECN-SYN]   ECN-SYN web page with simulation scripts,
               http://www.icir.org/floyd/ecn-syn.

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

[Kelson00] Dax Kelson, "8% of the Internet unreachable!", September
            10, 2000, email to the Linux kernel mailing list,
            http://lkml.indiana.edu/hypermail/linux/kernel/
            0009.1/0329.html.

[L08]   A. Landley, "Re: [tcpm] I-D Action:draft-ietf-tcpm-
            ecnsyn-06.txt", August 24, 2008, email to the tcpm mailing
            list, http://www.ietf.org/
            mail-archive/web/tcpm/current/msg03988.html.

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

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

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

[RFC3360] Floyd, S., "Inappropriate TCP Resets Considered Harmful",
            BCP 60, RFC 3360, August 2002.

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

   [RFC4987]   Eddy, W., "TCP SYN Flooding Attacks and Common
               Mitigations", RFC 4987, August 2007.

   [SCJO01]   F. Smith, F. Campos, K. Jeffay, and 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.
               http://www3.ietf.org/proceedings/07mar/slides/tsvarea-
               3/sld6.htm.

   [Tools]   S. Floyd, Ed., and E. Kohler, Ed., "Tools for the
               Evaluation of Simulation and Testbed Scenarios", Work in
               Progress, February 2008.

Authors' Addresses

   Aleksandar Kuzmanovic
   Northwestern University

   Phone: +1 (847) 467-5519
   EMail: akuzma@northwestern.edu
   URL: http://cs.northwestern.edu/~akuzma

   Amit Mondal
   Northwestern University

   Phone: +1 (847) 467-6455
   EMail: a-mondal@northwestern.edu
   URL: http://www.cs.northwestern.edu/~akm175/

   Sally Floyd
   ICIR (ICSI Center for Internet Research)

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

   K. K. Ramakrishnan
   AT&T Labs Research
   Rm. A161
   180 Park Ave.
   Florham Park, NJ 07932

   Phone: +1 (973) 360-8764
   EMail: kkrama@research.att.com
   URL: http://www.research.att.com/info/kkrama