Network Working Group
INTERNET-DRAFT

R. R. Stewart
M. A. Ramalho
Cisco Systems
Q. Xie
Motorola
P. Conrad
Temple University
M. Rose
Invisible Worlds, Inc

expires in six months November 3,2000

SCTP Stream based flow control
<draft-stewart-srwnd-sctp-sigtran-01.txt>

Status of This Memo

Abstract

Taking advantage of the extensibility of SCTP, this document adds a standard method for SCTP to provide a stream based flow control mechanism.

Table Of Contents

1. Introduction

Taking advantage of the extensibility of SCTP, this document adds a standard method for SCTP to provide a stream based flow control mechanism. This mechanism uses the reliable chunk transfer extension [ADDIP] to carry the flow control restrictions to peer endpoints that support this option. Some of the benefits of this extension are:

A) The ability to minimize the occurrence of a single stream hogging all transport level resources (e.g. a_rwnd).

B) The ability to dynamically change the stream buffering limits as the situation changes within the application.

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. Parameter Formats

Stream flow control requests MUST be delivered in a reliable fashion. This information is used by the sending SCTP peer to limit how much information one stream may send (based upon feedback from the receiving peers application layer). Being that these flow changes MUST be reliably delivered and are considered control information, the methods specified in [ADDIP] is used to communicate this information. Therefore this document will only specify the new parameter required to carry the flow control requests from the receiver side to the sender side. For the proper procedures for the actual Reliable Chunk Transfer please see [ADDIP].

3.1 New Parameter Types

Variable Parameters	Type Value
Stream Flow limit Request	32771 (0xC003)

3.2 Stream Flow Limit Change

```
    0                   1                   2                   3
    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
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |        Type = 32771           |    Length = Variable          |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |        Stream Number 1        |    Flow Limit 1               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   \                                                               /
   /                                                               \
   \                                                               /
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |        Stream Number N          |    Flow Limit N             |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Stream Number n : 16 bits (unsigned integer)

This is the stream number that is requesting a limit be placed on the sender based on the applications receive buffer sizes.

Flow Limit n : 16 bits (unsigned integer)

This is the limit the receiver is requesting (in bytes) as to the maximum amount of data that the receiver may accept. Note that the value 0 holds a special meaning described in Section 4.1.

4. Procedures

A stream in SCTP is an uni-directional logical channel established from one to another associated SCTP endpoint, within which all user messages are delivered in sequence except for those submitted to the unordered delivery service which may arrive out of sequence. Since each stream is uni-directional and no feedback mechanism exists to limit a sender, it is possible for one unique stream to hog all of the transport level receiver window space. The mechanism defined here attempts to alleviate this problem by allowing the receiver side to communicate to the sender a limit on how much outstanding data may be sent within a particular stream.

The procedures defined here are broken down into two sides:

- The stream receiver or peer requesting the limit. And,

- the stream sender side or peer that MUST honor the limit request.

The receivers side is mainly involved with sending the request to the peer. The senders side is where the actual limitations and flow control will occur.

4.1 Stream Receiver side procedures

The receiver side SCTP makes decisions on stream flow control based on upper layer input. Normally the upper layer makes a request to limit all or a subset of the active streams that send data to it via an API interface. How this decision is made is outside the scope of this document but suggested usage characteristics can be found in Appendix A [Editors note: appendix A will be completed in a future draft].

Any time flow limits are made known to the SCTP endpoint by the application, the receiver side will create a Reliable Control Chunk (based on the rules found in [ADDIP]) and attach to it one or more stream flow limits with there respective stream number. If the receiver wishes to remove all limits (previously placed on a particular stream) it may do so by placing the special value '0' in the 'Flow Limit' field. Once acknowledged by the peer endpoint the receiver should consider the limit in place.

Note that the parameter type field upper two bits dictates that any parameter not understood should be skipped and reported to the sender with an Operational Error. If an Operational Error is received that indicates that the 'Stream Flow Limit Request' is not understood, the sender of the limit request MUST not send subsequent limit requests. The endpoint SHOULD also inform the upper level application that the peer endpoint does not support this feature.

If the sender of the request receives a Operational Error indicating that the REL-REQ chunk type (described in [ADDIP]) is not understood then the sender must not send subsequent limit requests. The endpoint SHOULD also inform the upper level application that the peer endpoint does not support this feature.

4.2 Stream Sender side procedures

When a 'Stream Flow Limit Request' is received the sender MUST record each flow limit with its appropriate stream.

After a limit is set on a stream the sender MUST obey the following rules when sending to the peer on that stream:

R1) When the upper layer application attempts to send to the peer on a stream, check the number of outstanding bytes sent to that stream (those TSN's in queue to be sent, which the cumulative TSN Acknowledgement has not passed, on this stream) versus the limit set for that stream (The last received limit for this stream is henceforth termed the current limit).

R2) If the number of outstanding bytes is greater than or equal to the current limit, the SCTP endpoint MUST reject the request and NOT queue the data for transmit. Instead it SHOULD return an error to the sending application.

R3) If the number of outstanding bytes is less than the current limit, validate that the data to be sent plus the number of outstanding bytes is smaller than or equal to this limit. If the user data plus the number of outstanding bytes is smaller than or equal to the current limit accept the data for transmit and queue the user data (increasing the number of outstanding data bytes on this stream). If the user data plus the number of outstanding bytes is larger than the current limit for this stream, the SCTP endpoint MUST reject the request and NOT queue the data for transmit and instead SHOULD return an error to the application.

R4) Any time a stream limit is updated to the value of 0, consider this indication to mean no limit is in effect for this stream.

Note that the effect of rule R3 above places a maximum size upon a sender. Even though SCTP may be capable of sending and reassembling larger user messages, by placing a flow limit on a stream this also gates the largest single user message a receiver is willing to accept.

5. Security Considerations

This extension is not deemed to create any additional security hazards then currently exist in an SCTP association. All of the threats and measures as defined in [RFC2960] are applicable to this feature.

6. IANA considerations

No new IANA considerations are added by this document. One new parameter type is being allocated for use by this feature.

7. Authors' Addresses

```
Randall R. Stewart                        Tel: +1-815-342-5222
Cisco Systems, Inc.                       EMail: rrs@cisco.com
8745 W. Higgins Road, Suite 200
Chicago, Ill  60631
USA

Micheal A. Ramalho                        Tel: +1-732-809-0188
Cisco Systems, Inc.                  EMail: mramalho@cisco.com
1802 Rue de la Porte
Wall Township, NJ 0719-3784

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

Phil Conrad                              Tel: +1-XXX-XXX-XXXX
Netlab Research Group       Email conrad@joda.cis.temple.edu
Dept. Of Computer &
Information Sciences
Temple University
1805 N Broad St.
Philadelphia, PA 19122
USA

Marshall T. Rose                          Tel: +1 707 789 3700
Invisible Worlds, Inc.              EMail: mrose@invisible.net
1179 North McDowell Boulevard   URI:   http://invisible.net/
Petaluma, CA  94954-6559
USA
```

7. References

[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," RFC XXXX, October 2000.

[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.

[RFC2402] S. Kent, R. Atkinson., "IP Authentication Header.",
          RFC 2402, November 1998.

[ADDIP]   R. R. Stewart,Q. Xie, M. Tuexen, "SCTP Dynamic Addition
          of IP addresses", Work in Progress, ietf draft.

Appendix A   Suggested application usage characteristics

[This section will be filled in in a future version of the draft]

This Internet Draft expires in 6 months from November, 2000