Workgroup: Network Working Group
Internet-Draft: draft-stewart-tsvwg-sctpecn-06
Published: 20 October 2023
Intended Status: Informational
Expires: 22 April 2024
Authors: R. Stewart
Netflix, Inc.
M. Tüxen
Münster Univ. of Appl. Sciences

Explicit Congestion Notification for the Stream Control Transmission Protocol

Abstract

This document describes the addition of the Explicit Congestion Notification (ECN) to the Stream Control Transmission Protocol (SCTP).

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 https://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 22 April 2024.

Copyright Notice

Table of Contents

1. Introduction

At the time SCTP was initially defined in [RFC2960], ECN as specified in [RFC2481] was still an experimental document. This left the authors of SCTP in a position where they could not directly refer to ECN without creating a normative reference in a standards track document to an experimental RFC. To work around this problem the authors of SCTP decided to add two reserved chunk types for ECN (CWR and ECNE) but did not fully specify how they were to be used except in a vague way within an appendix of the document. This worked around the document reference problem, but left ECN and its implementation for SCTP unspecified. This document is intended to fill in the details of ECN processing in SCTP in a standards track document.

This document assumes that the reader is familiar with ECN [RFC3168]. Readers unfamiliar with ECN are strongly encouraged to first read [RFC3168] since this document will not repeat any of the details on how the various IP level bits are set. This document will use the same terminology has [RFC3168]. For example the term ECT is used to indicate that the IP level packet is marked indicating the transport (SCTP) supports ECN.

2. Conventions

The key words "**MUST**", "**MUST NOT**", "**REQUIRED**", "**SHALL**", "**SHALL NOT**", "**SHOULD**", "**SHOULD NOT**", "**RECOMMENDED**", "**NOT RECOMMENDED**", "**MAY**", and "**OPTIONAL**" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

3. Terminology

All integer fields defined in this document included in an SCTP packet **MUST** be transmitted in network byte order, unless otherwise stated.

ECT: The term used to indicate that the IP level packet is marked indicating the transport is willing to support ECN for this packet.

not-ECT: The term used to indicate that the IP level packet is marked indicating the transport is NOT willing to support ECN for this packet.

CE: The term used to indicate that the IP level packet is marked indicating that a router in the network has marked the packet as having experienced congestion.

4. Chunk and Parameter Formats

4.1. ECN Support Parameter (32768)

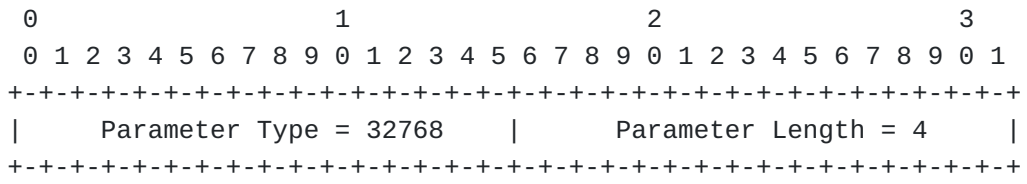

Figure 1: ECN Support Chunk Parameter

Type: 16 bits (unsigned integer)
This field holds the IANA defined parameter type for the "ECN Support" chunk parameter. IANA is requested to assign the value 32768 (0x8000) (suggested) for this parameter type.

Length: 16 bits (unsigned integer)
This field holds the length in bytes of the chunk parameter; the value **MUST** be 4.

The ECN Support Chunk Parameter **MAY** appear in INIT and INIT ACK chunks and **MUST NOT** appear in any other chunk.

4.2. ECN Echo (12)

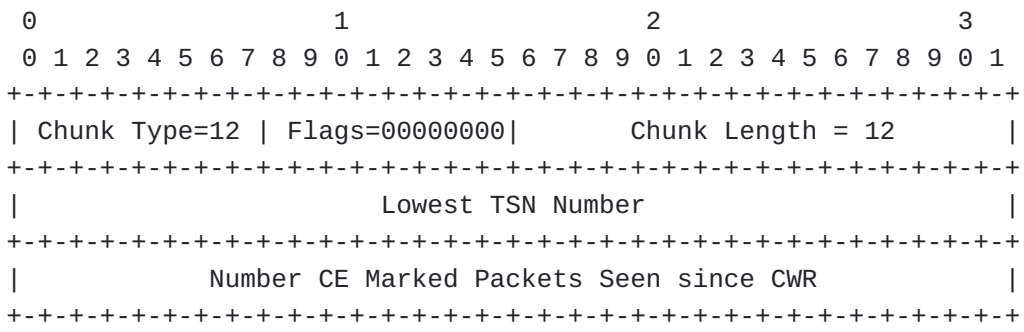

Figure 2: ECN Echo Chunk

Flags: 8 bits
Set to all zeros on transmit and ignored on receipt.

Length: 16 bits (unsigned integer)
This field holds the length in bytes of the chunk; the value **MUST** be 12.

Lowest TSN Number: 32 bits (unsigned integer)
This parameter contains the lowest TSN number contained in the last packet received that was marked by the network with a CE indication.

Number CE Marked Packets: 32 bits (unsigned integer)
This parameter contains the total number of CE marked packets that has been seen since the first CE mark received while waiting for a CWR chunk. Note that the CE counter will overflow from 0xffffffff to 0 if a CWR chunk is not recieved.

Note that the appendix of [RFC4960] did not have the field Number CE Marked Packets. Implementations **SHOULD** accept an 8 byte form of this chunk that does not include this field. In such a case the implementation **SHOULD** treat the missing field as indicating one CE marked packet for any purpose for which the implementation is using this field.

4.3. CWR Chunk(13)

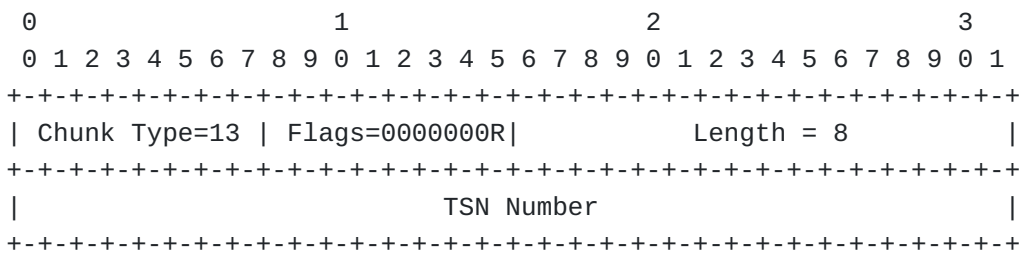

Figure 3: CWR Chunk

Flags: 8 bits

The R Bit indicates if the CWR is a retransmission of an earlier CWR that may have been lost. If this bit is set, then the TSN number included is the latest TSN that a CWR has been responded to. If the R bit is clear, than the TSN indicated is the latest TSN for that destination.

Length: 16 bits (unsigned integer)

This field holds the length in bytes of the chunk; the value **MUST** be 8.

TSN Number: 32 bits (unsigned integer)

This parameter contains the TSN number to which the sender has reduced his congestion window to.

5. Procedures

5.1. SCTP Initialization

In the SCTP association setup phase, the source and destination SCTP endpoints exchange information about their willingness to use ECN. After the completion of this negotiation, an SCTP sender sets an ECT codepoint in the IP header of data packets to indicate to the network that the transport is capable and willing to participate in ECN for this packet. This indicates to the routers that they may mark this packet with the CE codepoint.

If the SCTP association does not wish to use ECN notification for a particular packet, the sending SCTP sets the ECN codepoint to not-ECT, and the SCTP receiver ignores the CE codepoint in the received packet.

For this discussion we will call the endpoint initiating the SCTP association as EP-A and the listening SCTP endpoint as EP-Z.

Before an SCTP association can use ECN, EP-A sends an INIT chunk which includes the ECN Support parameter. By including the ECN Support parameter the sending endpoint (EP-A) will participate in ECN as both a sender and a receiver. Specifically, as a receiver, it will respond to incoming data packets that have the CE codepoint set in the IP header by sending an ECN Echo chunk bundled with the next outgoing SACK Chunk. As a sender, it will respond to incoming packets that include an ECN Echo chunk by reducing the congestion window and sending a CWR chunk when appropriate.

Including an ECN Support parameter in an INIT or INIT-ACK does not commit the SCTP sender to setting the ECT codepoint in any or all of the packets it may transmit. However, the commitment to respond appropriately to incoming packets with the CE codepoint set remains.

When EP-Z sends INIT-ACK chunk, it also includes an ECN Support parameter. Including the ECN Support parameter indicates that the SCTP transmitting the INIT-ACK chunk is ECN-Capable.

The following rules apply to the use of ECN for an SCTP association.

- *If the SCTP Endpoint supports ECN a sender of either an INIT or INIT-ACK chunk **MUST** always include the ECN Supported Parameter.

- *After the exchange of the INIT and INIT-ACK if both endpoints have not indicated support of ECN by including an ECN Supported Parameter, then ECT **MUST NOT** be set on any IP packets sent by any endpoint which is ECN capable. Furthermore upon receiving IP packets with a CE codepoint set, the ECN capable endpoint **SHOULD** ignore the CE codepoint.

- *If both endpoints have included an ECN Supported Parameter in the INIT and INIT-ACK exchange, then both endpoints **MUST** follow the ECN procedures defined in the rest of this document.

- *A sending endpoint **SHOULD** set the ECT code points on IP packets that carry DATA chunk. This includes IP packets that have other control chunks bundled with the Data.

5.2. The SCTP Sender

For an SCTP association using ECN, new data packets are transmitted with an ECT codepoint set in the IP header. When only one ECT codepoint is needed by a sender for all packets sent on an SCTP association ECT(0) **SHOULD** be used. If the sender receives an ECN-Echo chunk packet, then the sender knows that congestion was encountered in the network on the path from the sender to the receiver. The indication of congestion should be treated just as a congestion loss in non-ECN-Capable SCTP. That is, the SCTP source halves the congestion window "cwnd" for the destination address that the sender transmitted the data to and reduces the slow start threshold "ssthresh". A packet containing an ECN-Echo chunk shouldn't trigger new data to be sent. SCTP follows the normal procedures for increasing the congestion window when it receives a packet with a SACK chunk without the ECN Echo chunk.

SCTP should not react to congestion indications more than once every round-trip time. That is, the SCTP sender's congestion window should be reduced only once in response to a series of dropped and/or CE packets from a single window of data. In addition, the SCTP source should not decrease the slow-start threshold, ssthresh, if it has been decreased within the last round trip time.

One method to accomplish this is as following:

1. During association setup, create a new state variable ECN_ECHO_TSN and ECN_ECHO_LAST for each destination. The initial value of these variables are set to the initial TSN that will be assigned minus 1.

2. When an ECN Echo chunk arrives, use the TSN in the ECN Echo to establish which destination the packet was sent to. We will call this destination the selected destination. If the chunk cannot be found note that an override is occurring from the selected destination (if found) select its ECN Echo TSN.

3. Compare the ECN Echo TSN with the ECN_ECHO_TSN for the selected destination. If an override is not noted and the value of the ECN_ECHO_TSN is greater than the ECN Echo TSN proceed to step 4; else proceed to step 6.

4. Reduce the cwnd and ssthresh for the selected destination the same as if a loss was detected during a fast retransmit. For details, see [RFC9260] Section 7.2.3 and Section 7.2.4.

5. Record in the ECN_ECHO_TSN value, the last TSN that was sent and recorded in ECN_ECHO_LAST the TSN number from the ECN Echo Chunk.

6. If the implementation is tracking the number of marked packets, record the value found in the 'Number CE Marked Packets Seen since CWR' field and also add this number to the running loss count. If such a count is not being maintained, then proceed to step 8.

7. If the implementation is tracking the number of marked packets, compare the number in the ECN Echo Chunk TSN to the ECN_ECHO_LAST. If it is greater than ECN_ECHO_LAST, update ECN_ECHO_LAST with this value. Take the difference between the stored 'Number CE Marked Packets' field and the value from the newly arriving 'Number CE Marked Packets' and add this difference to the total loss count. Then update the stored 'Number CE Marked Packets' with the ECN Echo Chunk TSN.

8. Create a CWR chunk with the value found in the ECN_ECHO_LAST for the selected destination. If an override was noted, set the 'O' bit within the CWR flags. Queue this chunk for transmission to the peer destination. Note if there is already such a chunk in queue to be sent, remove that chunk and replace it with the new chunk.

After the sending SCTP reduces its congestion window in response to a ECN Echo, incoming SACKs that continue to arrive can "clock out"

outgoing packets as allowed by the reduced congestion window. Note that continued arrival of ECN Echo chunks should still be processed as described above, possibly reducing the cwnd, but always sending a CWR to the receiving SCTP. This assures that the ECN Echo and CWR are robust with regard to loss in either direction and that the implementation, if it desires, can maintain an accurate loss count per destination.

Note, originally in the appendix of [RFC4960] a definition was supplied for the ECN Echo chunk. This definition did not include the 'Number CE Marked Packets' field. An implementation **SHOULD** accept such a chunk, delineating it from the standards track version by the fact that the length field will be 8 bytes instead of 12. When processing this older style chunk, the 'Number CE Marked Packets' **SHOULD** be treated as if it contains the number 1. This may cause incorrect loss counts but will not cause any issues with SCTP's ECN handling.

5.3. The SCTP Receiver

When an SCTP endpoint first receives a CE data packet at the destination end-system, the SCTP data receiver creates an ECN Echo chunk and records the lowest TSN number found in the data packet. It also sets the 'Number CE Marked Packets' to 1 and queues this chunk for transmission at the next opportunity. If there is any ACK withholding implemented, as in current "delayed-SACK" SCTP implementations where the SCTP receiver can send an SACK for two arriving data packets, then the ECN Echo chunk will not be sent until the SACK is sent. If the next arriving data packet also has the CE codepoint set, then the receiver updates the queued ECN Echo chunk to have a higher TSN value (the lowest one in the newly arriving data packet) and increments the 'Number CE Marked Packets' field in the queued chunk.

Multi-homing requires one added restriction upon the ECN Echo chunk, such a chunk **MUST** be bundled with a SACK, and the SACK **MUST** follow the ECN Echo Chunk. This ordering is necessary so that the receiver of the ECN Echo chunk will at least one time find the proper destination to which the chunk was originally sent. Without this restriction it is possible a SACK could arrive ahead of the ECN Echo Chunk, no matter what the sending order, causing the sender to free the DATA chunk and thus loose the association with what destination it was sent to. For the same reason we also require the ECN Echo Chunk be earlier in the packet ahead of the SACK so that the SACK is not processed before the ECN Echo Chunk.

After transmission of the ECN Echo chunk, usually bundled with the SACK, the receiver does not discard the ECN Echo chunk. Instead it keeps the chunk in its queue and continues to send this chunk

bundled with at least a SACK chunk on each outgoing packet, updating it as described above if other CE codepoint data packets arrive. The ECN Echo chunk should only be discarded when a CWR Chunk arrives holding a TSN value that is greater than or equal to the value inside the ECN Echo Chunk.

This provides robustness against the possibility of a dropped SACK packet carrying an ECN Echo chunk. The SCTP receiver continues to transmit the ECN Echo chunk in subsequent SACK packets until the correct CWR is received.

After the receipt of the CWR chunk, acknowledgments for subsequent non-CE data packets will not have an ECN Echo chunk bundled with them. If another CE packet is received by the data receiver, the receiver would once again send SACK packets bundled with a newly created ECN Echo chunk. The receipt of a CWR packet guarantees that the data sender has received the ECN Echo chunk for the TSN specified, and reduced its congestion window at some point after it sent the data packet for which the CE codepoint was set.

When processing a CWR, it is important that the receiver of the CWR validate the source address from which the CWR came from. It **SHOULD** match the destination the ECN Echo was sent to unless the override bit is set in the CWR Chunk.

5.4. Congestion on the SACK Path

For the current generation of SCTP congestion control algorithms, pure acknowledgement packets (e.g., packets that do not contain any accompanying data) **MUST** be sent with the not-ECT codepoint. Current SCTP receivers have no mechanisms for reducing traffic on the SACK-path in response to congestion notification. Mechanisms for responding to congestion on the SACK-path are areas for current and future research. For current SCTP implementations, a single dropped SACK generally has only a very small effect on SCTP's sending rate.

5.5. Retransmitted SCTP Packets

This document specifies ECN-capable SCTP implementations **MUST NOT** set either ECT codepoint (ECT(0) or ECT(1)) in the IP header for retransmitted data packets, and that the SCTP data receiver **SHOULD** ignore the ECN field on arriving data packets that are outside of the receiver's current window. The reasons for this can be found in [RFC3168] Section 6.1.5.

5.6. SCTP Window Probes

When the SCTP data receiver advertises a zero window, the SCTP data sender sends window probes to determine if the receiver's window has increased. Window probe packets for SCTP do contain user data (one

chunk). If a window probe packet is dropped in the network, this loss can be detected by the receiver. Therefore, the SCTP data sender **MAY** set an ECT codepoint on the initial send of the window probe, but the SCTP sender **MUST NOT** set the ECT codepoint on retransmissions of that TSN.

6. Socket API Considerations

This section describes how the socket API defined in [RFC6458] needs to be extended to support ECN as defined in this document.

Please note that this section is informational only.

7. IANA Considerations

TBD.

8. Security Considerations

[RFC3168] defines the security considerations for ECN. These same consideration that are described for TCP are applicable to SCTP.

9. References

9.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC3168] Ramakrishnan, K., Floyd, S., and D. Black, "The Addition of Explicit Congestion Notification (ECN) to IP", RFC 3168, DOI 10.17487/RFC3168, September 2001, <https://www.rfc-editor.org/info/rfc3168>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[RFC9260] Stewart, R., Tüxen, M., and K. Nielsen, "Stream Control Transmission Protocol", RFC 9260, DOI 10.17487/RFC9260, June 2022, <https://www.rfc-editor.org/info/rfc9260>.

9.2. Informative References

[RFC2481] Ramakrishnan, K. and S. Floyd, "A Proposal to add Explicit Congestion Notification (ECN) to IP", RFC 2481, DOI 10.17487/RFC2481, January 1999, <https://www.rfc-editor.org/info/rfc2481>.

[RFC2960] Stewart, R., Xie, Q., Morneault, K., Sharp, C., Schwarzbauer, H., Taylor, T., Rytina, I., Kalla, M., Zhang, L., and V. Paxson, "Stream Control Transmission Protocol", RFC 2960, DOI 10.17487/RFC2960, October 2000, <https://www.rfc-editor.org/info/rfc2960>.

[RFC4960] Stewart, R., Ed., "Stream Control Transmission Protocol", RFC 4960, DOI 10.17487/RFC4960, September 2007, <https://www.rfc-editor.org/info/rfc4960>.

[RFC6458] Stewart, R., Tuexen, M., Poon, K., Lei, P., and V. Yasevich, "Sockets API Extensions for the Stream Control Transmission Protocol (SCTP)", RFC 6458, DOI 10.17487/RFC6458, December 2011, <https://www.rfc-editor.org/info/rfc6458>.

Acknowledgments

Special thanks to Xuesong Dong for being a coauthor on early versions of the document.

Thanks to Richard Scheffenegger for his helpful comments and review.

Authors' Addresses

Randall R. Stewart
Netflix, Inc.
15214 Pendio Drive
Bella Collina, FL 34756
United States of America

Email: randall@lakerest.net

Michael Tüxen
Münster University of Applied Sciences
Stegerwaldstrasse 39
48565 Steinfurt
Germany

Email: tuexen@fh-muenster.de