Network Working Group
Internet-Draft
Intended status: Experimental
Expires: September 3, 2014

Y. Nishida
GE Global Research
P. Natarajan
Cisco Systems
A. Caro
BBN Technologies
P. Amer
University of Delaware
K. Nielsen
Ericsson

March 2, 2014

Quick Failover Algorithm in SCTP
draft-ietf-tsvwg-sctp-failover-03.txt

Abstract

One of the major advantages of SCTP is supporting multi-homed communication. If a multi-homed end-point has a redundant network connections, the SCTP associations have a good chance to survive network failures by migrating traffic from inactive networks to active ones. However, if the SCTP standard is followed, there can be a significant delay during the migration. During this period, SCTP might not be able to transmit much data to the peer. This issue drastically impairs the usability of SCTP in some situations. This memo describes the issue of the SCTP failover mechanism and specifies an alternative failover procedure for SCTP that improves its performance during and after failover. The procedures require only minimal modifications to the current specification.

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 September 3, 2014.

Table of Contents

1. Introduction

The Stream Control Transmission Protocol (SCTP) as specified in [RFC4960] supports multihoming at the transport layer -- an SCTP association can bind to multiple IP addresses at each endpoint. SCTP's multihoming features include failure detection and failover procedures to provide network interface redundancy and improved end-to-end fault tolerance.

In SCTP's current failure detection procedure, the sender must experience Path.Max.Retrans (PMR) number of consecutive failed retransmissions on a destination before detecting a path failure. The sender fails over to an alternate active destination only after failure detection. Until detecting the failover, the sender continues to transmit data on the failed path, which degrades the SCTP performance. Concurrent Multipath Transfer (CMT) [IYENGAR06] is an extension to SCTP and allows the sender to transmit data on multiple paths simultaneously. Research [NATARAJAN09] shows that the current failure detection procedure worsens CMT performance during failover and can be significantly improved by employing a better failover algorithm.

This document specifies an alternative failure detection procedure for SCTP (and CMT) that improves the SCTP (and CMT) performance during a failover.

Also the operation after a failover impacts the performance of the protocol. With [RFC4960] procedures, SCTP will, after a failover from the primary path, switch back to use the primary path for data transfer as soon as this path becomes available. From a performance perspective, as confirmed in research [CARO02], such a switchback of the data transmission path is not optimal in general. As an alternative option to the switchback operation of [RFC4960], this document specifies the support the Permanent Failover switchover procedures proposed by [CARO02].

2. Conventions and Terminology

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. Issues with the SCTP Path Management

SCTP can utilize multiple IP addresses for a single SCTP association. Each SCTP endpoint exchanges the list of its usable addresses during initial negotiation with its peer. Then the endpoints select one address from the peer's list and define this as the primary destination. During normal transmission, SCTP sends all user data to the primary destination. Also, it sends heartbeat packets to all idle destinations at a certain interval to check the reachability of the path. Idle destinations normally include all non-primary destinations.

If a sender has multiple active destination addresses, it can retransmit data to secondary destination address, when the transmission to the primary times out.

When a sender receives an acknowledgment for DATA or HEARTBEAT chunks sent to one of the destination addresses, it considers that destination to be active. If it fails to receive acknowledgments, the error count for the address is increased. If the error counter exceeds the protocol parameter 'Path.Max.Retrans', SCTP endpoint considers the address to be inactive.

The failover process of SCTP is initiated when the primary path becomes inactive (error counter for the primary path exceeds Path.Max.Retrans). If the primary path is marked inactive, SCTP chooses a new destination address from one of the active destinations and start using this address to send data to. If the primary path becomes active again, SCTP uses the primary destination for subsequent data transmissions and stop using non-primary one.

One issue with this failover process is that it usually takes significant amount of time before SCTP switches to the new destination. Let's say the primary path on a multi-homed host becomes unavailable and the RTO value for the primary path at that time is around 1 second, it usually takes over 60 seconds before SCTP starts to use the secondary path. This is because the recommended value for Path.Max.Retrans in the standard is 5, which requires 6 consecutive timeouts before failover takes place. Before SCTP switches to the secondary address, SCTP keeps trying to send packets to the primary and only retransmitted packets are sent to the secondary can be reached at the receiver. This slow failover process can cause significant performance degradation and will not be acceptable in some situations.

Another issue is that once the primary path is active again, the traffic is switched back. This is not optimal in general.

4. Existing Solutions for Smooth Failover

The following approaches are conceivable for the solutions of this issue.

4.1. Reduce Path.Max.Retrans (PMR)

Smaller values for Path.Max.Retrans shorten the failover duration. In fact, this is recommended in some research results [JUNGMAIER02] [GRINNEMO04] [FALLON08]. For example, if when Path.Max.Retrans=0, SCTP switches to another destination on a single timeout. However, smaller value for Path.Max.Retrans also results in spurious failover. In addition, smaller Path.Max.Retrans values also affect 'Association.Max.Retrans' values. When the SCTP association's error count (sum of error counts on all ACTIVE paths) exceeds Association.Max.Retrans threshold, the SCTP sender considers the peer endpoint unreachable and terminates the association. Therefore, Section 8.2 in [RFC4960] recommends that Association.Max.Retrans value should not be larger than the summation of the Path.Max.Retrans of each of the destination addresses, else the SCTP sender considers its peer reachable even when all destinations are INACTIVE. To avoid such inconsistent behavior an SCTP implementation SHOULD reduce Association.Max.Retrans accordingly whenever it reduces Path.Max.Retrans. However, smaller Association.Max.Retrans value increases chances of association termination during minor congestion events.

Another issue is that the interval of heartbeat packet: 'HB.interval' could be in the order of seconds (recommended value is 30 seconds). When the primary path becomes inactive, the next HB can be transmitted only seconds later. Meanwhile, the primary path may have recovered. In such situations, post failover, an endpoint is forced to wait on the order of seconds before the endpoint can resume transmission on the primary path.

The advantage of tuning Path.Max.Retrans is that it requires no modification to the current standard. However, as we discuss above tuning Path.Max.Retrans ignores several recommendations in [RFC4960]. In addition, some research results indicate path bouncing caused by spurious failover does not cause serious problems. We discuss the effect of path bouncing in Section 5.2.

4.2. Adjust RTO related parameters

As several research results indicate, we can also shorten the duration of failover process by adjusting RTO related parameters [JUNGMAIER02] [FALLON08]. During failover process, RTO keeps being doubled. However, if we can choose smaller value for RTO.max, we can

stop the exponential growth of RTO at some point. Also, choosing smaller values for RTO.initial or RTO.min can contribute to keep RTO value small.

Similar to reducing Path.Max.Retrans, the advantage of this approach is that it requires no modification to the current specification, although it needs to ignore several recommendations described in the Section 15 of [RFC4960]. However, this approach requires to have enough knowledge about the network characteristics between end points. Otherwise, it can introduce adverse side-effects such as spurious timeouts.

5. SCTP with Potentially-Failed Destination State (SCTP-PF)

5.1. SCTP-PF Description

SCTP-PF stems from the following two observations about SCTP's failure detection procedure:

- In order to minimize performance impact during failover, the sender should avoid transmitting data to the failed destination as early as possible. In the current SCTP path management scheme, the sender stops transmitting data to a destination only after the destination is marked Failed. Thus, a smaller PMR value is ideal so that the sender transitions a destination to the Failed state quicker.

- Smaller PMR values increase the chances of spurious failure detection where the sender incorrectly marks a destination as Failed during periods of temporary congestion. Larger PMR values are preferable to avoid spurious failure detection.

From the above observations it is clear that tweaking the PMR value involves the following tradeoff -- a lower value improves performance but increases the chances of spurious failure detection, whereas a higher value degrades performance and reduces spurious failure detection in a wide range of path conditions. Thus, tweaking the association's PMR value is an incomplete solution to address performance impact during failure.

This proposal introduces a new "Potentially-failed" (PF) destination state in SCTP's path management procedure. The PF state was originally proposed to improve CMT performance [NATARAJAN09]. The PF state is an intermediate state between Active and Failed states. SCTP's failure detection procedure is modified to include the PF state. The new failure detection algorithm assumes that loss detected by a timeout implies either severe congestion or failure en-route. After a number of consecutive timeouts on a path, the sender is unsure, and marks the corresponding destination as PF. A PF destination is not used for data transmission except in special cases (discussed below). The new failure detection algorithm requires only sender-side changes. Details are:

1. The sender maintains a new tunable parameter called Potentially-failed.Max.Retrans (PFMR). The recommended value of PFMR = 0 when quick failover is used. When PFMR is larger or equal to PMR, quick failover is turned off.

2. Each time the T3-rtx timer expires on an active destination, the error counter of that destination address will be incremented.

When the value in the error counter exceeds PFMR, the endpoint should mark the destination transport address as PF.

3. The sender SHOULD avoid data transmission to PF destinations. When all destinations are in either PF or Inactive state, the sender MAY either move the destination from PF to Active state (and transmit data to the active destination) or the sender MAY transmit data to a PF destination. In the former scenario, (i) the sender MUST NOT notify the ULP about the state transition, and (ii) MUST NOT clear the destination's error counter. It is recommended that the sender picks the PF destination with least error count (fewest consecutive timeouts) for data transmission. In case of a tie (multiple PF destinations with same error count), the sender MAY choose the last active destination.

4. Only heartbeats MUST be sent to PF destination(s) once per RTO. This means the sender SHOULD ignore HB.interval for PF destinations. If an heartbeat is unanswered, the sender increments the error counter and exponentially backs off the RTO value. If error counter is less than PMR, the sender SHOULD transmit another heartbeat immediately after T3-timer expiration.

5. When the sender receives an heartbeat ACK from a PF destination, the sender clears the destination's error counter and transitions the PF destination back to Active state. The sender should perform slow-start as specified in Section 7.2.1 of [RFC4960] when it sends data on this destination.

6. Additional (PMR - PFMR) consecutive timeouts on a PF destination confirm the path failure, upon which the destination transitions to the Inactive state. As described in [RFC4960], the sender (i) SHOULD notify ULP about this state transition, and (ii) transmit heartbeats to the Inactive destination at a lower frequency as described in Section 8.3 of [RFC4960].

7. When all destinations are in the Inactive state, the sender picks one of the Inactive destinations for data transmission. This proposal recommends that the sender picks the Inactive destination with least error count (fewest consecutive timeouts) for data transmission. In case of a tie (multiple Inactive destinations with same error count), the sender MAY choose the last active destination.

8. ACKs for retransmissions do not transition a PF destination back to Active state, since a sender cannot disambiguate whether the ack was for the original transmission or the retransmission(s).

9. SCTP shall provide the means to expose the PF state of its destinations as well as SCTP SHOULD notify the ULP of the state transitions from Active to PF and from PF to Active state. SCTP can provide the means to suppress exposure of PF state and association state transitions and in this case the ULP MAY make SCTP suppress exposure of PF state to ULP. In this case the ULP will rely solely on the [RFC4960] state machine even if quick failover function is activated in SCTP.

5.2. Effect of Path Bouncing

The methods described above can accelerate the failover process. Hence, they might introduce the path bouncing effect where the sender keeps changing the data transmission path frequently. This sounds harmful to the data transfer, however several research results indicate that there is no serious problem with SCTP in terms of path bouncing effect [CARO04] [CARO05].

There are two main reasons for this. First, SCTP is basically designed for multipath communication, which means SCTP maintains all path related parameters (CWND, ssthresh, RTT, error count, etc) per each destination address. These parameters cannot be affected by path bouncing. In addition, when SCTP migrates the data transfer to another path, it starts with the minimal or the initial CWND. Hence, there is little chance for packet reordering or duplicating.

Second, even if all communication paths between the end-nodes share the same bottleneck, the quick failover results in a behavior already allowed by [RFC4960].

5.3. Permanent Failover

Post failover then, by [RFC4960] behavior, an SCTP sender migrates the traffic back to the original primary destination once this destination becomes active anew. As the CWND towards the original primary destination has to be rebuilt once data transfer resumes, the switch back to use the original primary path is not always optimal. Indeed [CARO02] shows that the switch over to the original primary may degrade SCTP performance compared to continuing data transmission on the same path, especially, but not only, in scenarios where this path's characteristics are better. In order to mitigate this performance degradation, Permanent Failover operation was proposed in [CARO02]. When SCTP changes the destination due to failover, Permanent Failover marks it as new primary. This means Permanent Failover allows SCTP sender to continue data transmission to the path even after the old primary destination becomes active again. This is achieved by having SCTP perform a switchover of the primary path to an alternative working path rather than having SCTP switch back data

transfer to the (previous) primary path.

The manner of switchover operation that is most optimal in a given scenario depends on the relative quality of a set primary path versus the quality of alternative paths available as well as it depends on the extent to which it is desired for the mode of operation to enforce traffic distribution over a number of network paths. I.e., load distribution of traffic from multiple SCTP associations may be sought to be enforced by distribution of the set primary paths with [RFC4960] switchback operation. However as [RFC4960] switchback behavior is suboptimal in certain situations, especially in scenarios where a number of equally good paths are available, it is recommended for SCTP to support also, as alternative behavior, the Permanent Failover modes of operation where forced switch back to a previously failed primary path is not always performed. The Permanent Failover operation requires only sender side changes. Details, as originally outlined in [CARO02], are:

1. The sender maintains a new tunable parameter, called Primary.Switchover.Max.Retrans (PSMR). When the path error counter on a set primary path exceeds PSMR, the SCTP implementation autonomously selects and sets a new primary path.

2. The primary path selected by the SCTP implementation shall be the path which at the given time would be chosen for data transfer. A previously failed primary path may come in use as data transfer path as per normal path selection when the present data transfer path fails.

3. The recommended value of PSMR is PFMR when Permanent failover is used. This means that no forced switchback to a previously failed primary path is performed.

4. It must be possible to disable the Permanent Failover and obtain the standard switchback operation of [RFC4960].

We recommend that SCTP-PF should stick to the standard RFC4960 behavior as default, i.e., switch back to the old primary destination once the destination becomes active again. However, implementors MAY implement Permanent Failover and MAY enable it based on network configurations or users' requests.

6. Socket API Considerations

This section describes how the socket API defined in [RFC6458] is extended to provide a way for the application to control and observe the quick failover behavior.

Please note that this section is informational only.

A socket API implementation based on [RFC6458] is, by means of the existing SCTP_PEER_ADDR_CHANGE event, extended to provide the event notification when a peer address enters or leaves the potentially failed state as well as the socket API implementation is extended to expose the potentially failed state of a peer address in the existing SCTP_GET_PEER_ADDR_INFO structure.

Furthermore, two new read/write socket options for the level IPPROTO_SCTP and the name SCTP_PEER_ADDR_THLDS and SCTP_EXPOSE_POTENTIALLY_FAILED_STATE are defined as described below. The first socket option is used to control the values of the PFMR and PSMR parameters described in Section 5. The second one controls the exposition of the potentially failed path state.

Support for the SCTP_PEER_ADDR_THLDS and SCTP_EXPOSE_POTENTIALLY_FAILED_STATE socket options need also to be added to the function sctp_opt_info().

6.1. Support for the Potentially Failed Path State

As defined in [RFC6458], the SCTP_PEER_ADDR_CHANGE event is provided if the status of a peer address changes. In addition to the state changes described in [RFC6458], this event is also provided, if a peer address enters or leaves the potentially failed state. The notification as defined in [RFC6458] uses the following structure:

```
struct sctp_paddr_change {
  uint16_t spc_type;
  uint16_t spc_flags;
  uint32_t spc_length;
  struct sockaddr_storage spc_aaddr;
  uint32_t spc_state;
  uint32_t spc_error;
  sctp_assoc_t spc_assoc_id;
}
```

[RFC6458] defines the constants SCTP_ADDR_AVAILABLE, SCTP_ADDR_UNREACHABLE, SCTP_ADDR_REMOVED, SCTP_ADDR_ADDED, and SCTP_ADDR_MADE_PRIM to be provided in the spc_state field. This document defines in addition to that the new constant

SCTP_ADDR_POTENTIALLY_FAILED, which is reported if the affected address becomes potentially failed.

The SCTP_GET_PEER_ADDR_INFO socket option defined in [RFC6458] can be used to query the state of a peer address. It uses the following structure:

```
struct sctp_paddrinfo {
  sctp_assoc_t spinfo_assoc_id;
  struct sockaddr_storage spinfo_address;
  int32_t spinfo_state;
  uint32_t spinfo_cwnd;
  uint32_t spinfo_srtt;
  uint32_t spinfo_rto;
  uint32_t spinfo_mtu;
};
```

[RFC6458] defines the constants SCTP_UNCONFIRMED, SCTP_ACTIVE, and SCTP_INACTIVE to be provided in the spinfo_state field. This document defines in addition to that the new constant SCTP_POTENTIALLY_FAILED, which is reported if the peer address is potentially failed.

6.2. Peer Address Thresholds (SCTP_PEER_ADDR_THLDS) Socket Option

Applications can control the quick failover behavior by getting or setting the number of consecutive timeouts before a peer address is considered potentially failed or unreachable and before the primary path is changed automatically. This socket option uses the level IPPROTO_SCTP and the name SCTP_PEER_ADDR_THLDS.

The following structure is used to access and modify the thresholds:

```
struct sctp_paddrthlds {
  sctp_assoc_t spt_assoc_id;
  struct sockaddr_storage spt_address;
  uint16_t spt_pathmaxrxt;
  uint16_t spt_pathpfthld;
  uint16_t spt_pathcpthld;
};
```

spt_assoc_id: This parameter is ignored for one-to-one style sockets. For one-to-many style sockets the application may fill in an association identifier or SCTP_FUTURE_ASSOC. It is an error to use SCTP_{CURRENT|ALL}_ASSOC in spt_assoc_id.

spt_address: This specifies which peer address is of interest. If a wildcard address is provided, this socket option applies to all current and future peer addresses.

spt_pathmaxrxt: Each peer address of interest is considered unreachable, if its path error counter exceeds spt_pathmaxrxt.

spt_pathpfthld: Each peer address of interest is considered potentially failed, if its path error counter exceeds spt_pathpfthld.

spt_pathcpthld: Each peer address of interest is not considered the primary remote address anymore, if its path error counter exceeds spt_pathcpthld. Using a value of 0xffff disables the selection of a new primary peer address. If an implementation does not support the automatically selection of a new primary address, it should indicate an error with errno set to EINVAL if a value different from 0xffff is used in spt_pathcpthld.

6.3. Exposing the Potentially Failed Path State (SCTP_EXPOSE_POTENTIALLY_FAILED_STATE) Socket Option

Applications can control the exposure of the potentially failed path state in the SCTP_PEER_ADDR_CHANGE event and the SCTP_GET_PEER_ADDR_INFO as described in Section 6.1. The default value is implementation specific.

This socket option uses the level IPPROTO_SCTP and the name SCTP_EXPOSE_POTENTIALLY_FAILED_STATE.

The following structure is used to control the exposition of the potentially failed path state:

```
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 the application may fill in an association identifier or SCTP_FUTURE_ASSOC. It is an error to use SCTP_{CURRENT|ALL}_ASSOC in assoc_id.

assoc_value: The potentially failed path state is exposed if and only if this parameter is non-zero.

7. Security Considerations

There are no new security considerations introduced in this document.

8. IANA Considerations

This document does not create any new registries or modify the rules for any existing registries managed by IANA.

9. References

9.1. 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.

9.2. Informative References

[CARO02] Caro Jr., A., Iyengar, J., Amer, P., Heinz, G., and R. Stewart, "A Two-level Threshold Recovery Mechanism for SCTP", Tech report, CIS Dept, University of Delaware , 7 2002.

[CARO04] Caro Jr., A., Amer, P., and R. Stewart, "End-to-End Failover Thresholds for Transport Layer Multihoming", MILCOM 2004 , 11 2004.

[CARO05] Caro Jr., A., "End-to-End Fault Tolerance using Transport Layer Multihoming", Ph.D Thesis, University of Delaware , 1 2005.

[FALLON08] Fallon, S., Jacob, P., Qiao, Y., Murphy, L., Fallon, E., and A. Hanley, "SCTP Switchover Performance Issues in WLAN Environments", IEEE CCNC 2008, 1 2008.

[GRINNEMO04] Grinnemo, K-J. and A. Brunstrom, "Performance of SCTP-controlled failovers in M3UA-based SIGTRAN networks", Advanced Simulation Technologies Conference , 4 2004.

[IYENGAR06] Iyengar, J., Amer, P., and R. Stewart, "Concurrent Multipath Transfer using SCTP Multihoming over Independent End-to-end Paths.", IEEE/ACM Trans on Networking 14(5), 10 2006.

[JUNGMAIER02] Jungmaier, A., Rathgeb, E., and M. Tuexen, "On the use of SCTP in failover scenarios", World Multiconference on Systemics, Cybernetics and Informatics , 7 2002.

[NATARAJAN09] Natarajan, P., Ekiz, N., Amer, P., and R. Stewart,

"Concurrent Multipath Transfer during Path Failure", Computer Communications , 5 2009.

[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.

Authors' Addresses

Yoshifumi Nishida
GE Global Research
2623 Camino Ramon
San Ramon, CA 94583
USA

Email: nishida@wide.ad.jp

Preethi Natarajan
Cisco Systems
510 McCarthy Blvd
Milpitas, CA 95035
USA

Email: prenatar@cisco.com

Armando Caro
BBN Technologies
10 Moulton St.
Cambridge, MA 02138
USA

Email: acaro@bbn.com

Paul D. Amer
University of Delaware
Computer Science Department - 434 Smith Hall
Newark, DE 19716-2586
USA

Email: amer@udel.edu

Karen E. E. Nielsen
Ericsson
Kistavaegen 25
Stockholm, 164 80
Sweden

Email: karen.nielsen@tieto.com