SCTP Tutorial

Randall Stewart (randall@lakerest.net)
Michael Tüxen (tuexen@fh-muenster.de)
Outline

• Overview
• Services provided by SCTP
• Deployment considerations
• Current developments
Timeline of Transport Protocols

- UDP (RFC 768, August 1980)
- TCP (RFC 793, September 1981)
- SCTP (RFC 2960, October 2000)
- UDP-Lite (RFC 3828, July 2004)
- DCCP (RFC 4340, March 2006)
- MP-TCP (RFC 6824, January 2013)
Timeline of SCTP RFCs

• Core Protocol
  – Initial Base Specification (RFC 2960, October 2000)
  – Checksum Change (RFC 3309, September 2002)
  – Errata and Issues (RFC 4460, April 2006)
  – Updated Base Specification (RFC 4960, September 2007)

• Protocol Extensions
  – Partial Reliability (RFC 3758, May 2004)
  – Chunk Authentication (RFC 4895, August 2007)
  – Address Reconfiguration (RFC 5061, September 2007)
  – Stream Reconfiguration (RFC 6525, February 2012)

• API
  – Socket API (RFC 6458, December 2011)
Protocol Overview

• Connection oriented (SCTP association)
• Supports unicast
• Same port number concept as other transport protocols
• Message oriented
  – Supports arbitrary large messages (fragmentation and reassembly)
  – Supports bundling of multiple small messages in one SCTP packet
  – Flexible ordering and reliability
• Supports multihoming using IPv4 and IPv6
• Packet consists of a common header followed by chunks
• Extendable
Association Setup

• Four way handshake
• Resistance against “SYN flooding”
• Negotiates
  – Initial number of streams
  – Initial set of IP addresses
  – Supported extensions
• User messages can already be transmitted on the third leg (after one RTT i.e. same as TCP)
• Handles the case of both sides initiating the association.
Data Transfer

- TCP friendly congestion control
- User messages are put into DATA chunks (possibly multiple in case of fragmentation)
- Each DATA chunk is identified by a Transmission Sequence Number (TSN)
- Acknowledgements (SACKs) reporting
  - Cumulative TSN
  - Gaps (up to approximately 300 in a sack)
  - Duplicate TSNs
- Retransmissions
  - Based on timer
  - Based on gap reports
Association Teardown

• Graceful shutdown
  – Teardown without message loss.
  – Based on an exchange of three messages.
  – Supervised by timer
  – No half close state is allowed

• Non-graceful shutdown
  – Possibly message loss
  – Uses a single message
Service: Preservation of Message Boundaries

• Most application protocols are message based
• Simplifies application protocols and its implementation
• Awareness of message boundaries makes optimal handling at the transport layer / application layer boundary possible
• But special attention is needed for supporting arbitrary large messages
Service: Partial Reliability

• Allows to avoid spending resources on user messages not being relevant anymore for the receiver.
• The sender can abandon user messages base on criteria called PR-SCTP policy
• PR-SCTP policies are implemented on the sender side and does not require negotiation.
• Examples of PR-SCTP policies:
  – Lifetime
  – Number of retransmissions
  – Priority with respect to buffering
Service: Partial Ordering

• An SCTP association provides up to $2^{16}$ uni-directional streams in each direction.
• The application is free to send a message on a stream of its choice.
• Minimizes head of line blocking, because message ordering is only preserved within each stream.
• In addition, messages can be marked for unordered delivery.
• The stream reconfiguration extension (RFC 6525) allows to
  – Add streams during the lifetime of an association
  – Reset streams (i.e. start over at stream sequence 0)
Service: Network Fault Tolerance

• Each end-point can have multiple IP-addresses
• Each path is continuously supervised
• Primary path is used for initial transmission of user data
• In case of a failure, another (working) address is used
• The Address Reconfiguration extension (RFC 5061) allows
  – Add and delete IP-addresses during the lifetime of an association
  – Select the local and remote primary path
• Currently being specified: loadsharing
Security

• SCTP over IPSec
  – Specified in RFC 3554, July 2003
  – Multihoming improvements for IPSec
  – Not implemented (as far as the authors know)

• TLS over SCTP
  – Specified in RFC 3436, December 2002
  – Doesn’t provide all services (no PR-SCTP, only ordered delivery)
  – Doesn’t scale well and can’t be implemented directly in OpenSSL, however can be build as part of the application

• DTLS over SCTP
  – Specified in RFC 6083, September 2010
  – Provides almost all services provided by SCTP and its extensions
  – Implemented in OpenSSL 1.0.1
Usage

- SIGTRAN: Telephony signaling networks
- RSerPool
- Diameter
- IPFIX
- Forces
- RTCWeb
RTCWeb

• Transport layer for data channels
• Encapsulated in DTLS running on top of UDP using ICE/STUN/TURN for NAT traversal
• Usage of
  – multiple streams
  – ordered / unordered delivery
  – partial reliability
  – stream reconfiguration
Implementations

• Provided by OS vendor for
  – FreeBSD
  – Linux
  – Solaris

• The FreeBSD has been ported to support
  – Mac OS X as a network kernel extension (NKE)
  – Windows as a kernel driver
  – Windows, Linux, FreeBSD, MacOS X as a userland stack (included in Firefox)

• Commercial implementations for various operating systems

• Implementations are interoperable as shown in nine interoperability tests.
Socket API (RFC 6458)

- Two programming models:
  - One to one Style API
  - One to many Style API
- Several socket options allowing fine-tuning of parameters
- Notifications (events that happen on the transport connection)
- Additional cmsgs for sendmsg()/recvmsg()
- Additional functions for
  - supporting multiple IP addresses per end-point
  - sending and receiving user messages
  - Transition of sockets between programming models
- Mostly supported by FreeBSD, Linux and Solaris allowing users to write portable programs
NAT Traversal

• Legacy NATs:
  – UDP encapsulation, allows UDP port numbers to be modified by middle-boxes
  – Requires support in the SCTP end-hosts
  – Doesn’t require special support in the middle-boxes

• SCTP aware NATs:
  – SCTP port numbers are not modified by middle-boxes
  – Requires support from the middle-boxes and the end-hosts, however no communication between middle-boxes is required
Ongoing SCTP-related Work in TSVWG

- UDP tunneling (in IESG discussion)
- SCTP aware NATs
- ECN support
- Interleaving of user messages
- Loadsharing
- Optimizations (sack immediately and others)
Conclusion

• SCTP provides a variety of flexible services
  – Network fault tolerance
  – Partial reliability
  – Partial ordering
• Interoperable implementations are available
• Middleboxes need to be taken into account