Network Working Group
Internet-Draft
Intended status: Experimental
Expires: July 21, 2012

M. Becke
T. Dreibholz
University of Duisburg-Essen
J. Iyengar
Franklin and Marshall College
P. Natarajan
Cisco Systems
M. Tuexen
Muenster Univ. of Applied Sciences

January 18, 2012

Load Sharing for the Stream Control Transmission Protocol (SCTP)
draft-tuexen-tsvwg-sctp-multipath-03.txt

Abstract

The Stream Control Transmission Protocol (SCTP) supports multi-homing for providing network fault tolerance. However, mainly one path is used for data transmission. Only timer-based retransmissions are carried over other paths as well.

This document describes how multiple paths can be used simultaneously for transmitting user messages.

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 http://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 July 21, 2012.

Table of Contents

1. Introduction

One of the important features of the Stream Control Transmission Protocol (SCTP), which is currently specified in [RFC4960], is network fault tolerance. This feature is for example required for Reliable Server Pooling (RSerPool, [RFC5351]). Therefore, transmitting messages over multiple paths is supported, but only for redundancy. So [RFC4960] does not specify how to use multiple paths simultaneously.

This document overcomes this limitation by specifying how multiple paths can be used simultaneously. This has several benefits:

- Improved bandwidth usage.
- Better availability check with real user messages compared to HEARTBEAT-based information.

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. Load Sharing

Basic requirement for applying SCTP load sharing is the Concurrent Multipath Transfer (CMT) extension of SCTP, which utilises multiple paths simultaneously. We denote CMT-enabled SCTP as CMT-SCTP throughout this document. CMT-SCTP is introduced in [IAS06] and in more detail in [I06], some illustrative examples of chunk handling are provided in [DBP10a]. CMT-SCTP provides three modifications to standard SCTP (split Fast Retransmissions, appropriate congestion window growth and delayed SACKs), which are described in the following subsections.

3.1. Split Fast Retransmissions

Paths with different latencies lead to overtaking of DATA chunks. This leads to gap reports, which are handled by Fast Retransmissions. However, due to the fact that multiple paths are used simultaneously, these Fast Retransmissions are usually useless and furthermore lead to a decreased congestion window size.

To avoid unnecessary Fast Retransmissions, the sender has to keep track of the path each DATA chunk has been sent on and consider

transmission paths before performing Fast Retransmissions. That is, on reception of a SACK, the sender MUST identify the highest acknowledged TSN on each path. A chunk SHOULD only be considered as missing if its TSN is smaller than the highest acknowledged TSN on its path. Section 3.1 of [DBP10a] contains an illustrated example.

3.2. Appropriate Congestion Window Growth

The congestion window adaptation algorithm for SCTP [RFC4960] allows increasing the congestion window only when a new cumulative ack (CumAck) is received by a sender. When SACKs with unchanged CumAcks are generated (due to reordering) and later arrive at a sender, the sender does not modify its congestion window. Since a CMT-SCTP receiver naturally observes reordering, many SACKs are sent containing new gap reports but not new CumAcks. When these gaps are later acked by a new CumAck, congestion window growth occurs, but only for the data newly acked in the most recent SACK. Data previously acked through gap reports will not contribute to congestion window growth, in order to prevent sudden increases in the congestion window resulting in bursts of data being sent.

To overcome the problems described above, the congestion window growth has to be handled as follows [IAS06]:

- The sender SHOULD keep track of the earliest non-retransmitted outstanding TSN per path.

- The sender SHOULD keep track of the earliest retransmitted outstanding TSN per path.

- The in-order delivery per path SHOULD be deduced.

- The congestion window of a path SHOULD be increased when the earliest non-retransmitted outstanding TSN of this path is advanced ("Pseudo CumAck") OR when the earliest retransmitted outstanding TSN of this path is advanced ("RTX Pseudo CumAck").

Section 3.2 of [DBP10a] contains an illustrated example of appropriate congestion window handling for CMT-SCTP.

3.3. Appropriate Delayed Acknowledgements

Standard SCTP [RFC4960] sends a SACK as soon as an out-of-sequence TSN has been received. Delayed Acknowledgements are only allowed if the received TSNs are in sequence. However, due to the load balancing of CMT-SCTP, DATA chunks may overtake each other. This leads to a high number of out-of-sequence TSNs, which have to be acknowledged immediately. Clearly, this behaviour increases the

overhead traffic (usually nearly one SACK chunk for each received packet containing a DATA chunk).

Delayed Acknowledgements for CMT-SCTP are handled as follows:

- In addition to [RFC4960], delaying of SACKs SHOULD *also* be applied for out-of-sequence TSNs.

- A receiver MUST maintain a counter for the number of DATA chunks received before sending a SACK. The value of the counter is stored into each SACK chunk (FIXME: add details; needs reservation of flags bits by IANA). After transmitting a SACK, the counter MUST be reset to 0. Its initial value MUST be 0.

- The SACK handling procedure for a missing TSN M is extended as follows:

  * If all newly acknowledged TSNs have been transmitted over the same path:

    + If there are newly acknowledged TSNs L and H so that L <= M <= H, the missing count of TSN M SHOULD be incremented by one (like for standard SCTP according to [RFC4960]).

    + Else if all newly acknowledged TSNs N satisfy the condition M <= N, the missing count of TSN M SHOULD be incremented by the number of TSNs reported in the SACK chunk.

  * Otherwise (that is, there are newly acknowledged TSNs on different paths), the missing count of TSN M SHOULD be incremented by one (like for standard SCTP according to [RFC4960]).

Section 3.3 of [DBP10a] contains an illustrated example of Delayed Acknowledgements for CMT-SCTP.

4. Buffer Blocking Mitigation

TBD. See [ADB11], [DBR10].

4.1. Sender Buffer Splitting

TBD. See [ADB11], [DBR10].

4.2. Receiver Buffer Splitting

TBD. See [ADB11], [DBR10].

4.3. Problems during Path Failure

This section discusses CMT's receive buffer related problems during path failure, and proposes a solution for the same.

4.3.1. Problem Description

Link failures arise when a router or a link connecting two routers fails due to link disconnection, hardware malfunction, or software error. Overloaded links caused by flash crowds and denial-of-service (DoS) attacks also degrade end-to-end communication between peer hosts. Ideally, the routing system detects link failures, and in response, reconfigures the routing tables and avoids routing traffic via the failed link. However, existing research highlights problems with Internet backbone routing that result in long route convergence times. The pervasiveness of path failures motivated us to study their impact on CMT, since CMT achieves better throughput via simultaneous data transmission over multiple end-to-end paths.

CMT is an extension to SCTP, and therefore retains SCTP's failure detection process. A CMT sender uses a tunable failure detection threshold called Path.Max.Retrans (PMR). When a sender experiences more than PMR consecutive timeouts while trying to reach an active destination, the destination is marked as failed. With PMR=5, the failure detection takes 6 consecutive timeouts or 63s. After every timeout, the CMT sender continues to transmit new data on the failed path increasing the chances of receive buffer (rbuf) blocking and degrading CMT performance during permanent and short-term path failures [NEA08].

4.3.2. Solution: Potentially-failed Destination State

To mitigate the rbuf blocking, we introduce a new destination state called "potentially-failed" state in SCTP (and CMT's) failure detection process [I-D.nishida-tsvwg-sctp-failover]. This solution is based on the rationale that loss detected by a timeout implies either severe congestion or failure en route. After a single timeout on a path, a sender is unsure, and marks the corresponding destination as "potentially-failed" (PF). A PF destination is not used for data transmission or retransmission. CMT's retransmission policies are augmented to include the PF state. Performance evaluations prove that the PF state significantly reduces rbuf blocking during failure detection [NEA08].

4.4. Non-Renegable SACK

This section discusses problems with SCTP's SACK mechanism and how it affects the send buffer and CMT performance.

4.4.1. Problem Description

Gap-acks acknowledge DATA chunks that arrive out-of-order to a transport layer data receiver. A gap-ack in SCTP is advisory, in that, while it notifies a data sender about the reception of indicated DATA chunks, the data receiver is permitted to later discard DATA chunks that it previously had gap-acked. Discarding a previously gap-acked DATA chunk is known as "reneging". Because of the possibility of reneging in SCTP, any gap-acked DATA chunk MUST NOT be removed from the data sender's retransmission queue until the DATA chunk is later CumAcked.

Situations exist when a data receiver knows that reneging on a particular out-of-order DATA chunk will never take place, such as (but not limited to) after an out-of-order DATA chunk is delivered to the receiving application. With current SACKs in SCTP, it is not possible for a data receiver to inform a data sender if or when a particular out-of-order "deliverable" DATA chunk has been "delivered" to the receiving application. Thus the data sender MUST keep a copy of every gap-acked out-of-order DATA chunk(s) in the data sender's retransmission queue until the DATA chunk is CumAcked. This use of the data sender's retransmission queue is wasteful. The wasted buffer often degrades CMT performance; the degradation increases when a CMT flow traverses via paths with disparate end-to-end properties [NEY08].

4.4.2. Solution: Non-Renegable SACKs

Non-Renegable Selective Acknowledgments (NR-SACKs) [I-D.natarajan-tsvwg-sctp-nrsack] are a new kind of acknowledgements, extending SCTP's SACK chunk functionalities. The NR-SACK chunk is an extension of the existing SACK chunk. Several fields are identical, including the Cumulative TSN Ack, the Advertised Receiver Window Credit (a_rwnd), and Duplicate TSNs. These fields have the same semantics as described in [RFC4960].

NR-SACKs also identify out-of-order DATA chunks that a receiver either: (1) has delivered to its receiving application, or (2) takes full responsibility to eventually deliver to its receiving application. These out-of-order DATA chunks are "non-renegable." Non-Renegable data are reported in the NR Gap Ack Block field of the NR-SACK chunk as described [I-D.natarajan-tsvwg-sctp-nrsack]. We refer to non-renegable selective acknowledgements as "nr-gap-acks."

When an out-of-order DATA chunk is nr-gap-acked, the data sender no longer needs to keep that particular DATA chunk in its retransmission queue, thus allowing the data sender to free up its buffer space sooner than if the DATA chunk were only gap-acked. NR-SACKs improve send buffer utilization and throughput for CMT flows [NEY08].

5. Handling of Shared Bottlenecks

5.1. Introduction

CMT-SCTP assumes all paths to be disjoint. Since each path independently uses a TCP-like congestion control, an SCTP association using N paths over the same bottleneck acquires N times the bandwidth of a concurrent TCP flow. This is clearly unfair. A reliable detection of shared bottlenecks is impossible in arbitrary networks like the Internet. Therefore, [DBA11], [DBP10b] apply the idea of Resource Pooling to CMT-SCTP. Resource Pooling (RP) denotes "making a collection of resources behave like a single pooled resource" [WHB09]. The modifications of RP-enabled CMT-SCTP, further denoted as CMT/RP-SCTP, are described in the following subsections. A detailed description of CMT/RP-SCTP, including congestion control examples, can be found in [DBA11], [DBP10b].

5.2. Initial Values

TDB.

5.3. Congestion Window Growth

TDB. See [DBA11].

5.4. Congestion Window Decrease

TDB. See [DBA11].

6. Chunk Scheduling

TDB. See [DST10].

7. Application Programming Interface

See [I-D.dreibholz-tsvwg-sctpsocket-multipath] and [I-D.dreibholz-tsvwg-sctpsocket-sqinfo].

8. IANA Considerations

TBD.

9. Security Considerations

This document does not add any additional security considerations in addition to the ones given in [RFC4960].

10. References

10.1. Normative References

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

[RFC4960] Stewart, R., "Stream Control Transmission Protocol", RFC 4960, September 2007.

[RFC5351] Lei, P., Ong, L., Tuexen, M., and T. Dreibholz, "An Overview of Reliable Server Pooling Protocols", RFC 5351, September 2008.

[I-D.nishida-tsvwg-sctp-failover] Nishida, Y., Natarajan, P., and A. Caro, "Quick Failover Algorithm in SCTP", draft-nishida-tsvwg-sctp-failover-04 (work in progress), September 2011.

[I-D.natarajan-tsvwg-sctp-nrsack] Ekiz, N., Amer, P., Natarajan, P., Stewart, R., and J. Iyengar, "Non-Renegable Selective Acknowledgements (NR-SACKs) for SCTP", draft-natarajan-tsvwg-sctp-nrsack-08 (work in progress), August 2011.

[I-D.dreibholz-tsvwg-sctpsocket-multipath] Dreibholz, T. and M. Becke, "SCTP Socket API Extensions for Concurrent Multipath Transfer", draft-dreibholz-tsvwg-sctpsocket-multipath-02 (work in progress), October 2011.

[I-D.dreibholz-tsvwg-sctpsocket-sqinfo] Dreibholz, T., Seggelmann, R., and M. Becke, "Sender Queue Info Option for the SCTP Socket API", draft-dreibholz-tsvwg-sctpsocket-sqinfo-02 (work in progress), October 2011.

10.2. Informative References

[I06] Iyengar, J., "End-to-End Concurrent Multipath Transfer Using Transport Layer Multihoming", PhD Dissertation Computer Science Dept., University of Delaware, April 2006.

[IAS06] Iyengar, J., Amer, P., and R. Stewart, "Concurrent Multipath Transfer Using SCTP Multihoming Over Independent End-to-End Paths", Journal IEEE/ACM Transactions on Networking, October 2006.

[NEA08] Natarajan, P., Ekiz, N., Iyengar, J., Amer, P., and R. Stewart, "Concurrent Multipath Transfer Using Transport Layer Multihoming: Introducing the Potentially-failed Destination State", Proceedings of the IFIP Networking, May 2008.

[NEY08] Natarajan, P., Ekiz, N., Yilmaz, E., Amer, P., Iyengar, J., and R. Stewart, "Non-Renegable Selective Acknowledgments (NR-SACKs) for SCTP", Proceedings of the 16th IEEE International Conference on Network Protocols (ICNP), October 2008.

[WHB09] Wischik, D., Handley, M., and M. Braun, "The Resource Pooling Principle", Journal ACM SIGCOMM Computer Communication Review, October 2009.

[DBP10a] Dreibholz, T., Becke, M., Pulinthanath, J., and E. Rathgeb, "Implementation and Evaluation of Concurrent Multipath Transfer for SCTP in the INET Framework", Proceedings of the 3rd ACM/ICST OMNeT++ Workshop, March 2010.

[DBP10b] Dreibholz, T., Becke, M., Pulinthanath, J., and E. Rathgeb, "Applying TCP-Friendly Congestion Control to Concurrent Multipath Transfer", Proceedings of the IEEE 24th International Conference on Advanced Information Networking and Applications (AINA), April 2010.

[DBR10] Dreibholz, T., Becke, M., Rathgeb, E., and M. Tuexen, "On the Use of Concurrent Multipath Transfer over Asymmetric Paths", Proceedings of the IEEE Global Communications Conference (GLOBECOM), December 2010.

[DST10] Dreibholz, T., Seggelmann, R., Tuexen, M., and E. Rathgeb, "Transmission Scheduling Optimizations for Concurrent Multipath Transfer", Proceedings of the 8th International

Workshop on Protocols for Future, Large-Scale and Diverse Network Transports (PFLDNeT) , November 2010.

[ADB11] Adhari, H., Dreibholz, T., Becke, M., Rathgeb, E., and M. Tuexen, "Evaluation of Concurrent Multipath Transfer over Dissimilar Paths", Proceedings of the 1st International Workshop on Protocols and Applications with Multi-Homing Support (PAMS), March 2011.

[DBA11] Dreibholz, T., Becke, M., Adhari, H., and E. Rathgeb, "On the Impact of Congestion Control for Concurrent Multipath Transfer on the Transport Layer", Proceedings of the 11th IEEE International Conference on Telecommunications (ConTEL), June 2011.

Authors' Addresses

Martin Becke
University of Duisburg-Essen, Institute for Experimental Mathematics
Ellernstrasse 29
45326 Essen, Nordrhein-Westfalen
Germany

Phone: +49-201-183-7667
Fax: +49-201-183-7673
Email: martin.becke@uni-due.de

Thomas Dreibholz
University of Duisburg-Essen, Institute for Experimental Mathematics
Ellernstrasse 29
45326 Essen, Nordrhein-Westfalen
Germany

Phone: +49-201-183-7637
Fax: +49-201-183-7673
Email: dreibh@iem.uni-due.de
URI: http://www.iem.uni-due.de/~dreibh/

Janardhan Iyengar
Franklin and Marshall College, Mathematics and Computer Science
PO Box 3003
Lancaster, Pennsylvania 17604-3003
USA

Phone: +1-717-358-4774
Email: jiyengar@fandm.edu
URI: http://www.fandm.edu/jiyengar/

Preethi Natarajan
Cisco Systems
425 East Tasman Drive
San Jose, California 95134
USA

Email: prenatar@cisco.com

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

Email: tuexen@fh-muenster.de