<div align="center">

Additional Policies for the Partially Reliable
Stream Control Transmission Protocol Extension

</div>

Abstract

   This document defines two additional policies for the Partially
   Reliable Stream Control Transmission Protocol (PR-SCTP) extension.
   These policies allow limitation of the number of retransmissions and
   prioritization of user messages for more efficient usage of the send
   buffer.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

   The Partially Reliable SCTP (PR-SCTP) extension defined in [RFC3758]
   provides a generic method for senders to abandon user messages. The
   decision to abandon a user message is sender side only, and the exact
   condition is called a "PR-SCTP policy" ([RFC3758] refers to them as
   "PR-SCTP Services"). [RFC3758] also defines one particular PR-SCTP
   policy, called "Timed Reliability". This allows the sender to
   specify a timeout for a user message after which the SCTP stack
   abandons the user message.

   This document specifies the following two additional PR-SCTP
   policies:

   Limited Retransmission Policy: Allows limitation of the number of
      retransmissions.

   Priority Policy: Allows removal of lower-priority messages if space
      for higher-priority messages is needed in the send buffer.

2. Conventions

   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].

3. Additional PR-SCTP Policies

   This section defines two new PR-SCTP policies, one in each
   subsection.

   Please note that it is REQUIRED to implement [RFC3758], if you want
   to implement these additional policies. However, these additional
   policies are OPTIONAL when implementing [RFC3758].

3.1. Limited Retransmissions Policy

   Using the Limited Retransmission Policy allows the sender of a user
   message to specify an upper limit for the number of retransmissions
   for each DATA chunk of the given user messages. The sender MUST
   abandon a user message if the number of retransmissions of any of the
   DATA chunks of the user message would exceed the provided limit. The
   sender MUST perform all other actions required for processing the
   retransmission event, such as adapting the congestion window and the
   retransmission timeout. Please note that the number of
   retransmissions includes both fast and timer-based retransmissions.

The sender MAY limit the number of retransmissions to 0. This will
result in abandoning the message when it would get retransmitted for
the first time. The use of this setting provides a service similar
to UDP, which also does not perform any retransmissions.

Please note that using this policy does not affect the handling of
the thresholds 'Association.Max.Retrans' and 'Path.Max.Retrans' as
specified in Section 8 of [RFC4960].

The WebRTC protocol stack (see [DATA-CHAN]) is an example of where
the Limited Retransmissions Policy is used.

3.2. Priority Policy

Using the Priority Policy allows the sender of a user message to
specify a priority. When storing a user message in the send buffer
while there is not enough available space, the SCTP stack at the
sender side MAY abandon other user message(s) of the same SCTP
association (with the same or a different stream) with a priority
lower than the provided one. User messages sent reliably are
considered to have a priority higher than all messages sent with the
Priority Policy. The algorithm for selecting the message(s) being
abandoned is implementation specific.

After lower-priority messages have been abandoned, high-priority
messages can be transferred without the send call blocking (if used
in blocking mode) or the send call failing (if used in non-blocking
mode).

The IP Flow Information Export (IPFIX) protocol stack (see [RFC7011])
is an example of where the Priority Policy can be used. Template
records would be sent with full reliability, while flow records
related to billing, security, and other monitoring would be sent
using the Priority Policy with varying priority. The priority of
security-related flow records would be set higher than the priority
of monitoring-related flow records.

4. Socket API Considerations

This section describes how the socket API defined in [RFC6458] is
extended to support the newly defined PR-SCTP policies, to provide
some statistical information, and to control the negotiation of the
PR-SCTP extension during the SCTP association setup.

Please note that this section is informational only.

4.1. Data Types

   This section uses data types from [IEEE.1003-1G.1997]: uintN_t means
   an unsigned integer of exactly N bits (e.g., uint16_t). This is the
   same as in [RFC6458].

4.2. Support for Added PR-SCTP Policies

   As defined in [RFC6458], the PR-SCTP policy is specified and
   configured by using the following sctp_prinfo structure:

   struct sctp_prinfo {
     uint16_t pr_policy;
     uint32_t pr_value;
   };

   When the Limited Retransmission Policy described in Section 3.1 is
   used, pr_policy has the value SCTP_PR_SCTP_RTX and the number of
   retransmissions is given in pr_value.

   When using the Priority Policy described in Section 3.2, pr_policy
   has the value SCTP_PR_SCTP_PRIO. The priority is given in pr_value.
   The value of zero is the highest priority, and larger numbers in
   pr_value denote lower priorities.

   The following table summarizes the possible parameter settings
   defined in [RFC6458] and this document:

   +-------------------+---------------------------+---------------+
   | pr_policy | pr_value | Specification |
   +-------------------+---------------------------+---------------+
SCTP_PR_SCTP_NONE	Ignored	[RFC6458]
SCTP_PR_SCTP_TTL	Lifetime in ms	[RFC6458]
SCTP_PR_SCTP_RTX	Number of retransmissions	Section 3.1
SCTP_PR_SCTP_PRIO	Priority	Section 3.2
   +-------------------+---------------------------+---------------+

4.3. Socket Option for Getting the Stream-Specific PR-SCTP Status
      (SCTP_PR_STREAM_STATUS)

   This socket option uses IPPROTO_SCTP as its level and
   SCTP_PR_STREAM_STATUS as its name. It can only be used with
   getsockopt() but not with setsockopt(). The socket option value uses
   the following structure:

   struct sctp_prstatus {
     sctp_assoc_t sprstat_assoc_id;
     uint16_t sprstat_sid;
     uint16_t sprstat_policy;
     uint64_t sprstat_abandoned_unsent;
     uint64_t sprstat_abandoned_sent;
   };

   sprstat_assoc_id: This parameter is ignored for one-to-one style
      sockets. For one-to-many style sockets, this parameter indicates
      for which association the user wants the information. It is an
      error to use SCTP_{CURRENT|ALL|FUTURE}_ASSOC in sprstat_assoc_id.

   sprstat_sid: This parameter indicates for which outgoing SCTP stream
      the user wants the information.

   sprstat_policy: This parameter indicates for which PR-SCTP policy
      the user wants the information. It is an error to use
      SCTP_PR_SCTP_NONE in sprstat_policy. If SCTP_PR_SCTP_ALL is used,
      the counters provided are aggregated over all supported policies.

   sprstat_abandoned_unsent: The number of user messages that have been
      abandoned using the policy specified in sprstat_policy on the
      stream specified in sprstat_sid for the association specified by
      sprstat_assoc_id, before any part of the user message could be
      sent.

   sprstat_abandoned_sent: The number of user messages that have been
      abandoned using the policy specified in sprstat_policy on the
      stream specified in sprstat_sid for the association specified by
      sprstat_assoc_id, after a part of the user message has been sent.

   There are separate counters for unsent and sent user messages because
   the SCTP_SEND_FAILED_EVENT supports a similar differentiation.
   Please note that an abandoned large user message requiring SCTP-level
   fragmentation is reported in the sprstat_abandoned_sent counter as
   soon as at least one fragment of it has been sent. Therefore, each
   abandoned user message is counted in either sprstat_abandoned_unsent
   or sprstat_abandoned_sent.

If more detailed information about abandoned user messages is
required, the subscription to the SCTP_SEND_FAILED_EVENT is
recommended. Please note that some implementations might choose not
to support this option, since it increases the resources needed for
an outgoing SCTP stream. For the same reasons, some implementations
might only support using SCTP_PR_SCTP_ALL in sprstat_policy.

sctp_opt_info() needs to be extended to support
SCTP_PR_STREAM_STATUS.

4.4. Socket Option for Getting the Association-Specific PR-SCTP Status (SCTP_PR_ASSOC_STATUS)

This socket option uses IPPROTO_SCTP as its level and
SCTP_PR_ASSOC_STATUS as its name. It can only be used with
getsockopt(), but not with setsockopt(). The socket option value
uses the same structure as described in Section 4.3:

```
struct sctp_prstatus {
  sctp_assoc_t sprstat_assoc_id;
  uint16_t sprstat_sid;
  uint16_t sprstat_policy;
  uint64_t sprstat_abandoned_unsent;
  uint64_t sprstat_abandoned_sent;
};
```

sprstat_assoc_id: This parameter is ignored for one-to-one style
   sockets. For one-to-many style sockets, this parameter indicates
   for which association the user wants the information. It is an
   error to use SCTP_{CURRENT|ALL|FUTURE}_ASSOC in sprstat_assoc_id.

sprstat_sid: This parameter is ignored.

sprstat_policy: This parameter indicates for which PR-SCTP policy
   the user wants the information. It is an error to use
   SCTP_PR_SCTP_NONE in sprstat_policy. If SCTP_PR_SCTP_ALL is used,
   the counters provided are aggregated over all supported policies.

sprstat_abandoned_unsent: The number of user messages that have been
   abandoned using the policy specified in sprstat_policy for the
   association specified by sprstat_assoc_id, before any part of the
   user message could be sent.

sprstat_abandoned_sent: The number of user messages that have been
   abandoned using the policy specified in sprstat_policy for the
   association specified by sprstat_assoc_id, after a part of the
   user message has been sent.

There are separate counters for unsent and sent user messages because
the SCTP_SEND_FAILED_EVENT supports a similar differentiation.
Please note that an abandoned large user message requiring SCTP-level
fragmentation is reported in the sprstat_abandoned_sent counter as
soon as at least one fragment of it has been sent. Therefore, each
abandoned user message is counted in either sprstat_abandoned_unsent
or sprstat_abandoned_sent.

If more detailed information about abandoned user messages is
required, the usage of the option described in Section 4.3 or the
subscription to the SCTP_SEND_FAILED_EVENT is recommended.

sctp_opt_info() needs to be extended to support SCTP_PR_ASSOC_STATUS.

4.5. Socket Option for Getting and Setting the PR-SCTP Support (SCTP_PR_SUPPORTED)

This socket option allows the enabling or disabling of the
negotiation of PR-SCTP support for future associations. For existing
associations, it allows one to query whether or not PR-SCTP support
was negotiated on a particular association.

Whether or not PR-SCTP is enabled by default is implementation
specific.

This socket option uses IPPROTO_SCTP as its level and
SCTP_PR_SUPPORTED as its name. It can be used with getsockopt() and
setsockopt(). The socket option value uses the following structure
defined in [RFC6458]:

```
struct sctp_assoc_value {
  sctp_assoc_t assoc_id;
  uint32_t assoc_value;
};
```

assoc_id: This parameter is ignored for one-to-one style sockets.
   For one-to-many style sockets, this parameter indicates upon which
   association the user is performing an action. The special
   sctp_assoc_t SCTP_FUTURE_ASSOC can also be used; it is an error to
   use SCTP_{CURRENT|ALL}_ASSOC in assoc_id.

assoc_value: A non-zero value encodes the enabling of PR-SCTP,
   whereas a value of 0 encodes the disabling of PR-SCTP.

sctp_opt_info() needs to be extended to support SCTP_PR_SUPPORTED.

5. Security Considerations

   This document does not add any security considerations to those given
   in [RFC4960], [RFC3758], and [RFC6458]. As indicated in the Security
   Considerations of [RFC3758], transport-layer security in the form of
   TLS over SCTP (see [RFC3436]) can't be used for PR-SCTP. However,
   DTLS over SCTP (see [RFC6083]) could be used instead. If DTLS over
   SCTP as specified in [RFC6083] is used, the Security Considerations
   of [RFC6083] do apply. It should also be noted that using PR-SCTP
   for an SCTP association doesn't allow that association to behave more
   aggressively than an SCTP association not using PR-SCTP.

6. References

6.1. Normative References

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

   [RFC3758] Stewart, R., Ramalho, M., Xie, Q., Tuexen, M., and P.
              Conrad, "Stream Control Transmission Protocol (SCTP)
              Partial Reliability Extension", RFC 3758, May 2004,
              <http://www.rfc-editor.org/info/rfc3758>.

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

6.2. Informative References

   [RFC3436] Jungmaier, A., Rescorla, E., and M. Tuexen, "Transport
              Layer Security over Stream Control Transmission Protocol",
              RFC 3436, December 2002,
              <http://www.rfc-editor.org/info/rfc3436>.

   [RFC6083] Tuexen, M., Seggelmann, R., and E. Rescorla, "Datagram
              Transport Layer Security (DTLS) for Stream Control
              Transmission Protocol (SCTP)", RFC 6083, January 2011,
              <http://www.rfc-editor.org/info/rfc6083>.

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

   [RFC7011] Claise, B., Ed., Trammell, B., Ed., and P. Aitken,
              "Specification of the IP Flow Information Export (IPFIX)
              Protocol for the Exchange of Flow Information", STD 77,
              RFC 7011, September 2013,
              <http://www.rfc-editor.org/info/rfc7011>.

   [DATA-CHAN]
              Jesup, R., Loreto, S., and M. Tuexen, "WebRTC Data
              Channels", Work in Progress, draft-ietf-rtcweb-data-
              channel-13, January 2015.

   [IEEE.1003-1G.1997]
              IEEE, "Protocol Independent Interfaces", IEEE Standard
              1003.1G, March 1997.

Acknowledgments

Authors' Addresses

   Michael Tuexen
   Muenster University of Applied Sciences
   Stegerwaldstrasse 39
   48565 Steinfurt
   Germany

   EMail: tuexen@fh-muenster.de

   Robin Seggelmann
   Metafinanz Informationssysteme GmbH
   Leopoldstrasse 146
   80804 Muenchen
   Germany

   EMail: rfc@robin-seggelmann.com

   Randall R. Stewart
   Netflix, Inc.
   Chapin, SC   29036
   United States

   EMail: randall@lakerest.net

   Salvatore Loreto
   Ericsson
   Hirsalantie 11
   Jorvas 02420
   Finland

   EMail: Salvatore.Loreto@ericsson.com