Network Working Group
INTERNET-DRAFT

Q. Xie
Motorola
R. R. Stewart
C. Sharp
Cisco
I. Rytina
Ericsson

Expires in six months

February 2001

SCTP Unreliable Data Mode Extension
<draft-ietf-sigtran-usctp-01.txt>

Status of This Memo

This document is an Internet-Draft and is in full conformance with all provisions of Section 10 of RFC 2026 [RFC2026]. Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html.

Abstract

This document describes an extension to the Stream Control Transmission Protocol (SCTP) [RFC2960] to provide unreliable data transfer services. The benefits of this extension includes unified congestion control over reliable and unreliable data streams, single association for multi-content data services, link level fault tolerance for unreliable data transfer, unreliable data stream multiplexing, etc.

The data transfer service extension will also support partial payload checksum in order to facilitate bit-error tolerance media applications.

TABLE OF CONTENTS

1. Introduction..... 2
2. Conventions..... 3
3. Unreliable Data and Partial Checksum Design..... 3
  3.1 New INIT and INIT-ACK parameters..... 3
    3.1.1 Unreliable Streams Parameter Definition..... 3
    3.1.2 Partial Checksum Parameter Definition..... 5

3.2 New chunk definitions..... 5
3.2.1 Forward Cumulative TSN Chunk (FORWARD TSN)..... 5
3.2.2 Partial Checksum Data Chunk (P-DATA)..... 6

4. Unreliable Stream Operations..... 8
4.1 Initialization of Unreliable Streams..... 8
4.2 Send Unreliable Data..... 9
4.3 Receive Unreliable Data.....11
4.4 Other Issues on Unreliable Data.....11
4.4.1 Unreliable Data Stream Multiplexing.....11
4.4.2 Fault Tolerant Unreliable Data Transfer.....12
4.4.3 Unreliable Data Out-of-order Detection.....12
5. Usage of the P-DATA chunk.....12
6. Acknowledgments.....13
7. Authors' Addresses.....13
8. References.....14

1. Introduction

Taking advantage of the extensibility of SCTP, this document adds unreliable data transfer services to SCTP and an optional method to send SCTP Data Chunks with limited checksum coverage. The design presented here allows the co-existence of unreliable data streams and reliable streams in a single SCTP association.

The following are some of the advantages for integrating unreliable and partial checksum data services into SCTP:

1) Unreliable extension to SCTP (U-SCTP) supports congestion control and congestion avoidance over unreliable data traffic; this is very desirable since it is much friendlier towards the network than UDP.

2) Some applications services can greatly benefit from U-SCTP by using a single SCTP association to carry both reliable content (e.g., text, billing, accounting, set-up information, etc.) and unreliable content (e.g., Fiber channel, SCSI over IP, etc.).

3) U-SCTP allows the use of a unified congestion control across both reliable and unreliable traffic between two endpoints. This has the potential for better utilization of network resources, achieving similar objectives of the Endpoint Congestion Management (ecm) Working Group.

4) Taking advantage of SCTP data chunk bundling function, sending multiple unreliable data streams across a single SCTP association creates a very efficient and effective way of data multiplexing.

5) U-SCTP gives even the unreliable data traffic "link-level" fault tolerance, taking advantage of SCTP multi-homing ability. This is not possible with UDP.

6) U-SCTP can achieve either ordered or unordered unreliable data

transfer, while UDP is incapable of controlling the order of data delivery.

7) An application can control its retransmission policies if

retransmission is deemed needed.

8) Some applications may find it desirable to limit the coverage of the Adler32 checksum over the actual data chunks.

2. Conventions

The keywords MUST, MUST NOT, REQUIRED, SHALL, SHALL NOT, SHOULD, SHOULD NOT, RECOMMENDED, NOT RECOMMENDED, MAY, and OPTIONAL, when they appear in this document, are to be interpreted as described in RFC 2119 [RFC2119].

3. Unreliable Data and Partial Checksum Design

With the unreliable data extension, an SCTP data sender will be allowed to designate a sub-set of its outbound streams to be unreliable streams. The user data chunks sent to an unreliable stream will share the same TSN space, the same congestion control/avoidance treatment, and the same transmission priority as those sent to a reliable stream, but they will not be retransmitted (or only be retransmitted for a limited times) if they are found missing at the data receiver.

The partial checksum extension allows the sending application to specify a portion of an outbound user message to be excluded from SCTP checksum protection.

3.1 New INIT and INIT-ACK parameters

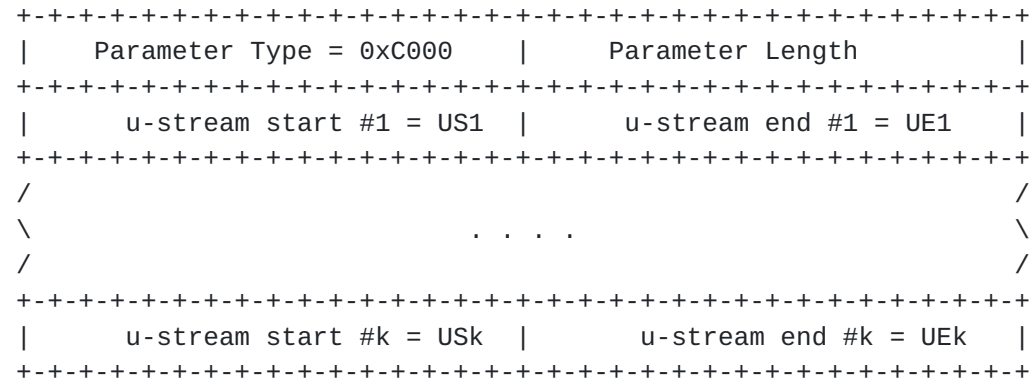
The following new optional parameter, are added to the INIT and INIT ACK chunks. At the initialization of the association, the sender of the INIT or INIT ACK chunk may include these parameters to indicate its ability to support these features.

Parameter Name	Status	Type Value
Unreliable Streams	Optional	0xC000
Partial Checksum Support	Optional	0xC004

3.1.1 Unreliable Streams Parameter Definition

The Unreliable Streams parameter is added to the INIT and INIT ACK chunks. At the initialization of the association, the sender of the INIT or INIT ACK chunk shall include this optional parameter to inform its peer that it is able to support unreliable streams and to designate its unreliable outbound streams. If no streams are marked as unreliable but the sender does support the unreliable streams option the sender SHOULD include a parameter

with no u-stream ranges and a fixed Parameter Length of 4.



Type: 16 bit u_int

0xC000, indicating Unreliable Streams parameter

Length: 16 bit u_int

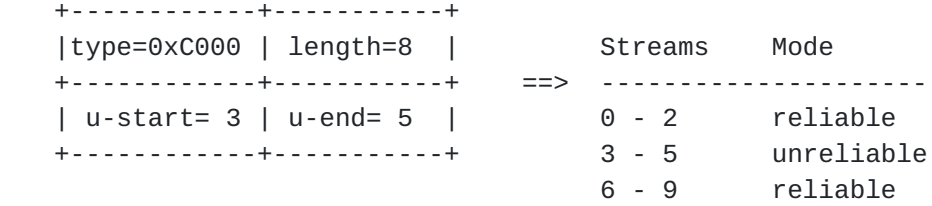
Indicate the size of the parameter in octets, including the Type, Length, u-stream start, and u-stream end fields.

u-stream start: 16 bit u_int, and
u-stream end: 16 bit u_int

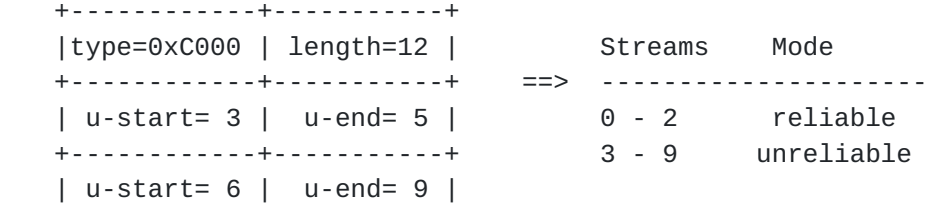
Each pair of u-stream start and u-stream end fields defines one or more unreliable outbound streams, starting from stream number US and ending with stream number UE. The union of all the pairs together defines the complete sub-set of all unreliable outbound streams.

The following are some examples of unreliable stream designation (assuming OS = 10):

Example 1: (assuming OS = 10)



Example 2: (assuming OS = 10)



Example 3: (assuming OS = 10)

```
      +------------+-----------+
      |type=0xC000 | length=12 |         Streams    Mode
      +------------+-----------+    ==>  ---------------------
      | u-start= 9 |  u-end= 9 |         0          unreliable
      +------------+-----------+         1 - 8      reliable
      | u-start= 0 |  u-end= 0 |         9          unreliable
      +------------+-----------+
```

Example 4: (assuming OS = 10)

```
      +------------+-----------+
      |type=0xC000 | length=8  |         Streams    Mode
      +------------+-----------+    ==>  ---------------------
      | u-start= 0 |  u-end= 9 |         0 - 9      unreliable
      +------------+-----------+
```

Example 5: (assuming OS = 10)

```
      +------------+-----------+
      |type=0xC000 | length=4  |         Streams    Mode
      +------------+-----------+    ==>  ---------------------
                                         0 - 9      reliable
```

3.1.2 Partial Checksum Parameter Definition

The Partial Checksum Parameter is added to the INIT and INIT ACK chunks. At the initialization of the association, the sender of the INIT or INIT ACK chunk shall include this optional parameter to inform its peer that it is able to support the partial checksum P-DATA (Chunk Type = 0xC3 or 195), see section 3.2.2.

```
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|    Parameter Type = 0xc004    |     Parameter Length=4        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

3.2 New chunk definitions

The following new chunk types are added to support the two new features in SCTP.

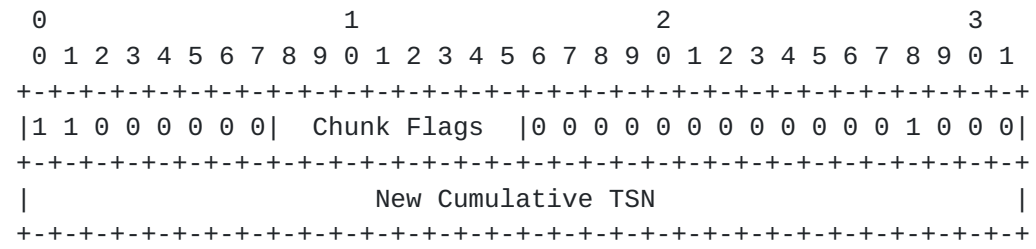
Chunk Type	Chunk Name
11000000	Forward Cumulative TSN (FORWARD TSN)
11000011	Partial Checksum Data Chunk (P-DATA)

The FORWARD TSN supports the unreliable stream. The Partial Checksum Data Chunk supports data chunks that are not completely covered by the

Adler32 checksum in the SCTP packet header.

3.2.1 Forward Cumulative TSN Chunk Definition (FORWARD TSN)

Forward Cumulative TSN chunk shall be used by the data sender to inform the data receiver to adjust its cumulative received TSN point forward because some missing TSNs are associated with unreliable data chunks and will no longer be retransmitted by the sender.



Chunk Flags:

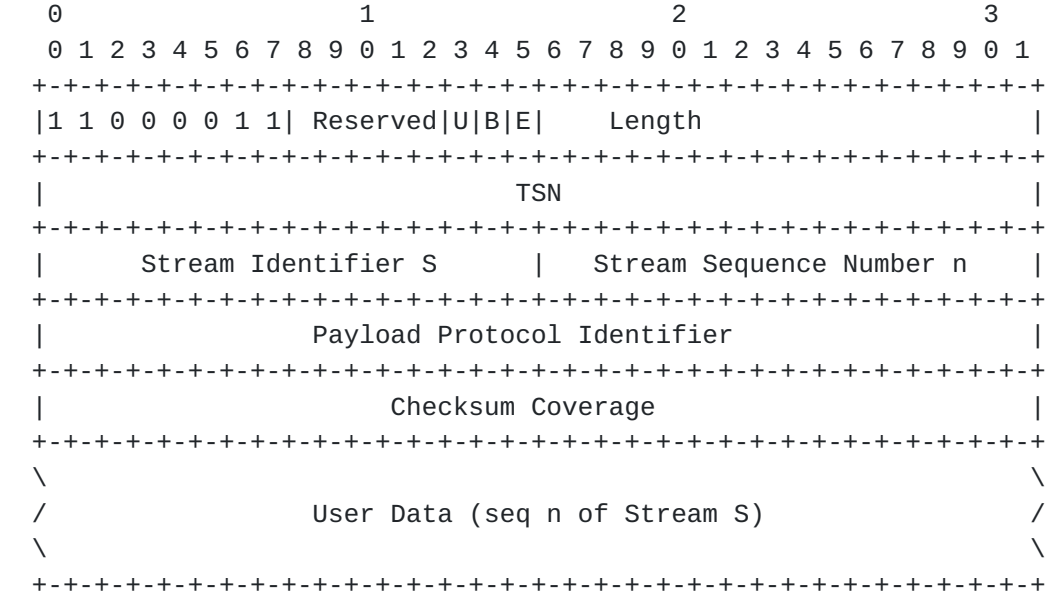
Set to all zeros on transmit and ignored on receipt.

New Cumulative TSN: 32 bit u_int

This indicates the new cumulative TSN to the data receiver. Upon the reception of this value, the data receiver shall consider any missing TSNs earlier than this value received and stop reporting them as gaps in the subsequent SACKs.

3.2.2 Partial Checksum Data Chunk (P-DATA)

The following format MUST be used for the P-DATA chunk:



Reserved: 5 bits

Should be set to all '0's and ignored by the receiver.

U bit: 1 bit

Same as in a DATA chunk [RFC2960], the (U)nordered bit, if set to '1', indicates that this is an unordered P-DATA chunk, and there is no Stream Sequence Number assigned to this P-DATA chunk. Therefore, the receiver MUST ignore the Stream Sequence Number field.

Unordered P-DATA chunks MUST be dispatched to the upper layer by the receiver without any attempt to re-order them.

B bit: 1 bit

MUST be set to 1 by the sender.

E bit: 1 bit

MUST be set to 1 by the sender.

Length: 16 bits (unsigned integer)

This field indicates the length of the P-DATA chunk in bytes from the beginning of the type field to the end of the user data field excluding any padding. For example, a P-DATA chunk with 5 bytes of user data field will have Length set to 25 (indicating 25 bytes).

TSN: 32 bits (unsigned integer)

This value represents the TSN for this P-DATA chunk. The valid range of TSN is from 0 to 4294967295 (2**32 - 1). TSN wraps back to 0 after reaching 4294967295.

Stream Identifier S: 16 bits (unsigned integer)

Identifies the stream to which the following the user data belongs.

Stream Sequence Number n: 16 bits (unsigned integer)

This value represents the stream sequence number of the following user data within the stream S. Valid range is 0 to 65535.

Payload Protocol Identifier: 32 bits (unsigned integer)

Same as in a DATA chunk [RFC2960], this value represents an application (or upper layer) specified protocol identifier. This value is passed to SCTP by its upper layer and sent to its peer. This identifier is not used by SCTP but can be used by certain network entities as well as the peer application to identify the type of information being carried in this P-DATA chunk.

The value 0 indicates no application identifier is specified by the upper layer for this payload data.

Checksum Coverage: 32 bits (unsigned integer)

This field contains an integer that indicates how much User Data (in octets) of this P-DATA chunk is covered by the Adler32 checksum in

the SCTP packet common header.

Note, all chunk headers, as well as the common header of the SCTP packet are always covered by the packet checksum, regardless of the value of the Checksum Coverage.

The coverage area is defined as starting from the first transmitted byte in the User Data part of the Chunk for exactly 'Checksum Coverage' bytes in length.

For example, if a value of "5" appears in the Checksum Coverage field, only the first 5 bytes of the User Data are included in the calculation of the packet checksum. A value of "0" in the Checksum Coverage field will indicate that all the User Data in this P-DATA chunk is excluded from the packet checksum.

The sender of a P-DATA chunk SHOULD NOT set a value of Checksum Coverage larger than the length of the User Data in the chunk. In other words, the Checksum Coverage SHOULD be set smaller than or equal to the chunk Length - 20.

On the receiver side, if the value of Checksum Coverage in a received P-DATA chunk is found larger than the length of the User Data in the chunk, the receiver MUST accept the chunk and MUST consider the entire User Data in the chunk covered by the Adler-32 checksum.

User Data: variable length

This is the payload user data. The implementation MUST pad the end of the data to a 4 byte boundary with all-zero bytes. Any padding MUST NOT be included in the Length field. A sender MUST never add more than 3 bytes of padding.

4. Unreliable Stream Operations

In this section, we first defines the procedures for opening unreliable streams in an SCTP association. Then, we will discuss procedures for sending and receiving unreliable SCTP data chunks.

4.1 Initialization of Unreliable Streams

If the SCTP data sender plans to send unreliable data, at the initialization of the association it MUST include the Unreliable Streams parameter in its INIT or INIT ACK chunk to indicate to its peer which of its outbound streams are going to be used as unreliable streams.

Upon the reception of the Unreliable Streams parameter, the data receiver SHALL determine and record the mode (reliable or unreliable) of each inbound stream, as it allocates resource for its inbound streams.

Note, if the data receiver does not support unreliable inbound streams, it SHOULD treat the Unreliable Streams parameter as an invalid or unrecognized parameter and respond to the data sender with an operational error, following the rules defined in Section 5.1 of [RFC2960].

Upon reception of the operational error indicating that its peer does not support unreliable streams, the data sender may choose to either:

1) end the initiation process, in consideration of the peer's inability of meeting the requested features for the new association, or
2) continue the initiation process, but with the understanding that ALL its outbound streams will be reliable.

In either case, the data sender SHOULD inform its upper layer its peer's inability of supporting unreliable data transfer.

Initiation of streams as reliable and/or unreliable may be under the control of the SCTP user. Hence, the ULP primitive "ASSOCIATE" (see Section 10.1 of [RFC2960]) should be expanded to contain the optional U-stream-start and U-stream-end values.

4.2 Send Unreliable Data

During the lifetime of the association, any user data sent to an unreliable stream will be treated as unreliable user data and will automatically be transmitted in unreliable mode.

Note, fragmentation MUST NOT be performed on an unreliable user message. In other words, an unreliable user message MUST be sent in a single DATA or P-DATA chunk regardless of its size.

The SCTP data sender shall handle user data sent to an unreliable stream the same way as it handles user data sent to a reliable stream (i.e., the same timer rules, congestion control rules, failure detection rules, RTO control rules, etc.), with the following exceptions:

A1) The sender maintains a "Peer.Cumulative.TSN" for each peer so as to track the latest cumulative TSN point of the peer (Note, this variable may already exist in a classic SCTP implementation for outqueue management and for detecting out-of-order SACKs).

A2) Before retransmitting a DATA chunk (due to either a T3-rtx timer expiration as defined in 6.3.3 of [RFC2960] or a 4th missing indication as defined in 7.2.4 of [RFC2960]), the SCTP data sender MUST check whether the DATA chunk is being transmitted on

an unreliable stream. If so, it will perform the following:

B1) Check the value of the unreliable retransmission counter "Unrel.Trans.Count" value for the DATA chunk. This value may

be set by the SCTP user to 0 (no retransmission) for complete unreliability, or N (where N >0) for limited reliability at the time when the user message is passed to SCTP.

B2) If the "Unrel.Trans.Count" of the chunk is currently greater than 0, the sender MUST retransmit the data chunk and then decrease the "Unrel.Trans.Count" by 1. The same rules for retransmission as defined in [RFC2960] SHALL be used for RTO calculation, destination selection, error reporting, etc.

B3) If the "Unrel.Trans.Count" is currently 0, the sender MUST NOT retransmit the data chunk. Instead, the sender MUST mark the data chunk as being finally acked.

A3) whenever the data sender receives a SACK from the data receiver, it SHALL first process the SACK using the normal procedures as defined in [RFC2960] for classic SCTP. This includes, among other things: a) check if the SACK is received out-of-order; if not, b) adopt the Cumulative TSN ACK carried in the SACK as the new "Peer.Cumulative.TSN"; and c) process any gap reports if present (see [RFC2960]).

The data sender MUST then perform the following additional steps:

C1) Try to further advance the "Peer.Cumulative.TSN" locally, that is, to move "Peer.Cumulative.TSN" up as long as the chunk next in the out-queue is marked as acknowledged. For example (assuming a SACK arrived with the Cumulative TSN ACK = 102),

```
 out-queue at the end of  ==>   out-queue after cmTSN
 normal SACK processing           local advancement

         ...                              ...
cmTSN -> 102 acked                        102 acked
         103 acked                        103 acked
         104 acked              cmTSN ->  104 acked
         105                              105
         106 acked                        106 acked
         ...                              ...
```

Here, the data sender successfully advanced the "Peer.Cumulative.TSN" from 102 to 104 locally.

C2) Whenever a local advancement of the "Peer.Cumulative.TSN" is made, the data sender MUST send the data receiver a FORWARD TSN chunk containing the new value of the "Peer.Cumulative.TSN" (which is 104 in the above example).

Note, an endpoint MUST NOT use the forward TSN for any other purposes other than the above circumstance.

Note, if a TSN is indicated as missing by a SACK carrying gap

reports AND the TSN is earlier than the current "Peer.Cumulative.TSN", the data sender MUST NOT take any action on this TSN, i.e., it MUST ignore this missing report to this TSN. When this happens, it is normally an indication that a previous FORWARD TSN from the data sender may have been lost in the network.

Note, the detection criterion for out-of-order SACKs MUST remains the same as stated in RFC2960, that is, a SACK is only considered out-of-order if the Cumulative TSN ACK carried in the SACK is earlier than that of the previous received SACK (i.e., the comparison MUST NOT be made against the locally advanced "Peer.Cumulative.TSN").

The ULP primitive "DATA" (defined in Section 10.1 of [RFC2960]) should be expanded to contain an optional unreliable retransmission parameter to assign a "Unrel.Trans.Count" value to each user message to be sent to an unreliable stream.

4.3 Receive Unreliable Data

Regardless whether a DATA chunk arrives from a reliable stream or an unreliable stream, the receiver MUST perform the same TSN handling (e.g, duplicate detection, gap detection, SACK generation, cumulative TSN advancement, etc.) as defined in [RFC2960].

However, whenever a FORWARD TSN chunk arrives the data receiver MUST update its cumulative TSN to the value carried in the FORWARD TSN chunk, and MUST stop reporting any missing TSNs before the new cumulative TSN.

Whenever an unreliable DATA chunk arrives with the 'U' bit set to '0' (indicating ordered delivery) and is out of order, the receiver must hold the chunk for reordering. However since it is possible that the DATA chunk(s) being waited upon is one that will not be retransmitted by the sender, when a FORWARD TSN chunk arrives, the receiver MUST examine all of its unreliable stream reordering queues, and immediately make available for delivery any messages that carry a TSN earlier than the new cumulative TSN updated by the FORWARD TSN.

4.4. Other Issues on Unreliable Data

4.4.1 Unreliable Data Stream Multiplexing

Sometimes, it is desirable to aggregate different real time media streams (e.g., RTP streams) and send them over a single communication connection, and normally unreliable transport is preferred for these types of media streams.

With U-SCTP this is easily achieved by assigning each different media stream to a different unreliable SCTP stream and enabling the SCTP data bundling to perform the multiplexing.

The implementation of the data sender MAY use a bundling timer with a time interval adjusted to the timing characteristics of the specific media type in order to achieve the optimal multiplexing efficiency.

4.4.2 Fault Tolerant Unreliable Data Transfer

When the data receiver is multi-homed, unreliable data transfer using U-SCTP will obtain the same fault tolerance benefit as that of the reliable data services across an SCTP association.

This is because the data sender still follows the same failure detection rules and still counts the omitted retransmission against the association and the destination transport address to which the unreliable DATA chunk was originally sent. Thus, when failure occurs, the data sender will detect the failure and shift the unreliable data services to an alternate destination address, following the same procedures as defined in Section 8 of [RFC2960] for reliable data transfer.

4.4.3 Unreliable Data Out-of-order Detection

Detecting out-of-order data in an unreliable stream is useful for some applications (e.g. Fiber channel or SCSI over IP). With U-SCTP this becomes possible - the upper layer simply needs to examine the the stream sequence number of the arrived user messages to detect any missing data or out-of-order data. This detection only works when the DATA chunks are sent in order, i.e. their "U" bit MUST NOT be set.

5 Usage of the P-DATA chunk.

For some applications it is beneficial NOT to discard a data packet due to bit errors within the User Data portion. For this type of applications this new optional chunk type is defined.

All rules defined in [RFC2960] for DATA Chunks MUST be followed for the P-DATA chunk with the following exceptions:

E1) When passing a partial checksum user message to SCTP data sender, the upper layer should indicate how many bytes of the first portion of the user message need checksum protection. The data sender will then put this value into the Checksum Coverage field of the outbound P-DATA chunk.

E2) The Checksum Coverage field defines how much of this P-DATA chunk the Adler-32 checksum is to cover. During checksum computation the sender and receiver MUST use this field

to determine how much of a P-DATA chunk to add to the
Adler-32 checksum of the SCTP packet.

For each P-DATA in a received SCTP packet, after summing the

chunk header bytes and the specified number of User Data bytes to the checksum, the receiver MUST skip to the next chunk if this P-DATA is not the last chunk in the SCTP packet and MUST NOT include the rest of the User Data in the P-DATA chunk in its checksum computation.

E3) A sender MUST NOT send a P-DATA chunk unless the 'Partial Checksum Support' parameter was seen in the INIT or INIT-ACK from its peer.

It is important to note that P-DATA chunk uses the SAME TSN values as the normal DATA Chunk type. The sender and receiver do NOT hold separate TSN sequence spaces for DATA and P-DATA. Instead, the two chunk types use a single TSN sequence space.

In effect the P-DATA chunk is treated in all considerations to be a DATA chunk, with all of the normal DATA chunk rules for congestion control effecting this chunk. The only difference in treatment of P-DATA chunk comes during the calculation and verification of the Adler-32 checksum.

P-DATA chunk MAY be used with either a reliable or un-reliable stream, no restrictions are placed on its usage except those listed above.

Use of the P-DATA chunk may be under the control of the SCTP user. Hence, the ULP primitive "DATA" (see section 10.1 of RFC2960) should be expanded to contain an optional Checksum Coverage value.

When a reliable user message subscribing for partial checksum protection is fragmented at the SCTP sender, the sender shall calculate the Checksum Coverage value for each of the resulting P-DATA chunks, based upon the user's original coverage requirement.

6. Acknowledgments

The authors would like to thank Scott Bradner, Jon Berger, and others for their comments.

7. Authors' Addresses

Qiaobing Xie
Motorola, Inc.
1501 W. Shure Drive, #2309
Arlington Heights, IL 60004
USA
Tel: +1-847-632-3028
EMail: qxie1@email.mot.com

Randall R. Stewart
Cisco Systems, Inc.
8725 West Higgins Road
Tel: +1-815-477-2127
EMail: rrs@cisco.com

Suite 300
Chicago, Ill 60631

Chip Sharp Tel: +1-919-392-3121

Cisco Systems Inc. EMail:chsharp@cisco.com
7025 Kit Creek Road
Research Triangle Park, NC 27709
USA

Ian Rytina Tel: +61-3-9301-6164
Ericsson Australia EMail:ian.rytina@ericsson.com
37/360 Elizabeth Street
Melbourne, Victoria 3000
Australia

8. References

[RFC2026] Bradner, S., "The Internet Standards Process -- Revision 3", RFC 2026, October 1996.

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

[RFC2960] R. R. Stewart, Q. Xie, K. Morneault, C. Sharp, H. J. Schwarzbauer, T. Taylor, I. Rytina, M. Kalla, L. Zhang, and, V. Paxson, "Stream Control Transmission Protocol," RFC2960, October 2000.

This Internet Draft expires in 6 months from February 2001.