Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: August 26, 2010

Y. Nishida
WIDE Project
Natarajan
Cisco Systems

February 22, 2010

# Quick Failover Algorithm in SCTP
# draft-nishida-natarajan-sctp-failover-00

Abstract

One of the major advantages in SCTP is supporting multi-homing communication. If an multi-homed end-point has redundant network connections, sctp sessions can have a good chance to survive from network failures by migrating inactive network to active one. However, if we follow the SCTP standard, there can be significant delay for the network migration. During this migration period, SCTP cannot transmit much data to the destination. This issue drastically impairs the usability of SCTP in some situations. This memo describes the issue of SCTP failover mechanism and discuss its solutions which require minimal modification to the current standard.

Status of this Memo

Copyright Notice

## Table of Contents

## 1. Introduction

Multihoming support is one of the major advantage of SCTP which is not supported in other transport protocols such as TCP or UDP. If an multi-homed end-point has redundant network interfaces, SCTP sessions can survive from the network failures by migrating inactive path to active one. This feature can be expected to be a driving force for deploying SCTP, however, because of minor issues in the SCTP specification, most of SCTP sessions will have significant delay to failover and will cause significant performance degradation during the failover process. We believe this issue is impairing the usability of SCTP and it is important to address it to make SCTP more efficient and attractive.

In this memo, we describe the issue of SCTP failover process and discuss the solutions. Our main focus is to propose a solution that does not require major modification to the current standard. Using Concurrent Multipath Transfer (CMT) [IYENGAR06] allows SCTP to utilize multiple paths simultaneously for data transmission. While CMT can reduce the impact of path failures, CMT is not yet a standard. In addition, some may not want concurrent data transfer feature, but want to use smooth failover feature in SCTP. From this reason, we believe the proposals in this document can be useful and meaningful.

## 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].

Since this document describes a potential risk in NewReno, it uses the same terminology and definitions in RFC4690. [RFC4690].

## 3. Issue in SCTP Path Management Process

SCTP can utilize multiple IP addresses for single SCTP association. Each SCTP endpoint exchanges the list of available addresses on the node during initial negotiation. After this, endpoints select one address from the list and define this as the destination of the primary path. Basically, SCTP sends all data through this primary path for normal data transmissions. Also, it sends heartbeat packets to other (non-primary) destinations at a certain interval to check the reachability of the path.

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

When sender receives the acknowledgment for data or heartbeat packets from one of the destination addresses, it considers the destination is 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 is inactive.

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

An issue in 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.

## 4. Solutions for Smooth Failover

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

### 4.1. Reduce Path.Max.Retrans

If we choose smaller value for Path.Max.Retrans, we can shorten the duration of failover process. In fact, this is recommended in some research results [JUNGMAIER02] [GRINNEMO04] [FALLON08]. For example, if we set Path.Max.Retrans to 0, SCTP switches to another destination on a single timeout. However, smaller value for Path.Max.Retrans might cause spurious failover. In addition, if we use smaller value for Path.Max.Retrans, we may also need to choose smaller value for 'Association.Max.Retrans'. The Association.Max.Retrans indicates the threshold for the total number of consecutive error count for the entire SCTP association. If the total of the error count for all paths exceeds this value, the endpoint considers the peer endpoint unreachable and terminates the association. According to the Section 8.2 in RFC4960, we should avoid having the value of Association.Max.Retrans larger than the summation of the Path.Max.Retrans of all the destination addresses. Otherwise, even if all the destination addresses become inactive, the endpoint still considers the peer endpoint reachable. The behavior in this situation is not defined in the RFC and depends on each implementation. In order to avoid inconsistent behavior between implementations, we had better use smaller value for Association.Max.Retrans. However, if we choose smaller value for Association.Max.Retrans, associations will prone to be terminated with minor congestion.

Another issue is that the interval of heartbeat packet: 'HB.interval' may not be small. (recommended value is 30 seconds) This means once failover takes place, an endpoint might need a certain amount of time to use the primary path again. This can cause undesirable effects in case of spurious failover. If we choose smaller value for HB.interval, the traffic used for path probing in a session will be increased.

The advantage of tuning Path.Max.Retrans is that it requires no modification to the current standard, although it needs to ignore several recommendations. 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 the section 5.

## 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 standard, although it needs to ignore several recommendations. 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.

## 4.3. Introduce Potential Failure Status in Failure Detection Algorithm

As seen above, one difficulty of tuning Path.Max.Retrans is that it is required to meet the following two inconsistent requirements.

- o In order to respond network failure quickly, we need to mark a path as inactive as soon as we detect failure.

- o In order to make an association persistent and robust against network failure, we need to be conservative to mark a path as inactive.

To satisfy these requirements, we propose to introduce "Potentially-failed" (PF) destination state in failure detection algorithm in SCTP. PF state is the intermediate state between Active and Inactive. It indicates that the path is possibly inactive, but not confirmed yet. By using the PF state, SCTP can respond to network failures quickly, while preserving a conservative policy of marking path as inactive. The idea of using PF state was originally proposed in [NATARAJAN08] for CMT.

In this algorithm, when sender receives the acknowledgment for data or heartbeat packets from one of the destination addresses, it considers the destination is Active. If it fails to receive acknowledgments, SCTP endpoint increment the error count for the path and transitions the destination to the PF state. (we might need to have new threshold value for error counter to be conservative to migrate from Active to PF. But, we choose this way for now)

If the primary path is marked PF, SCTP chooses new destination address from one of the active destinations and starts using this

address to send data. SCTP endpoints should not send any data packet to destinations in the PF state, however, it can send heartbeat packets at a certain interval. To allow quick recover from the PF state, we also propose to introduce a new protocol parameter 'PFHB.Interval'. PFHB.interval is used to determine the interval of heartbeat packets. It is recommended that a heartbeat packet is sent once per RTO of each destination address plus PFHB.interval with jittering of +/- 50% of the RTO value. (Preethi: wondering why we need jittering?) It is also recommended to use relatively smaller value than HB.interval for PFHB.interval.

If the heartbeat is answered, SCTP marks the path Active again. If unanswered, SCTP increments the error count and use an exponential backoff algorithm to increase the RTO. If the error count exceeds Path.Max.Retrans, the path is marked as Inactive. If all destinations are marked PF, SCTP endpoint can choose one destination to send data to its peer. How SCTP chooses a path is implementation specific. One possibility is to select the destination with the least error count. Once a PF destination is chosen for data transmission, the chosen destination must be transitioned from PF to the Active state. Except the use of PFHB.interval, other rules of sending heartbeats are completely the same as those of the standard.

The advantage of this approach is that we can keep the same values for Path.Max.Retrans, Association.Max.Retrans and HB.interval used in the current implementations, while it can respond network failure quickly. In addition, new transmission algorithm becomes effective only when the path is in the PF state. When the primary path is in Active or Inactive, the behavior is completely the same as that of the current standard. When the failure detection threshold is most aggressive (PMR=0), both SCTP and SCTP-PF detect path failure after the first timeout. Specifically, SCTP-PF's failure detection does not involve the PF state transition and is equivalent to SCTP's failure detection procedure. In other words, when PMR=0, both SCTP and SCTP-PF perform similarly during path failure. As PMR increases, SCTP's failure detection takes longer and the performance difference between SCTP and SCTP-PF widens (SCTP-PF performs better)."

## 5. Discussion

### 5.1. Effect of Path Bouncing

The methods described above can accelerate failover process. Hence, it might introduce path bouncing effect which keeps changing the data transmission path frequently. This sounds harmful for 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 to another path, it starts with minimal cwnd because of slow-start. Hence, there is little chance for packet reordering or duplicating.

Second, even if all communication paths between end-nodes share the same bottleneck, the proposed method does not make situations worse. In case of congestion, the current standard tries to transmit data packets to the primary during failover, while the proposed method tries to explore other destinations. In any case, the same amount of data packets sent to the same bottleneck.

### 5.2. Permanent Failover

When primary path becomes active again after failover, SCTP migrates back to the primary path. After this, SCTP starts data transfer with minimal cwnd. This is because SCTP must perform slow-start when it migrates to new path. However, this might degrade the communication performance in case that the performance of the alternative path is relatively good. In order to mitigate this effect of slow-start, permanent failover was proposed in [CARO02]. Permanent failover allows SCTP to remain the alternative path even if the primacy path becomes active again. This approach can improve performance in some cases, however, it will require more detail analysis since it might impact on SCTP failover algorithm. Since we prefer to keep the current behavior of the standard as possible, we recommend not to take this approach for now.

## 6. Security Considerations

There are no new security considerations introduced in this document.

## 7. IANA Considerations

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

## 8. Normative 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, "Peformance 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 scenrarios", World Multiconference on Systemics, Cybernetics and Informatics , 7 2002.

[NATARAJAN08] Natarajan, P., Ekiz, N., Iyengar, J., Amer, P., and R. Stewart, "Concurrent Multipath Transfer using SCTP Multihoming: Introducing Potentially-failed Destination State", IFIP Networking , 5 2008.

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

[RFC4690] Klensin, J., Faltstrom, P., Karp, C., and IAB, "Review and Recommendations for Internationalized Domain Names (IDNs)", RFC 4690, September 2006.

## Authors' Addresses

Yoshifumi Nishida
WIDE Project
Endo 5322
Fujisawa, Kanagawa 252-8520
Japan

Email: nishida@wide.ad.jp

Preethi Natarajan
Cisco Systems
425 E. Tasman Drive
San Jose, CA 95134
USA

Email: prenatar@cisco.com