Stream Control Transmission Protocol and its use in DataCenter Protocol Stack

Lovish

April 20, 2013

Introduction

Since its proposal in 1974 TCP had come a long way, both in terms of protocol design and development related optimizations. TCP (and its variants) are still used in many scenarios, contrasting in their basic purpose, be it Wireless Networks or Datacenters.

SCTP was introduced in 2000 as a signalling layer protocol to be used for transferring call control signals on IP networks. Meanwhile, other protocols were introduced for the same purpose such as Diameter and RSerPool (Reliable Server Spooling). From then, SCTP had focused more on being a protocol with design principles based on valuable features of both TCP and UDP.

Motivation

Many of the RFC's related to SCTP are still in experimental stage and many of the informational one's are constantly being updated or replaced by new ones. In that sense it seemed to be a very exciting and cutting-edge protocol which may be used as a replacement of TCP in near future.

Also, some of the work in the field of Datacenters NextGen (connected by 100Gbps Ethernet Connection) considers SCTP as a potential replacement for TCP in TCP/IP/Ethernet Stack.
Packet Structure

SCTP packet consists of a header, some SCTP control chunks and data chunks. These chunks are basic units of delivery containing a chunk header and some content specific to that particular chunk. A user message may be bundled in the same SCTP packet with other chunks occupying a particular data chunk.

Note: The control chunks INIT and INIT ACK cannot be bundled with other control or data chunks.

A basic DATA chunk looks like:

<table>
<thead>
<tr>
<th>Type=0x00</th>
<th>Flags=UBE</th>
<th>Length=variable</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>TSN Value</td>
</tr>
<tr>
<td></td>
<td>Stream Identifier</td>
<td>Stream Sequence Num</td>
</tr>
<tr>
<td></td>
<td>Payload Protocol Identifier</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Variable Length User Data</td>
<td></td>
</tr>
</tbody>
</table>

Figure 2: SCTP Data Chunk

U Bit represents if the data is unordered/ordered. There would be no stream Stream Identifier and Stream sequence number if its set to 1. B and E Bits represents if the given data chunk signifies the beginning or end of a user message. If both are set to one then the chunk is supposed to be the only chunk corresponding to the message.

Stream Identifier represents the stream number in which the data chunk belongs and stream sequence number determines the order of the chunk within the stream.

Also some of the control chunks are INIT, INIT-ACK, SACK, HEARTBEAT, COOKIE-ECHO, ASCONF, FORWARD-TSN. We will discuss these and the feature they provide in detail in the rest of the report.

Features

As seen in Figure 1, many of features are unique to SCTP. Let us discuss them in detail:

Security Enhancements

4-way Handshake

- Primary Purpose - Imparting statelessness to receiver side

SYN Flooding is a major loophole present in 3-way handshake. Thus a node can lead to excessive backlogs by filling the TCB of receiver and thus completely choking it from accepting new connections. This is countered by a 4-way handshake.

No half-open state can be created in a 4-way handshake. The main messages in it are:
- INIT -
- INIT-ACK - A cookie, is sent from server to client which is expected to be echoed. Cookie contains the following information:
  - COOKIE lifetime
  - COOKIE signature to verify it
  - Details to identify Note that here the server is not saving any state.
- COOKIE-ECHO - Client just echoes the cookie it receives from the server and piggybacks the data. Hence effectively the data transfer starts from here.
- COOKIE-ACK - The server then using the state information in cookie determines if it is genuine. Once confirmed the association is established.

Note :- COOKIE based mechanism only works if the Attacker can only send data from spoofed IP address and not receive the data sent to the Victim’s IP.

![4-way Handshake](image)

**Figure 3: 4-way Handshake**

### Validation

A **Verification Tag** is agreed upon at the start of every association. It is used as a defence shield against DOS attacks and stale SCTP packets from previous association. INIT packet has 0 verification tag.

Note that this and CRC checksum (which is also included in TCP) are two different things. Checksum is used to find and correct the erroneous bits in a packet instead.

### Use of Tie tags in association restart

Suppose that a node A goes down inbetween an association with Node Z. After recovery A would try to reconnect to Z but it would not have the verification tag. So at the time it will send a INIT to Z. Z knowing that A is **restarting** previous association (looking at the source IP address of packet) would send a **Tie-Tag** which can be considered as a new verification tag. A echos the same tag to Z and the association is restablished in a secure way. The whole process is depicted in figure 4.
Figure 4: Role of Tie Tags in Restarting Association
Multi-Streaming

Multi-streaming plays a major role in avoiding the HOL (Head of Line) blocking problem of TCP. Suppose we are transferring 3 images I1, I2, I3 (depicted in Figure 4) from A to B through Stream 0, 1, 2 respectively and the HOL data chunk in Stream 0 is lost/corrupted beyond recovery then it would not effect the transfer of I2 and I3. The chunk used for Configuring Multiple Streams is RE-CONFIG. The information contained may be of the following type:

- **Add/Reset Outgoing Streams**
- **Add/Reset Incoming Streams**
- **Reconfiguration Response** - The response of one peer to the requests of other peer for adding/resetting streams. The request parameters may contain the following responses:
  - Success
  - Failure
  - In Progress
  - Error
    * Sequence number from 1 to $2^{32}$ is out of range.
    * Sequence number is in range but the response sequence number is smaller than the sent one.

![Figure 5: Multi-Streaming avoiding HOL](image)

Multi-Homing

If a client is connected via 2 different ISP’s, SCTP makes the system fault tolerant by using a secondary IP if primary IP goes down due to some reason. Figure 4 shows a Multi-Homed host connected both via Ethernet and Wireless.

Dynamic Address REconfiguration

The aim is to dynamically add/delete/update the current primary address which makes a node apparent as a hot pluggable device. Contrast this to the situation of TCP where any change in IP or MAC layer would result in restart of the connection.

Two chunks take care of Address Reconfiguration in SCTP:

- **ASCONF** - Address Configuration Chunk is used primarily used for conveying the changes that are required by a node to its peer. Along with the essential sequence number, it contains ASCONF parameters such as
• Set Primary Address
• Add IP Address

The information in the chunk is laid out in the form of Type-Length-Value. So here is an example of Add-IP-Addr Chunk:-

![Diagram of Multi-Homing]

Figure 6: Multi-Homing

**Correlation ID** is a 32-bit value which is inserted into ASCONF-ACK by receiver so that sender can find out the request for which response has arrived.

• Delete IP Address
• Error Cause Indication - Error may occur due to the following :-
  – Request to delete the last remaining IP Address of itself
  – Resource Shortage at one side
  – Request to delete the the source IP address - Deleting the very same address from which the packet has arrived
  – No Authentication

**Partial Reliability**

**Security Issues**

Multihoming in SCTP makes it vulnerable to some different kind of attacks. Most of these are of the genre of DOS attack.
**Bombing Attack I** - As shown in the figure below, the attacker A gives the victim’s IP address (as secondary IP address) in the INIT packet while setting up the association. It then asks Server S for a large amount of data. When the data arrives on its primary IP address (i.e., the attacker’s address), it does not send an ACK, hence Server S is bound to assume that the primary address is down and sends data to the secondary IP (victim’s address). The attacker can send the ACKs at periodic intervals to continue the attack.

Note that for this attack to work, the victim should not support SCTP as it can send back an OOTB (Out-Of-The-Blue) packet to the server to stop receiving packets.

**Bombing Attack II** - First, the attacker sends malicious ICMP packets which increase the MTU towards the side of the victim. Hence, while sending a packet, the victim would need to segment it and send the chunks in more than one packet.

To avoid this attack, a SCTP packet is not allowed to send a large number of packets in response to a single packet.

![Bombing Attack Diagram](image)

Figure 7: Bombing Attack I

**Use in Datacenters**

Today DataCenters use the TCP/IP/Ethernet stack and specialized channels such as Infiniband. But the future of datacenter requires sending data at much lower latencies and high rate. SCTP is a potential candidate to replace TCP in a way that it is a much flexible and extensible protocol. All we need to do is to define the chunk as per our need. In [16] authors have compared TCP implementation to SCTP implementation and concluded that current implementation of SCTP needs much improvement for any use. They have also suggested some improvements like re-implementation of multi-streaming feature to increase concurrency and SACK being made optional due to high overhead.

In [17] the same authors have proposed (not tested) USTP (Unified Scaleout Transport Protocol) which they claim has adaptive flow/congestion control across various interfaces such as Ethernet, 802.11 or Optical, is power aware and is interoperable with existing transport layer protocols like TCP and UDP.
References


Figure References

Figure 1 and 2 have been taken from University of Delaware slides, link.