Congestion Exposure (ConEx)
Internet-Draft
Intended status: Experimental
Expires: January 16, 2014
M. Kuehlewind, Ed.
University of Stuttgart
R. Scheffenegger
NetApp, Inc.
July 15, 2013

# TCP modifications for Congestion Exposure
## draft-ietf-conex-tcp-modifications-04

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).

Table of Contents

1. Introduction . . . . . . . . . . . . . . . . . . . . . . . . 2
1.1. Requirements Language . . . . . . . . . . . . . . . . . . . 3
2. Sender-side Modifications . . . . . . . . . . . . . . . . . . 3
3. Accounting congestion . . . . . . . . . . . . . . . . . . . . 4
3.1. ECN . . . . . . . . . . . . . . . . . . . . . . . . . . . 5
3.1.1. Accurate ECN feedback . . . . . . . . . . . . . . . . . 6
3.1.2. Classic ECN support . . . . . . . . . . . . . . . . . . 6
3.2. Loss Detection with/without SACK . . . . . . . . . . . . . 8
4. Setting the ConEx Bits . . . . . . . . . . . . . . . . . . . 9
4.1. Setting the E and the L Bit . . . . . . . . . . . . . . . 9
4.2. Credit Bits . . . . . . . . . . . . . . . . . . . . . . . 9
4.3. Loss of ConEx information . . . . . . . . . . . . . . . . 10
5. Timeliness of the ConEx Signals . . . . . . . . . . . . . . . 11
6. Acknowledgements . . . . . . . . . . . . . . . . . . . . . . 11
7. IANA Considerations . . . . . . . . . . . . . . . . . . . . . 11
8. Security Considerations . . . . . . . . . . . . . . . . . . . 11
9. References . . . . . . . . . . . . . . . . . . . . . . . . . 11
9.1. Normative References . . . . . . . . . . . . . . . . . . . 11
9.2. Informative References . . . . . . . . . . . . . . . . . . 12
Appendix A. Revision history . . . . . . . . . . . . . . . . . . 13
Authors' Addresses . . . . . . . . . . . . . . . . . . . . . . . 14

## 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. ConEx concepts and use cases are further explained in [RFC6789]. The abstract ConEx mechanism is explained in [draft-ietf-conex-abstract-mech]. This document describes the necessary modifications to use ConEx with the Transmission Control Protocol (TCP).

ConEx is defined as a destination option for IPv6 [draft-ietf-conex-destopt]. 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.

The ConEx signal is based on loss or Explicit Congestion Notification (ECN) marks [RFC3168] as a congestion indication. This congestion information is retrieved by the sender based on existing feedback mechanisms from the receiver to the sender in TCP.

This document describes mechanisms for both TCP with and without the Selective Acknowledgment (SACK) extension [RFC2018]. However, ConEx benefits from more accurate information about the number of packets dropped in the network. We therefore recommend using the SACK extension when using TCP with ConEx.

While loss-based congestion feedback should be minimized, ECN could actually provide more fine-grained feedback information. ConEx-based traffic measurement or management mechanism would benefit from this. Unfortunately the current ECN does not reflect multiple congestion markings which occur within the same Round-Trip Time (RTT). A more accurate feedback extension to ECN is defined in a separate document [draft-kuehlewind-tcpm-accurate-ecn], as this is also useful for other mechanisms.

### 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 negotiate for both SACK and ECN or the more accurate ECN feedback in the TCP handshake if these TCP extension are available at the sender. Thus a ConEx SHOULD also implement SACK and ECN. Depending on the capability of the receiver, the following operation modes exist:

- SACK-accECN-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 all congestion information to the network according to the congestion information received by ECN or based on loss information provided by the TCP feedback loop. A TCP sender SHOULD account congestion byte-wise (and not packet-wise). A sender MUST mark subsequent packets (after the congestion notification) with the respective ConEx bit in the IP header.

With SACK only the number of lost payload bytes is known, but not the number of packets carrying these bytes. With classic ECN only an indication is given that a marking occurred which is not giving an exact number of payload bytes nor packets. As network congestion is usually byte-congestion [draft-briscoe-tsvwg-byte-pkt-mark], the exact number of bytes should be taken into account if available to make the ConEx signal as exact as possible.

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 ConEx sender, thats accounts congestion byte-wise based on the congestion information received by ECN or loss detection provided by TCP, will maintain two different counters. These counters hold the 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 in Section 3.1.

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 in Section 3.2.

Furthermore, those counters will be reduced every time a ConEx capable packet with the E or L bit set is sent. This is explained from both counters in Section 4.1.

Usually all bytes of an IP packet must be accounted. Therefore the sender SHOULD take the headers into account, too. If equal sized packets, or at least equally distributed packet sizes can be assumed, the sender MAY only account the TCP payload bytes. In this case there should be about the same number of ConEx marked packets as the original packets that were causing the congestion. Thus both contain about the same number of header bytes. This case is assumed in the following sections.

Otherwise if this is not the case and a sender sends different sized packets (with unequally distributed packet sizes), the sender needs to memorize or estimate the number of ECN-marked or lost packets. A sender might be able to reconstruct the number of packets and thus the header bytes if the packet sizes of the last RTT are known. Otherwise if no additional information is available the worst 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). If the number of ConEx marked packets is smaller (or larger) than the estimated number of ECN-marked or lost packets, the additional header bytes should the added to (or can be subtracted from) the respective counter.

## 3.1. ECN

ECN [RFC3168] is an IP/TCP mechanism that allows network nodes to mark packets with the Congestion Experienced (CE) mark instead of (early) dropping them when congestion occurs. As soon as a CE mark is seen at the receiver, with classic ECN it will feed this information back to the sender by setting the Echo Congestion Experienced (ECE) bit in the TCP header of all subsequent ACKs until a packet with Congestion Window Reduced (CWR) bit in the TCP header is received to acknowledge the reception of the congestion notification. The sender sets the CWR bit in the TCP header once when the first ECE of a congestion notification is received.

A receiver can support 'classic' ECN, a more accurate ECN feedback scheme, or neither. In the case ECN is not supported at all, of course, no ECN marks will occur, thus the E bit will never be set. Otherwise, a ConEx sender must maintain a counter, the congestion exposure gauge (CEG), for the number of outstanding bytes that have to be ConEx marked with the E bit.

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.

Unfortunately in case of duplicate acknowledgements the number of newly acknowledged bytes will be zero even though (CE marked) data has been received. Therefore, we increase the CEG by DeliveredData, as defined below:

DeliveredData covers the number of bytes which has been newly delivered to the receiver. Therefore on each arrival of an ACK, DeliveredData will be calculated by the newly acknowledged bytes (acked_bytes) as indicated by the current ACK, relative to all past ACKs. Moreover with SACK, DeliveredData is increased by the number of bytes provided by (new) SACK information (SACK_diff). Note, if less unacknowledged bytes are announced in the new SACK information than in the previous ACK, SACK_diff can be negative. In this case, data is newly acknowledged (in acked_byte), that has previously already been accounted to DeliveredData based on SACK information.

Without SACK, DeliveredData is estimated to be 1 SMSS on duplicate acknowledgements.  For the subsequent partial or full ACK, DeliveredData is estimated to be the the newly acknowledged bytes, minus one SMSS for each preceding duplicate ACK.

DeliveredData = acked_bytes + SACK_diff + (is_dup)*1SMSS - (is_after_dup)*num_dup*1SMSS

Thus is_dup is one if the current ACK is a duplicated ACK without SACK, and zero otherwise.  is_after_dup is only one for the next full or partial ACK after a number of duplicated ACKs without SACK and num_dup counts the number of duplicated ACKs in a row.

The two cases, with and without more accurate ECN depending on the receiver capability, are discussed in the following sections.

### 3.1.1.  Accurate ECN feedback

With a more accurate ECN feedback scheme either the number of marked packets/received CE marks or directly the number of marked bytes is known.  In the later case the CEG can directly be increased by the number of marked bytes.  Otherwise if D is assumed to be the number of marks, the gauge CEG has to be increased by the amount of bytes sent which were marked:

CEG += min(SMSS*D, DeliveredData)

### 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:

o  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 at maximum by the SMSS:

   CEG += min(SMSS, DeliveredData)

   Note that most often, a session adhering to these semantics may not provide enough ConEx marks as usually more than one CE mark will occur during one congestion event (within one RTT).  We assume that the credits build up during the Slow Start phase will cover the mismatch for short connections with only light congestion.  Otherwise this will cause appropriate sanctions by an

audit device in a ConEx enabled network. To avoid this in any case, on whole RTT of packets need to be regarded as congestion marked. Thus increasing the CEG by the number of DeliveredData for each ACK with the ECE bit set, would cover the worst case estimation.

o Simple compatibility mode:

The sender will set the CWR permanently to force the receiver to signal only one ECE per CE mark. Unfortunately, the use of delayed ACKs [RFC5681], as it is usually done today, will prevent a feedback of every CE mark. An CWR confirmation will be received before the ECE can be sent out with the next ACK. 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 for delayed ACKs). Thus, in this mode the ConEx sender MUST increase CEG as if M congestion notification were received for each received ECE signal:

CEG += min(M*SMSS, DeliveredData + (M-1)*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. Even though the sender will send sufficient ConEx marks on average due to the scheme proposed above, these ConEx marks might be shifted in time and an audit might penalize this behavior. 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. In the case where no other flow is reacting, the congestion level will increase and it will get more likely that the congestion feedback is delivered. To provide a fair share over time, a TCP sender implementing this simple ECN compatibility mode could react more strongly when receiving an ECN feedback signal. This of course depends on the congestion control used.

o Advanced compatibility mode:

To avoid the loss of ECN feedback information in the proposed simple compatibility mode, a sender could set CWR only on those data segments, that will actually trigger a (delayed) ACK. The sender would need an additional control loop to estimated which data segment will trigger an ACK. Such a more sophisticated heuristics could extract congestion notifications more timely. In addition, if this advanced compatibility mode is used, 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, DeliveredData). Else if the predecessor ACK was received with the

ECE flag cleared, CEG need only be increase at maximum by one SMSS:

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

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, (at least) the same number of TCP payload 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 bytes 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, the ConEx L bit is set as long as LEG is positive, and LEG is decreased by the size of the sent TCP payload bytes 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.

If SACK is not available or SACK information has been reset for any reason, spurious retransmission are more likely. In this case it might be valuable to slightly delay the ConEx loss feedback until a spurious retransmission might be detected. But the ConEx signal MUST NOT be delayed more than one RTT.

## 4. Setting the ConEx Bits

ConEx is defined as a destination option for IPv6 [draft-ietf-conex-destopt]. 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. No congestion feedback information are available about control packets as pure ACKs which are not carrying any payload. 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 or LEG counter is positive, ConEx-capable packets SHOULD be marked with E or L respectively, and the CEG or LEG counter is decreased by the TCP payload bytes carried in this packet. If the CEG or LEG counter is negative, the respective counter SHOULD be reset to zero within one RTT after it was decreased the last time or one RTT after recovery if no further congestion occurred.

### 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 loss while the number of ECN markings within one RTT depends on parameterization of the used Active Queue management (AQM). The average or maximum number of ECN marks per congestion event could potentially be estimated over time. This case is not further expanded here.

In TCP Slow Start the sending rate will increase exponentially and that means double every RTT. Thus the number of credits should equal at least half the number of packets in flight in every RTT. If the used AQM is not overly aggressive with ECN marking, maintaining the number of credit as half the number of packets in flight should be sufficient for both, congestion signaled by loss or ECN. Under the assumption that all ConEx marks will not get invalid for the whole Slow Start phase, marks of a previous RTT have to be added up. Thus the marking of every fourth packet will allow sufficient credits in Slow Start as it can be seen in Figure Figure 1.

```
             RTT1  |------XC------>|
                   |------X------->|
                   |------X------->|   credit=1  in_flight=3
                   |               |
             RTT2  |------X------->|
                   |------XC------>|
                   |------X------->|
                   |------X------->|
                   |------X------->|
                   |------XC------>|   credit=3  in_flight=6
                   |               |
             RTT3  |------X------->|
                   |------X------->|
                   |------X------->|
                   |------XC------>|
                   |------X------->|
                   |------X------->|
                   |------X------->|
                   |------XC------>|
                   |------X------->|
                   |------X------->|
                   |------X------->|
                   |------XC------>|   credit=6  in_flight=12
                   |      .        |
                   |      :        |
```

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

Moreover, a ConEx sender should maintain a counter of the sent credits c. In Congestion Avoidance phase, the sender should to monitor the number of packets in flight f. If f every gets larger than c, the ConEx sender should send new credits.

The audit might loose state due to e.g. rerouting or memory limitation.  Therefore, the sender needs to detect this case and resend credits.  Thus a ConEx sender should reset the credit count c if losses occur in two subsequent RTTs (assuming that the sending rate was correctly reduced based on the received congestion signal).

### 4.3. Loss of ConEx information

The audit can have wrong information if e.g. ConEx got lost on the channel (or a wrong number of ConEx marking has been estimated by the sender due to a lack of feedback information).  In this case the audit might penalize a sender wrongly.  The ConEx sender should detect this case and send further credits which should solve the situation (see Section 4.2).

## 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 is not 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. In cases where it is preferable to slightly 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. As ConEx-capable packet can carry different ConEx marks at the same time, these information do not need to be distributed over several packets and thus can be sent without further delay.

## 6. Acknowledgements

The authors would like to thank Bob Briscoe who contributed with this initial ideas and valuable feedback. Moreover, thanks to Jana Iyengar who provided valuable feedback.

## 7. IANA Considerations

This document does not have any requests to IANA.

## 8. Security Considerations

With some of the advanced ECN compatibility modes it is possible to miss congestion notifications. Thus a sender will not decrease its sending rate. If the congestion is persistent, the likelihood to receive a congestion notification increases. In the worst case the sender will still react correctly to loss. This will prevent a congestion collapse.

## 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.

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

[draft-ietf-conex-abstract-mech] Mathis, M. and B. Briscoe, "Congestion Exposure (ConEx) Concepts and Abstract Mechanism", draft-ietf-conex-abstract-mech-06 (work in progress), October 2012.

[draft-ietf-conex-destopt] Krishnan, S., Kuehlewind, M., and C. Ucendo, "IPv6 Destination Option for ConEx", draft-ietf-conex-destopt-04 (work in progress), March 2013.

### 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.

[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.

[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.

[RFC6789] Briscoe, B., Woundy, R., and A. Cooper, "Congestion Exposure (ConEx) Concepts and Use Cases", RFC 6789, December 2012.

[draft-briscoe-tsvwg-byte-pkt-mark] Briscoe, B. and J. Manner, "Byte and Packet Congestion Notification", draft-briscoe-tsvwg-byte-pkt-mark-010 (work in progress), May 2013.

[draft-kuehlewind-tcpm-accurate-ecn] Kuehlewind, M. and R. Scheffenegger, "More Accurate ECN Feedback in TCP", draft-kuehlewind-tcpm-accurate-ecn-02 (work in progress), Jun 2013.

## Appendix A. Revision history

RFC Editior: This section is to be removed before RFC publication.

00 ... initial draft, early submission to meet deadline.

01 ... refined draft, updated LEG "drain" from per-packet to RTT-based.

02 ... added Section 4.3 and expanded discussion about ECN interaction.

03 ... expanded the discussion around credit bits.

04 ... review comments of Jana addressed. (Change in full compliance mode.)

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