Congestion Exposure (ConEx)
Internet-Draft
Intended status: Experimental
Expires: June 23, 2012

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

December 21, 2011

TCP modifications for Congestion Exposure
draft-ietf-conex-tcp-modifications-00

Abstract

Congestion Exposure (ConEx) is a mechanism by which senders inform the network about the congestion encountered by previous packets on the same flow. This document describes the necessary modifications to use ConEx with the Transmission Control Protocol (TCP).

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 June 23, 2012.

Table of Contents

1. Introduction

Congestion Exposure (ConEx) is a mechanism by which senders inform the network about the congestion encountered by previous packets on the same flow. This document describes the necessary modifications to use ConEx with the Transmission Control Protocol (TCP). The ConEx signal is based on loss or ECN marks [RFC3168] as a congestion indication.

With standard TCP without Selective Acknowledgments (SACK) [RFC2018] the actual number of losses is hard to detect, thus we recommend to enable SACK when using ConEx. However, we discuss both cases, with and without SACK support, later on.

Explicit Congestion Notification (ECN) is defined in such a way that only a single congestion signal is guaranteed to be delivered per Round-trip Time (RTT). For ConEx a more accurate feedback signal would be beneficial. Such an extension to ECN is defined in a seperate document [draft-kuehlewind-conex-accurate-ecn], as it can also be useful for other mechanisms, as e.g. [DCTCP] or whenever the congestion control reaction should be proportional to the expirienced congestion.

ConEx is currently/will be defined as an destination option for IPv6. The use of four bits have been defined, namely the X (ConEx-capable), the L (loss experienced), the E (ECN experienced) and C (credit) bit.

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 [RFC2119].

2. Sender-side Modifications

A ConEx sender MUST negotitate for both SACK and ECN or the more accurate ECN feedback in the TCP handshake if these TCP extension are available at the sender. Depending on the capability of the receiver, the following operation modes exist:

- Full-ConEx (SACK and accurate ECN feedback)
- accECN-ConEx (no SACK but accurate ECN feedback)
- ECN-ConEx (no SACK and no accurate ECN feedback but 'classic' ECN)

- SACK-ECN-ConEx (SACK and 'classic' instead of accurate ECN)

- SACK-ConEx (SACK but no ECN at all)

- Basic-ConEx (neither SACK nor ECN)

A ConEx sender MUST expose congestion to the network according to the congestion information received by ECN or based on loss provided by the TCP feedback loop. A TCP sender MUST account congestion byte-wise (and not packet-wise) and MUST mark the respective number of payload bytes in subsequent packets (after the congestion notification) with the respective ConEx bit in the IP header. The congestion accounting based on different operation modes is described in the next section and the handling of the IPv6 bits itself in the subsequent section afterwards.

3. Accounting congestion

A TCP sender MUST account congestion byte-wise (and not packet-wise) based the congestion information received by ECN or loss detection provided by TCP. For this purpose a TCP sender will maintain two different counters for number outstanding bytes that need to be ConEx marked either with the E bit or the L Bit.

The outstanding bytes accounted based on ECN feedback information are maintained in the congestion exposure gauge (CEG). The accounting of these bytes from the ECN feedback is explained in more detail next.

The outstanding bytes for congestion indications based on loss are maintained in the loss exposure gauge (LEG) and the accounting is explained in subsequent to the CEG accounting.

The subtraction of bytes which have been ConEx marked from both counters is explained in the next section.

Usually all byte of an IP packet must be accounted. If we assume equal sized packets or at least equally distributed packet sizes the sender MAY only account the TCP payload bytes, as the ConEx marked packets as well as the original packets causing the congestion will both contain about the same number of headers. Otherwise the sender MUST take the headers into account. A sender which sends different sized packets with unequally distributed packet sizes should know about reason to do so and thus may be able to reconstruct the exact number of headers based on this information. Otherwise if no additional information is available the worse case number of headers SHOULD be estimated in a conservative way based on a minimum packet size (of all packets sent in the last RTT).

3.1. ECN

A receiver can support the accurate ECN feedback scheme, the 'classic' ECN or neither. In the case ECN is not supported at all, the transport is not ECN-capable and no ECN marks will occur, thus the E bit will never be set. In the other cases a ConEx sender MUST maintain a gauge for the number of outstanding bytes that has to be ConEx marked with the E bit, the congestion exposure gauge (CEG).

The CEG is increased when ECN information is received from an ECN-capable receiver supporting the 'classic' ECN scheme or the accurate ECN feedback scheme. When the ConEx sender receives an ACK indicating one or more segments were received with a CE mark, CEG is increased by the appropriate number of bytes. The two cases, depending on the receiver capability, are discussed in the following sections.

3.1.1. Accurate ECN feedback

With an more accurate ECN feedback scheme either the number of marked packets/received CE marks is know or the number of marked bytes directly. In the later case the CEG can directly be increased by the number of marked bytes. Otherwise when the accurate ECN feedback scheme is supported by the receiver, the receiver will maintain an echo congestion counter (ECC). The ECC will hold the number of CE marks received. A sender that is understanding the accurate ECN feedback will be able to reconstruct this ECC value on the sender side by maintaining a counter ECC.r.

On the arrival of every ACK, the sender calculates the difference D between the local ECC.r counter, and the signaled value of the receiver side ECC counter. The value of ECC.r is increased by D, and D is assumed to be the number of CE marked packets that arrived at the receiver since it sent the previously received ACK.

Whenever the counter ECC.r is increased, the gauge CEG has to be increased by the amount of bytes sent which were marked:

```
CEG += min( SMSS*D, acked_bytes )
```

3.1.2. Classic ECN support

A ConEx sender that communicates with a classic ECN receiver (conforming to [RFC3168] or [RFC5562]) MAY run in one of these modes:

- Full compliance mode:

  The ConEx sender fully conforms to all the semantics of the ECN

signaling as defined by [RFC5562]. In this mode, only a single congestion indication can be signaled by the receiver per RTT. Whenever the ECE flag toggles from "0" to "1", the gauge CEG is increased by the SMSS:

CEG += SMSS

Note that under severe congestion, a session adhering to these semantics may not provide enough ConEx marks. This may cause appropriate sanctions by an audit device in a ConEx enabled network.

- Simple compatibility mode:

The sender will set the CWR permanently to force the receiver to signal only one ECE per CE mark. Unfortunately, in a high congestion situation where all packets are CE marled over a certain period of time, the use of delayed ACKs, as it is usually done today, will prevent a feedback of every CE mark. With an ACK rate of m, about m-1/m CE indications will not be signaled back by the receiver (e.g. 50% with M=2). Thus, in this mode the ConEx sender MUST increase CEG by a count of M*SMSS for each received ECE signal:

CEG += M*SMSS

In case of a congestion event with low congestion (that means when only a very smaller number of packets get marked), the sender might miss the whole congestion event. In average the sender will sent sufficient ConEx marks due to the scheme proposed above but these ConEx marks might be timely shifted. Regarding congestion control it is not a general problem to miss a congestion event as by chance a marking scheme in the network node might also miss a certain flow. Even if then no other flow is reacting, the congestion level will increase and it will get more likely that the congestion feedback is delivered. But to provide a fair share over time, a TCP sender could react more strong when receiving a ECN feedback signal. This of course depends on the congestion control used. A TCP sender using this scheme MUST take the impact on congestion control into account.

- Advanced compatibility mode:

More sophisticated heuristics, such as a phase locked loop, to set CWR only on those data segments, that will actually trigger an (delayed) ACK, could extract congestion notifications more timely. A ConEx sender MAY choose to implement such a heuristic. In addition, further heuristics SHOULD be implemented, to determine

the value of each ECE notification. E.g. for each consecutive ACK received with the ECE flag set, CEG should be increased by min(M*SSMS, acked_bytes). Else if the predecessor ACK was received with the ECE flag cleared, CEG need only be increase by one SMSS:

```
if previous_marked: CEG += min( M*SSMS, acked_bytes)
else: CEG += SMSS
```

This heuristic is conservative during more serious congestion, and more relaxed at low congestion levels.

3.2. Loss Detection with/without SACK

For all the data segments that are determined by a ConEx sender as lost, an identical number of IP bytes MUST be be sent with the ConEx L bit set. Loss detection typically happens by use of duplicate ACKs, or the firing of the retransmission timer. A ConEx sender MUST maintain a loss exposure gauge (LEG), indicating the number of outstanding bytes that must be sent with the ConEx L bit. When a data segment is retransmitted, LEG will be increased by the size of the TCP payload packet containing the retransmission, assuming equal sized segments such that the retransmitted packet will have the same number of header as the original ones. When sending subsequent segments (including TCP control segments), the ConEx L bit is set as long as LEG is positive, and LEG is decreased by the size of the sent TCP payload with the ConEx L bit set.

Any retransmission may be spurious. To accommodate that, a ConEx sender SHOULD make use of heuristics to detect such spurious retransmissions (e.g. F-RTO [RFC5682], DSACK [RFC3708], and Eifel [RFC3522], [RFC4015]). When such a heuristic has determined, that a certain number of packets were retransmitted erroneously, the ConEx sender should subtract the payload size of these TCP packets from LEG.

Note that the above heuristics delays the ConEx signal by one segment, and also decouples them from the retransmissions themselves, as some control packets (e.g. pure ACKs, window probes, or window updates) may be sent in between data segment retransmissions. A simpler approach would be to set the ConEx signal for each retransmitted data segment. However, it is important to remember, that a ConEx signal and TCP segments do not natively belong together.

4. Setting the ConEx IPv6 Bits

ConEx is currently/will be defined as an destination option for IPv6. The use of four bits have been defined, namely the X (ConEx-capable),

the L (loss experienced), the E (ECN experienced) and C (credit) bit.

By setting the X bit a packet is marked as ConEx-capable. All packets carrying payload MUST be marked with the X bit set including retransmissions. About control packets as pure ACKs which are not carrying any payload no congestion feedback information is available thus these packet should not be taken into account when determining ConEx information. These packet MUST carry a ConEx Destination Option with the X bit unset.

4.1. Setting the E and the L Bit

As long as the CEG/LEG is positive, ConEx-capable packets MUST be marked with E or respective L and the CEG/LEG is decreased by the TCP payload bytes carried in this packet. If the CEG/LEG is negative, the CEG/LEG is drained by one byte with every packet sent out, as ConEX information are only meaningful for a certain time:

```
if CEG > 0: CEG -= TCPpayload.length else: CEG--
if LEG > 0: LEG -= TCPpayload.length else: LEG--
```

4.2. Credit Bits

The ConEx abstract mechanism requires that the transport SHOULD signal sufficient credit in advance to cover any reasonably expected congestion during its feedback delay. To be very conservative the number of credits would need to equal the number of packets in flight, as every packet could get lost or congestion marked. With a more moderate view, only an increase in the sending rate should cause congestion.

For TCP sender using the [RFC5681] congestion control algorithm, we recommend to only send credit in Slow Start, as in Congestion Avoidance an increase of one segment per RTT should only cause a minor amount of congestion marks (usually at max one). If a more aggressive congestion control is used, a sufficient amount of credits need to be set.

In TCP Slow Start the sending rate will increase exponentially and that means double every RTT. Thus the number of credits should equal half the number of packets in flight in every RTT. Under the assumption that all marks will not get invalid for the whole Slow Start phase, marks of a previous RTT have to be summed up. Thus the marking of every fourth packet will allow sufficient credits in Slow Start.

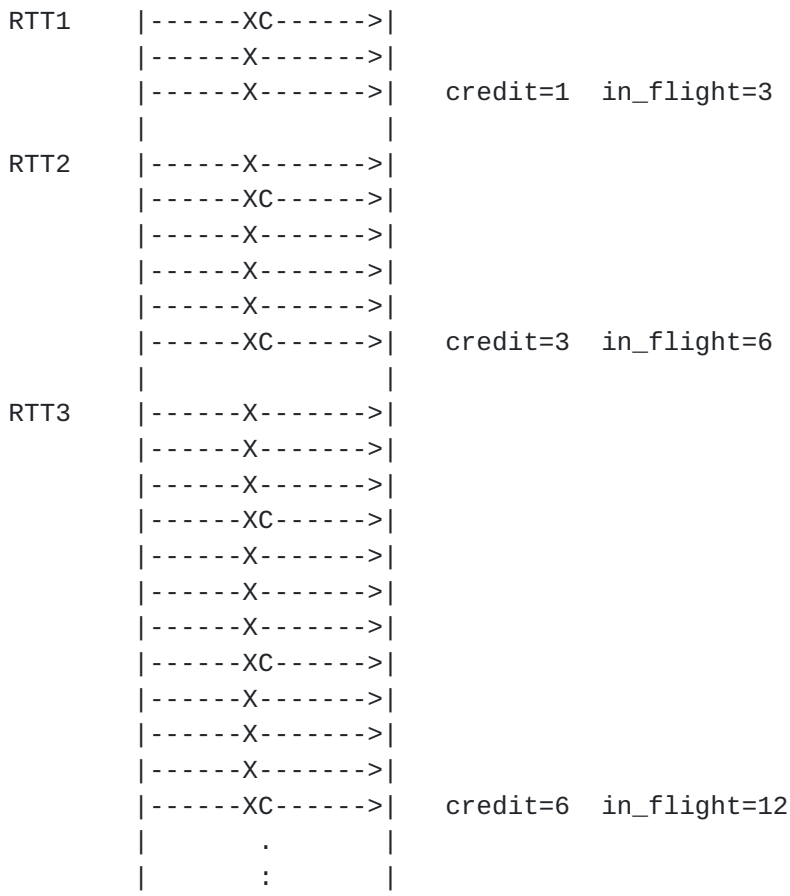

Figure 1: Credits in Slow Start (with an initial window of 3)

If a ConEx sender detects an increasing number of losses even though the sender reduced the sending rate, the sender SHOULD assume that those losses are incorporated by an audit device and thus should send further credits. Up to now its not clear if the credits say valid as long as the connection is established or if an expiration of the credits need to be assumed by the sender.

5. Timeliness of the ConEx Signals

ConEx signals will anyway be evaluated with a slight time delay of about one RTT by a network node. Therefore, it would not be absolutely necessary to immediately signal ConEx bits when they become known (e.g. L and E bits), but a sender SHOULD sent the ConEx signaling with the next available packet. If cases are available where it is preferable to slight delay the ConEx signal, the sender MUST NOT delay the ConEx signal more than one RTT.

Multiple ConEx bits may become available for signaling at the same time, for example when an ACK is received by the sender, that

indicates that at least one segment has been lost, and that one or more ECN marks were received at the same time. This may happen during excessive congestion, where buffer queues overflow and some packets are marked, while others have to be dropped nevertheless. Another possibility when this may happen are lost ACKs, so that a subsequent ACK carries summary information not previously available to the sender.

It is important to remember, that ConEx bits and TCP retransmissions do not interact with each other. However, a retransmission should be accompanied by one ConEx L bit in close proximity nevertheless. This does not mean, that TCP retransmissions may never contain ConEx marks. In a typical scenario using SACK, the first retransmission would not carry a ConEx L bit, while subsequent retransmissions in the same recovery episode, would be marked with the ConEx L bit. Spreading the ConEx bits over a small number of segments increases the likelihood that most devices along the path will see some ConEx marks even during heavy congestion.

6. Acknowledgements

7. IANA Considerations

8. Security Considerations

9. References

9.1. Normative References

[RFC2018] Mathis, M., Mahdavi, J., Floyd, S., and A. Romanow, "TCP Selective Acknowledgment Options", RFC 2018, October 1996.

[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.

[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.

9.2. Informative References

[DCTCP] 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.

[RFC3522] Ludwig, R. and M. Meyer, "The Eifel Detection Algorithm for TCP", RFC 3522, April 2003.

[RFC3708] Blanton, E. and M. Allman, "Using TCP Duplicate Selective Acknowledgement (DSACKs) and Stream Control Transmission Protocol (SCTP) Duplicate Transmission Sequence Numbers (TSNs) to Detect Spurious Retransmissions", RFC 3708, February 2004.

[RFC4015] Ludwig, R. and A. Gurtov, "The Eifel Response Algorithm for TCP", RFC 4015, February 2005.

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

[RFC5682] Sarolahti, P., Kojo, M., Yamamoto, K., and M. Hata, "Forward RTO-Recovery (F-RTO): An Algorithm for Detecting Spurious Retransmission Timeouts with TCP", RFC 5682, September 2009.

[draft-kuehlewind-conex-accurate-ecn] Kuehlewind, M. and R. Scheffenegger, "Accurate ECN Feedback in TCP", draft-kuehlewind-conex-accurate-ecn-00 (work in progress), Jun 2011.

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