TCP Maintenance and Minor Extensions (tcpm)
Internet-Draft
Intended status: Informational
Expires: April 18, 2013

M. Kuehlewind, Ed.
University of Stuttgart
R. Scheffenegger
NetApp, Inc.

October 15, 2012

Problem Statement and Requirements for a More Accurate ECN Feedback
draft-kuehlewind-tcpm-accecn-reqs-00

Abstract

Explicit Congestion Notification (ECN) is an IP/TCP mechanism where network nodes can mark IP packets instead of dropping them to indicate congestion to the end-points. An ECN-capable receiver will feedback this information to the sender. ECN is specified for TCP in such a way that only one feedback signal can be transmitted per Round-Trip Time (RTT). Recently, new TCP mechanisms like ConEx or DCTCP need more accurate ECN feedback information in the case where more than one marking is received in one RTT. This documents specifies requirement for different ECN feedback scheme in the TCP header to provide more than one feedback signal per RTT.

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on April 18, 2013.

Copyright Notice

Table of Contents

1. Introduction

Explicit Congestion Notification (ECN) [RFC3168] is an IP/TCP mechanism where network nodes can mark IP packets instead of dropping them to indicate congestion to the end-points. An ECN-capable receiver will feedback this information to the sender. ECN is specified for TCP in such a way that only one feedback signal can be transmitted per Round-Trip Time (RTT). Recently, proposed mechanisms like Congestion Exposure (ConEx) or DCTCP [Ali10] need more accurate ECN feedback information in case when more than one marking is received in one RTT.

The following scenarios should briefly show where the accurate feedback is needed or provides additional value:

A Standard (RFC5681) TCP sender that supports ConEx:
: In this case the congestion control algorithm still ignores multiple marks per RTT, while the ConEx mechanism uses the extra information per RTT to re-echo more precise congestion information.

A sender using DCTCP congestion control without ConEx:
: The congestion control algorithm uses the extra info per RTT to perform its decrease depending on the number of congestion marks.

A sender using DCTCP congestion control and supports ConEx:
: Both the congestion control algorithm and ConEx use the accurate ECN feedback mechanism.

A standard TCP sender (using RFC5681 congestion control algorithm) without ConEx:
: No accurate feedback is necessary here. The congestion control algorithm still react only on one signal per RTT. But it is best to have one generic feedback mechanism, whether it is used or not.

This documents ...

1.1. Requirements Language

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 [RFC2119].

We use the following terminology from [RFC3168] and [RFC3540]:

The ECN field in the IP header:

CE: the Congestion Experienced codepoint, and

ECT(0): the first ECN-Capable Transport codepoint, and

ECT(1): the second ECN-Capable Transport codepoint.

The ECN flags in the TCP header:

CWR: the Congestion Window Reduced flag,

ECE: the ECN-Echo flag, and

NS: ECN Nonce Sum.

In this document, we will call the ECN feedback scheme as specified in [RFC3168] the 'classic ECN' and our new proposal the 'more accurate ECN feedback' scheme. A 'congestion mark' is defined as an IP packet where the CE codepoint is set. A 'congestion event' refers to one or more congestion marks belong to the same overload situation in the network (usually during one RTT).

2. Overview ECN and ECN Nonce in IP/TCP

ECN requires two bits in the IP header. The ECN capability of a

packet is indicated when either one of the two bits is set. An ECN sender can set one or the other bit to indicate an ECN-capable transport (ECT) which results in two signals, ECT(0) and ECT(1). A network node can set both bits simultaneously when it experiences congestion. When both bits are set the packet is regarded as "Congestion Experienced" (CE).

In the TCP header the first two bits in byte 14 are defined for the use of ECN. The TCP mechanism for signaling the reception of a congestion mark uses the ECN-Echo (ECE) flag in the TCP header. To enable the TCP receiver to determine when to stop setting the ECN-Echo flag, the CWR flag is set by the sender upon reception of the feedback signal. This leads always to a full RTT of ACKs with ECE set. Thus any additional CE markings arriving within this RTT can not signaled back anymore.

ECN-Nonce [RFC3540] is an optional addition to ECN that is used to protect the TCP sender against accidental or malicious concealment of marked or dropped packets. This addition defines the last bit of

byte 13 in the TCP header as the Nonce Sum (NS) bit. With ECN-Nonce a nonce sum is maintain that counts the occurrence of ECT(1) packets.

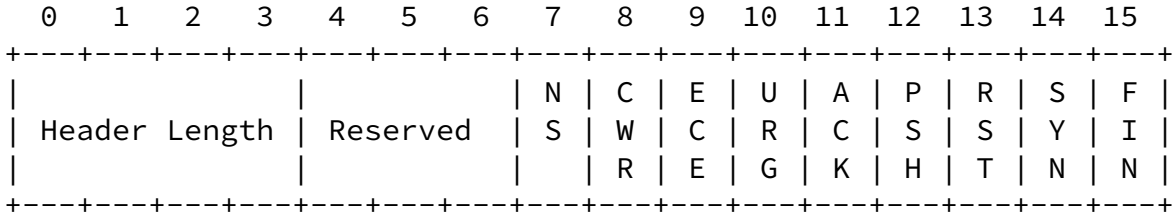

Figure 1: The (post-ECN Nonce) definition of the TCP header flags

3. Requirements

The requirements of the accurate ECN feedback protocol for the use of e.g. Conex or DCTCP are to have a fairly accurate (not necessarily perfect), timely and protected signaling. This leads to the following requirements:

Resilience
: The ECN feedback signal is carried within the TCP

acknowledgment. TCP ACKs can get lost. Moreover, delayed ACK are mostly used with TCP. That means in most cases only every second data packets triggers an ACK. In a high congestion situation where most of the packet are marked with CE, an accurate feedback mechanism must still be able to signal sufficient congestion information. Thus the accurate ECN feedback extension has to take delayed ACK and ACK loss into account.

Timely

The CE marking is induced by a network node on the transmission path and echoed by the receiver in the TCP acknowledgment. Thus when this information arrives at the sender, its naturally already about one RTT old. With a sufficient ACK rate a further delay of a small number of ACK can be tolerated but with large delays this information will be out dated due to high dynamic in the network. TCP congestion control which introduces parts of these dynamics operates on a time scale of one RTT. Thus the congestion feedback information should be delivered timely (within one RTT).

Integrity

With ECN Nonce, a misbehaving receiver or network node can be detected with a certain probability. As this accurate ECN feedback is reusing the NS bit, it is encouraged to ensure

integrity as least as good as ECN Nonce. If this is not possible, alternative approaches should be provided how a mechanism using the accurate ECN feedback extension can re-ensure integrity or give strong incentives for the receiver and network node to cooperate honestly.

Accuracy

Classic ECN feeds back one congestion notification per RTT, as this is supposed to be used for TCP congestion control which reduces the sending rate at most once per RTT. The accurate ECN feedback scheme has to ensure that if a congestion events occurs at least one congestion notification is echoed and received per RTT as classic ECN would do. Of course, the goal of this extension is to reconstruct the number of CE marking more accurately. However, a sender

should not assume to get the exact number of congestion marking in all situations.

Complexity
: Of course, the more accurate ECN feedback can also be used, even if only one ECN feedback signal per RTT is need. The implementation should be as simple as possible and only a minimum of addition state information should be needed. A proposal fulfilling this for a more accurate ECN feedback can then also be the standard ECN feedback mechanism.

4. Design Approaches

4.1. Re-use of Header Bits

The idea is to use the ECE, CWR and NS bits for additional capability negotiation during the TCP handshake exchange, and then for the more accurate ECN feedback itself on subsequent packets in the flow (where SYN is not set). This appraoch only provide a limited resiliency against ACK lost.

There have been several codings proposed so far: The one bit scheme sends one ECE for each CE received (+ redundancy in next ACK using the CWR bit). The 3 bit counter scheme uses all three bits for continuesly feeding the three most significant bits of a CE counter back. The 3 bit codepoint scheme encodes either a CE counter or an ECT(1) counter in 8 codepoints.

Discussion on ACK loss and ECN...

ToDo: Use of other header bit?

4.2. Use of Reserved Bits

As seen in Figure 1, there are currently three unused flag bits in the TCP header. The proposed scheme could be extended by one or more bits, to add higher resiliency against ACK loss. The relative gain would be proportionally higher resiliency against ACK loss, while the respective drawbacks would remain identical.

4.3. TCP Option

Alternatively, a new TCP option could be introduced, to help maintain the accuracy, and integrity of the ECN feedback between receiver and sender. Such an option could provide more information. E.g. ECN for RTP/UDP provides explicit the number of ECT(0), ECT(1), CE, non-ECT marked and lost packets. However, deploying new TCP options has its own challenges. A separate document proposes a new TCP Option for accurate ECN feedback [I-D.kuehlewind-tcpm-accurate-ecn-option]. This option could be used in addition to a more accurate ECN feedback scheme described here or in addition to classic ECN, when available and needed.

5. Acknowledgements

6. IANA Considerations

This memo includes no request to IANA.

7. Security Considerations

TBD

8. References

8.1. Normative References

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

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

[RFC3540] Spring, N., Wetherall, D., and D. Ely, "Robust Explicit Congestion Notification (ECN) Signaling with Nonces",

RFC 3540, June 2003.

8.2. Informative References

[Ali10] Alizadeh, M., Greenberg, A., Maltz, D., Padhye, J., Patel, P., Prabhakar, B., Sengupta, S., and M. Sridharan, "DCTCP: Efficient Packet Transport for the Commoditized Data Center", Jan 2010.

[I-D.briscoe-tsvwg-re-ecn-tcp] Briscoe, B., Jacquet, A., Moncaster, T., and A. Smith, "Re-ECN: Adding Accountability for Causing Congestion to TCP/IP", draft-briscoe-tsvwg-re-ecn-tcp-09 (work in progress), October 2010.

[I-D.kuehlewind-tcpm-accurate-ecn-option] Kuehlewind, M. and R. Scheffenegger, "Accurate ECN Feedback Option in TCP", draft-kuehlewind-tcpm-accurate-ecn-option-01 (work in progress), July 2012.

[RFC5562] Kuzmanovic, A., Mondal, A., Floyd, S., and K. Ramakrishnan, "Adding Explicit Congestion Notification (ECN) Capability to TCP's SYN/ACK Packets", RFC 5562, June 2009.

[RFC5681] Allman, M., Paxson, V., and E. Blanton, "TCP Congestion Control", RFC 5681, September 2009.

[RFC5690] Floyd, S., Arcia, A., Ros, D., and J. Iyengar, "Adding Acknowledgement Congestion Control to TCP", RFC 5690, February 2010.

Authors' Addresses

Mirja Kuehlewind (editor)
University of Stuttgart
Pfaffenwaldring 47
Stuttgart 70569
Germany

Email: mirja.kuehlewind@ikr.uni-stuttgart.de

Richard Scheffenegger
NetApp, Inc.
Am Euro Platz 2
Vienna, 1120
Austria

Phone: +43 1 3676811 3146
Email: rs@netapp.com