Congestion Exposure (ConEx)
Internet-Draft
Intended status: Experimental
Expires: December 01, 2013

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

May 30, 2013

TCP modifications for Congestion Exposure
draft-ietf-conex-tcp-modifications-03

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

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. This congestion information is retrieved by the sender based on existing feedback mechanisms from the receiver to the sender in TCP.

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) from the receiver to the sender. 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 experienced congestion. ConEx also works with classic ECN but it is less accurate when multiple congestion markings occur within on RTT.

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 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. 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 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 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 bytes nor packets. As network congestion is usually byte-congestion, 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 TCP sender SHOULD 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 bytes of an IP packet must be accounted. Therefore the sender SHOULD take the headers into account. If equal sized packets, or at least equally distributed packet sizes can be assumed, 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.

If a sender sends different sized packets with unequally distributed packet sizes, the sender might be able to reconstruct the exact number of headers based on information which packet sizes has been sent in the last RTT. 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).

3.1. ECN

ECN 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 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 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 have 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.

In case of duplicate acknowledgements the number of acknowledged bytes will be zero even though (CE marked) data has been received. Therefore, we calculated a variable DeliveredData. DeliveredData covers the total number of bytes that the current ACK indicates have been delivered to the receiver, relative to all past ACKs. With SACK, DeliveredData is increased by the number of bytes given by changes in the SACK information. Note the change in in the SACK information can also be negative if the number of acknowledged bytes increases. Without SACK, DeliveredData is estimated to be 1 SMSS on duplicate acknowledgements, and on a subsequent partial or full ACK, DeliveredData is estimated to be the change in acknowledged bytes, minus one SMSS for each preceding duplicate ACK.

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

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 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.

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. On average the sender will send sufficient ConEx marks due to the scheme proposed above but these ConEx marks might be shifted in time. 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 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, 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.

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 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. 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).

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 shuould 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 summed 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 needs 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.

6. Acknowledgements

The authors would like to thank Bob Briscoe who contributed with this initial ideas and 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.

[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.

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.

[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.

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.

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