SCTP’s Reliability and Fault Tolerance

Brad Penoff, Mike Tsai, and Alan Wagner
Department of Computer Science
University of British Columbia
Vancouver, Canada

Seattle Conference on Scalability
June 23, 2007
What is SCTP?

Stream Control Transmission Protocol
RFC 2960

An IETF standardized transport protocol for IP
• Can be used in the **same way** TCP or UDP is used.

• A **relatively new** transport protocol for **reliable** message-oriented data transfer.

• The **only new reliable** IETF standardized transport protocol introduced in the last **30 years**.
Why should I care about SCTP?

• New features

  → New Opportunities
  → like added robustness

  → New Challenges
  → like multi-homing
Cast of characters

• **UBC SCTP Project** (Alan Wagner)
  – Brad Penoff,
  – Mike Tsai,
  – Karol Mroz,
  – Humaira Kamal

• **Collaborators:**
  – **Randall Stewart** (Cisco) – co-inventor, FreeBSD
  – **Peter Lei** (Cisco) – FreeBSD/Mac OS X
  – **Michael Tüxen** (U. Münster) – Mac OS X/Testing
  – **Janardhan Iyengar** (Ph.D., U. of Delaware)
What are our interests?

• SCTP-based MPI (Message Passing Interface) middleware for parallel processing
  – Implemented MPICH2 SCTP channel
  – Implementing a module for Open MPI

• Testing and Evaluating SCTP-based MPI programs.
  – e.g., task farming matrix correlation, bioinformatics search

• Possible advantages of SCTP for commodity clusters. (scalability → fault tolerance)
Talk Goals

• SCTP features

• Reliable task farming
SCTP Features
SCTP is like TCP

• Connection-oriented between two endpoints using unicast addresses
• TCP-like congestion control (RFC2581)
• Reliable, in-order delivery
• Flow controlled
• Available on most major operating systems
SCTP is *not* like TCP

1. Multi-homing
2. Message-based
3. Multi-streaming
4. Selective ACK (SACK) built-in
5. Stronger checksum
6. Sockets API Choice
Feature 1. Multi-homing
Feature 1. Multi-homing

Use #1 - Retransmissions

Endpoint X

NIC1

NIC2

IP=207.10.3.20

Network 207.10.x.x

IP=168.1.10.30

Network 168.1.x.x

Endpoint Y

NIC3

NIC4

IP=207.10.40.1

IP=168.1.140.10
Feature 1. Multi-homing

Use #1 - Retransmissions
Feature 1. Multi-homing

Use #1 - Retransmissions
Feature 1. Multi-homing

Use #1 - Retransmissions
Feature 1. Multi-homing

Use #1 - Retransmissions

Endpoint X

NIC1
NIC2

Endpoint Y

NIC3
NIC4

Association
Feature 1. Multi-homing

Use #1 - Retransmissions
Feature 1. Multi-homing

Use #1 - Retransmissions

Endpoint X

NIC1

NIC2

Endpoint Y

NIC3

NIC4

Association
Feature 1. Multi-homing

Use #1 - Retransmissions

Retransmissions occur on alternate path.
Feature 1. Multi-homing

Use #1 - Retransmissions

Retransmissions occur on alternate path.
Feature 1. Multi-homing

Use #1 - Retransmissions

Retransmissions occur on alternate path.
Feature 1. Multi-homing

Use #1 - Retransmissions

Retransmissions occur on alternate path.
Feature 1. Multi-homing

Use #1 - Retransmissions

Retransmissions occur on alternate path.
Feature 1. Multi-homing

Use #1 - Retransmissions

Retransmissions occur on alternate path.
Feature 1. Multi-homing

Use #2 – Path redundancy and failover

Path failure detected so failover occurs.
Feature 1. Multi-homing

Use #3 – Concurrent Multipath Transfer (CMT)*

* = See Janardhan Iyengar’s PhD dissertation
Feature 2. Message-based
Feature 2. Message-based
Feature 2. Message-based
Feature 2. Message-based
Feature 2. Message-based
Feature 2. Message-based
Feature 2. Message-based
Feature 2. Message-based
Feature 2. Message-based
Feature 2. Message-based
Feature 2. Message-based

TCP example
Feature 2. Message-based

TCP example

Endpoint X

RECEIVER

read()

Msg A

Msg B

Msg C

Endpoint Y

SENDER
Feature 2. Message-based

TCP example
Feature 2. Message-based

TCP example

Endpoint X
RECEIVER

Msg A

Msg

Msg C

B

Endpoint Y
SENDER

No Boundary between A & B
Feature 2. Message-based

TCP example

Message framing must occur within TCP applications.
Feature 2. Message-based

SCTP example

Endpoint X
RECEIVER
Msg A
Msg B
Msg C

Endpoint Y
SENDER
Feature 2. Message-based

SCTP example
Feature 2. Message-based

SCTP example

Endpoint X
RECEIVER
Msg A
Msg B
Msg C

Endpoint Y
SENDER
Feature 2. Message-based

SCTP example

Endpoint X

RECEIVER

read()

Msg A

Msg B

Msg C

Endpoint Y

SENDER
Feature 2. Message-based

SCTP example

Endpoint X

RECEIVER

Msg A

Msg B

Msg C

Endpoint Y

SENDER
Feature 2. Message-based

SCTP example

Endpoint X

RECEIVER

read()

Msg A

Msg B

Msg C

Endpoint Y

SENDER
Feature 2. Message-based

SCTP example

Endpoint X

RECEIVER

Endpoint Y

SENDER

Msg A

Msg B

Msg C
Feature 2. Message-based

SCTP example

SAME Boundaries as on Sender

SCTP applications do not need to do message framing.
Feature 3. Multi-streaming

Problem: Head-of-Line Blocking in TCP
Feature 3. Multi-streaming

Problem: Head-of-Line Blocking in TCP
Feature 3. Multi-streaming

Problem: Head-of-Line Blocking in TCP
Feature 3. Multi-streaming

Problem: Head-of-Line Blocking in TCP
Feature 3. Multi-streaming

Problem: Head-of-Line Blocking in TCP
Feature 3. Multi-streaming

Problem: Head-of-Line Blocking in TCP

Head-of-Line Blocking: Must wait for Msg A to be retransmitted before B & C are delivered
Feature 3. Multi-streaming

Solution: Multiple Streams in an Association

Endpoint X

SEND
Stream 1
Stream 2

RECEIVE

Endpoint Y

SEND
Stream 1
Stream 2
Stream 3

Outbound Streams

Inbound Streams
Feature 3. Multi-streaming

Send order

Endpoint X

Endpoint Y

SEND

Stream 1

Stream 2

Stream 3

Msg A  Msg B  Msg C  Msg D  Msg E

Msg A  Msg B  Msg C  Msg D  Msg E
Feature 3. Multi-streaming

Send order

Endpoint X

Endpoint Y

Stream 1
Stream 2
Stream 3
Feature 3. Multi-streaming

Send order

Msg C  Msg D  Msg E

Endpoint X
RECEIVE

Msg A  Msg B

Endpoint Y
SEND

Stream 1
Stream 2
Stream 3
Feature 3. Multi-streaming

Send order

Endpoint X

RECEIVE

Msg A

Msg B

Msg C

Endpoint Y

SEND

Stream 1

Stream 2

Stream 3

Msg D

Msg E
Feature 3. Multi-streaming

Send order

Endpoint X

RECEIVE

Msg A  Msg C  Msg D

Msg B

Endpoint Y

SEND

Stream 1

Stream 2

Stream 3

Msg E
Feature 3. Multi-streaming

Send order

Endpoint X

RECEIVE

Msg A  Msg C  Msg D

Msg B

Endpoint Y

SEND

Stream 1

Stream 2

Stream 3

Msg E
Feature 3. Multi-streaming

Receive order

Endpoint X

RECEIVE

Msg A
Msg C
Msg D

Msg B

Endpoint Y

SEND

Stream 1
Stream 2
Stream 3

Msg E
Feature 3. Multi-streaming

Receive order

No Head-of-Line Blocking:
Msg E could be delivered to the application since it is on a different stream.
Feature 3. Multi-streaming

Receive order

Msg E
Msg A

Msg D
Msg C
Msg B

Endpoint X
RECEIVE

Endpoint Y
SEND
Stream 1
Stream 2
Stream 3
Feature 3. Multi-streaming

Receive order

Endpoint X

RECEIVE

Msg D

Msg C

Endpoint Y

SEND

Stream 1

Stream 2

Stream 3
Feature 3. Multi-streaming

Order within a stream maintained.
Feature 4. Selective Acknowledgment
Feature 4. Selective Acknowledgment
Feature 4. Selective Acknowledgment

- SACK describes exactly what was lost.
Feature 4. Selective Acknowledgment

- SACK reduces number of retransmissions.

Msg C needs resent if using cumulative ACKs
Feature 4. Selective Acknowledgment

- SACK is built-in to SCTP.
- SCTP SACK can express more gaps than TCP.
Feature 5. Stronger Checksum

- End-to-end reliability
- TCP - additive 16-bit checksum (RFC 1071)
  - 1 in $1 \times 10^7$ packets arrives corrupted & accepted as valid (router bugs) **
- SCTP – uses 32-bit CRC32c (RFC 3309)
  - More computationally intensive
  - Currently not offloaded on most NICs
    - Available on Intel I/O procs (315, 331, 332) for iSCSI
- Cost of reliability

Feature 6. Sockets API Choice

• Support for standard sockets API
  – “TCP-like” one-to-one socket
  – “UDP-like” one-to-many socket
  – Simplifies porting existing applications

• Extended sockets API
  – To utilize new features
    • Multiple streams
    • Socket options to tune various SCTP parameters
SCTP features not covered

- Improved security features (cookie, v-tag).
- Path MTU discovery built-in.
- Provision for future protocol extensibility.
- API provides event notifications
- **Partial reliability** (RFC 3758) allows users to set an SCTP-specific TTL.
- MSG_UNORDERED can be used by applications for signaling.
Summary

1. Multi-homing
2. Message-based
3. Multi-streaming
4. Selective ACK (SACK) built-in
5. Stronger checksum
6. Sockets API Choice
Experience in using SCTP for task-farming
Task farming

- Common strategy for work distribution
- Execute a single function $F$ over $N$ independent data items
  - $F (d_0, d_1, d_2, \ldots, d_{N-1})$ executed by the system
    - Different data $d_i$ executed on by different workers
Task Farming Issues

- Keep everything busy

- Latency tolerance
  - Buffering
  - Overlapping communication and computation
SCTP Features

1. Multi-homing
2. Message-based
3. Multi-streaming
4. Selective ACK (SACK) built-in
5. Stronger checksum
6. Sockets API Choice
Feature

2. message-based

6. one-to-many style

Message-based is more natural

Fewer sockets to manage
3. multi-streaming
4. selective acks

Reduces potential delay

SACK reduces delay, any messages
- Adds additional reliability
  - Fault tolerance by path fail-over
  - Stronger message reliability
- All automatic!
• MapReduce
  – Better congestion avoidance
  – Heart-beat to the workers
    • SCTP notification of fail-over or loss of connection
  – Better tuning of task buffers, less jitter
    • H. Kamal, B. Penoff, M. Tsai, E. Vong, and A. Wagner., “Using SCTP to hide latency in MPI programs”. In HCW: IPDPS, April 2006
Conclusion

• SCTP is available on most major OSes (FreeBSD, Mac OS X, Linux, Solaris 10, etc)

• For more information, you can visit:
  – SCTP websites:
    • www.sctp.org
    • www.sctp.de
      • Papers and projects

• Download and try it out!
Thank you!

Google “sctp mpi” for more information about our work