Introduction to the Stream Control Transmission Protocol (SCTP):

The next generation of the Transmission Control Protocol (TCP)

Introduction

We are guided through security scans at the airport, and then whisked away in a speeding caravan of vehicles. As we reach the semi-secret world of the Internet Engineering Task Force (IETF) working groups, we enter the glass doors that mark the point of no return. Ducking into a maze of hallways we come to a plain door. Lurking in the backroom, between the dark navy suits the Stream Control Transmission Protocol (SCTP) exists. Maneuvering for a closer view, we find AIX from IBM, BSD with three different variants, Cisco IOS, Linux, third-parties for the Microsoft operating system, and Sun Solaris. Each has implemented their interpretation of this standards-based protocol. We stare at what could be the Holy Grail for IP-based multimedia and telecommunications. Passing between the cubicles, we enter a door curiously labeled "Test/Laboratory area."

Within the confines of the laboratory, badges on the clean-room suits indicate that SCTP is the result of the IETF Signal Transmission (SIGTRAN) working groups' efforts. Their intent was to produce something akin to the telephone Signaling System 7 (SS7) switching network that was capable of traversing IP networks. Simply put, SCTP was built to carry call control signals using IP networks. SCTP is not new; it was created in 2000, and is taking some time to outgrow its confinement within the large telecommunications companies' interconnects.

Quietly, we are informed that the secrecy was created to prevent a mass rush to implement SCTP when everyone finds out about SCTP's capabilities. With more than twenty years of work under their belts, the Transmission Control Protocol (TCP) and User Datagram Protocol (UDP) were not singularly up to the task, but each provided parts and ideas to make SCTP possible. Neither TCP nor UDP can handle multi-homing, or the ability to send information to an alternate address if the primary becomes unreachable. SCTP's closest competition, TCP, will need to improve or become a relic.

Within the IP Multimedia Subsystem (IMS) specifications, SCTP provides connections between various Session Initiated Protocol (SIP) servers and proxy servers, collectively called Call Session Control Function (CSCF). The secrecy seems to reside around the fact that the SCTP communications that flow between or through the Proxy-CSCF, Interrogating-CSCF, Serving-CSCF, and other selected pieces of the IMS infrastructure are handled in the control-plane versus the data-plane. (The data-plane contains the actual telecommunications data being delivered to a subscriber/user, and the support-plane provides the control and connectivity for the data-plane.) Since the SCTP is hidden behind the curtains of the support-plane networks, it has not received the publicity or the focus of some of the other IMS specifications. SIP and RTSP have broken through the curtain by being able to process Voice over IP (VoIP) telecommunications within the enterprise space. But, SIP and RTSP's notoriety may be short lived when SCTP enters the enterprise space.

Now here is the crucial part: without SCTP's capabilities the IMS would not have the capability to reliably pass call control signaling to the various systems and it would not be possible to use TCP or UDP. You could use these if you only allow twenty simultaneous sessions, which is inconsequential as most Service Providers count their subscribers in millions. Prying open the specifications for SCTP reveals many features or services similar to TCP.

SCTP Features/Services

By looking at Table 1, you can easily see the benefits of SCTP over TCP and UDP. (See *Appendix 1* for an explanation of features/services.) Is it the next generation of TCP? It easily could be, as SCTP is powerful and, when implemented, can provide innumerable benefits to connection-oriented communications.

Feature/Service	SCTP	TCP	UDP
Allow half-closed connections	✗	✓	N/A
Application PDU bundling	✓	✓	✗
Application PDU fragmentation	✓	✓	✗
Congestion control	✓	✓	✗
Connection-oriented	✓	✓	✗
ECN capable	✓	✓	✗
Flow control	✓	✓	✗
Full duplex	✓	✓	✓
Multi-homing	✓	✗	✗
Multi-streaming	✓	✗	✗
Ordered data delivery	✓	✓	✗
Partial-reliable data transfer	Optional	x	✗
Path MTU discovery	✓	✓	✗
Preserve message boundaries	✓	✗	✓
Protect against SYN flooding attacks	✓	✗	N/A
Pseudo-header for checksum	Uses vtags	✓	✓
Reachability check	✓	✓	✗
Reliable data transfer	✓	✓	✗
Selective acknowledgements	✓	Optional	✗
Time wait state	For vtags	For 4-tuple	N/A
Unordered data delivery	✓	✗	✓

(✓ = yes, ✗ = no, and N/A = not applicable)

Table 1: *SCTP compared to TCP and UDP* [i]

SCTP Key Benefits

SCTP improves upon TCP and UDP by integrating components of each. But the designers of SCTP did not stop there. There have been two new concepts added: multi-homing and multi-streaming.

Multi-homing

SCTP was designed to handle the signaling of telecommunications over IP. Since telecommunications are very susceptible to time delays, every millisecond counts. Multi-homing enables systems that have multiple interfaces, for redundancy, to use one over the other without

having to wait. Within SCTP one interface is established as the primary and the rest become secondary. If the primary should fail for whatever reason, a secondary is selected and utilized. When the primary becomes available again, the communications can be transferred back without the application being aware there was an issue. While establishing the connections, the primary and secondary interfaces are checked and monitored using a heartbeat/heartbeat acknowledgement process that validates addresses, and maintains a Round Trip Time (RTT) calculation for each address. The RTT can indicate that the primary is slower than a secondary and allow for the communications to migrate to the secondary interface.

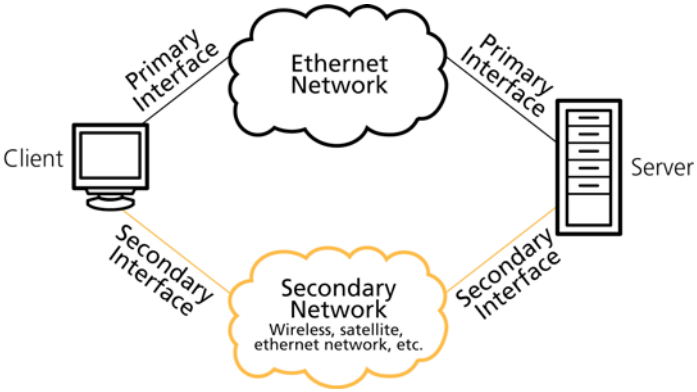


Figure 2: *Multi-homing*

Multi-streaming

Using TCP, only one single data stream is allowed per connection. All of the information must be passed through that one stream. SCTP allows multiple simultaneous data streams within a connection or association. Each message sent to a data stream can have a different final destination, but each must maintain message boundaries. For example, systems cannot send parts of the same message through different streams; one message must go through one stream. When running an ordered data delivery system, if one of the packets is out of order or missing, the stream is blocked pending resolution to the order. This is called “Head-of-Line Blocking.” With the use of multi-streams, only the stream that is affected would be blocked; the other streams would continue to flow.

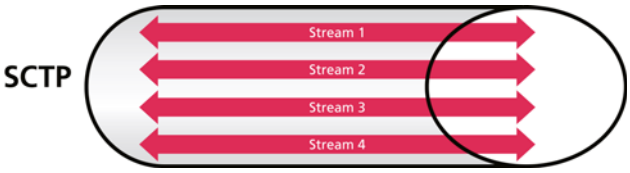


Figure 3: *Multi-streaming*

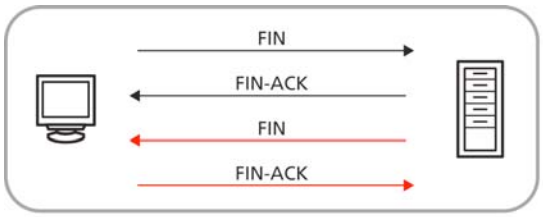
By using multi-streaming with SCTP, the issue with web browsers only having the ability to handle two simultaneous connections goes away. The client or the web server could immediately open additional streams and send pictures, text, etc. through each stream, reducing overall latency. This could also reduce overhead that servers often incur with the numerous separate connections required to fulfill a request.

SCTP’s multi-streaming would also help with infrastructures that connect multiple means of communications simultaneously. For example, if we look at Figure 3 above, Stream 1 could be voice communications, Stream 2 could be video, Stream 3 a shared application, and so on. By enabling SCTP to meet future needs, the SIGTRAN working group has simplified the connectivity for developers. The developers would only need to tap into the appropriate streams to feed information to the applications.

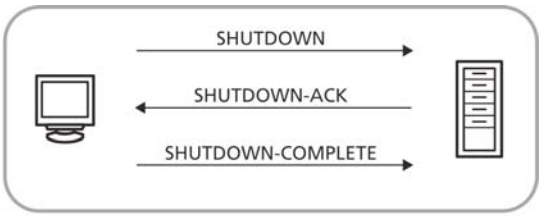
SCTP Additional Benefits Compared to TCP

Allow half-closed connections

Half-closed connections can occur when one side of the conversation believes the connection is closed, but the other believes it is still open. TCP uses a four-way finish command consisting of bidirectional FIN and FIN-ACK messages. The half-open connection could exist if the red messages are not sent (see picture below). SCTP removes this possibility by using a three-way shutdown consisting of SHUTDOWN, SHUTDOWN-ACK, and SHUTDOWN-COMPLETION. Once this is initiated, both sides immediately cease communications. If more information needs to be sent, then a new connection is required.



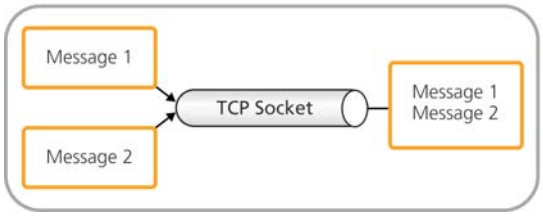
TCP Connection Termination
(Half-open shown by red)

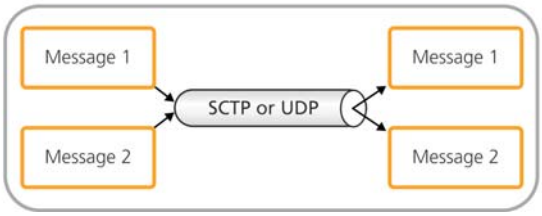


SCTP Close Connection

Preservation of message boundaries

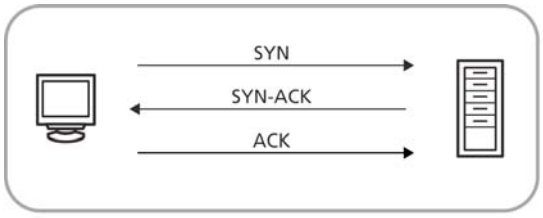
If a client sends a 100 byte and then a 50 byte message, the information is presented to the server with preserved message boundaries. With SCTP and UDP the messages are sent as 100 bytes and 50 bytes. With TCP the messages may be sent/received as a 150 byte message. In the example below, the message was received as one message. This forces the application to separate the messages back into their original format. With SCTP and UDP, the message's boundaries are maintained and the application does not have to split the messages.



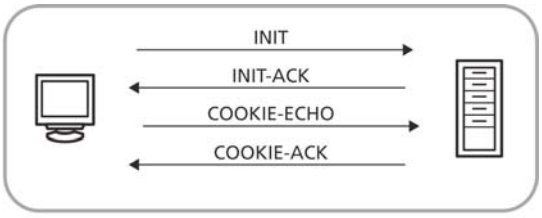


Protect against SYN flooding attacks

In order to understand SYN flooding attacks, we need to look at how a typical connection is established. With TCP the client initiates communications with a synchronize request or a SYN packet. The server responds acknowledging with a SYN-ACK, and finally the client acknowledges the acknowledgement with an ACK packet. This is commonly referred to as a "three-way handshake." Once all three steps are complete, communications can begin. By contrast SCTP uses a "four-way handshake," but may begin sending information on the third step. Then, the SCTP client initiates communications with an INIT packet. The server acknowledges with the INIT-ACK packet and a cookie (unique context that identifies the connection). The client then sends the server's cookie back to the server; the client can also send additional information after the COOKIE-ECHO. The server then acknowledges the COOKIE-ECHO with a COOKIE-ACK.

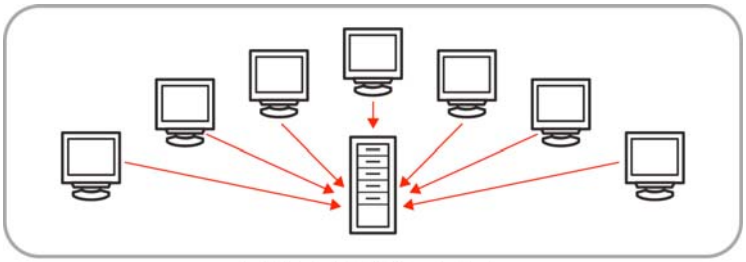


TCP Connection Initiation



SCTP Connection Initiation

A SYN flood attack occurs when a client or multiple clients send SYN packets to a server. This causes the target to commit resources and will usually overload the server, causing it to reboot or worse. Within TCP, the server has already committed resources for the connection. Within a TCP connection, the two endpoints must commit ephemeral ports, memory, and CPU processing for each new connection. For the purposes of committing resources, TCP connections begin with receiving a SYN packet. Since UDP is a connection-less protocol it does not apply. Using SCTP, the servers do not commit resources for the connection until the COOKIE-ECHO is received. This means that the SCTP client must initially commit resources in order to be able send information.



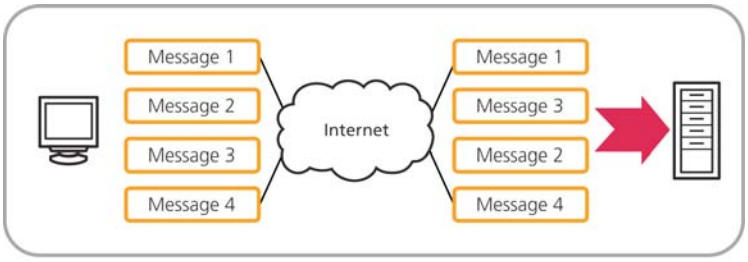
Distributed SYN flood attack

Selective acknowledgements

In standard TCP, every message, or packet of information must be accounted for, resent as necessary, and processed in the order they were sent. SCTP has the ability to selectively acknowledge receipt of missing, disordered, or duplicated messages. Due to the nature of telecommunications most applications would end up discarding any unsynchronized messages. Therefore, the need to send and receive the information is forgone. This would mean that a portion of a word, a portion of a video, or a piece of the whiteboard refresh would be skipped over. The applications and users may notice a slight skip in the voice, video, or refresh. This is referred to as jitter within the telecommunications world and a small amount of jitter is often preferred to having the packet resent and reprocessed which would double the amount of jitter, usually making it more noticeable to the users.

Unordered data delivery

Due to the very nature of networks not all packets may travel across the exact same path. If there is a time-delay using one path over another, the original messages could be out of order when received. Unordered data delivery allows for this instance and can correct the issue by reordering the messages correctly. Using TCP's reliable data transfer feature requires that packets be processed in order. If one is missing or out of order, the packet must be reordered before processing can continue. For example, in the diagram below, if we were using TCP, once Message 3 was received all processing would stop, and wait for Message 2, it would be processed and then Message 3 would be processed. SCTP allows for unordered data delivery and since it has multiple streams, only the one affected is temporarily blocked. As in the diagram below, SCTP would process the messages in the order they arrived, not waiting for them to be numerically ordered. With SCTPs reliable transfer, many networked disk solutions already provide ordering service; SCTP's ability to simply pass the data on relieves the server of the unnecessary overhead of reordering.



Unordered Data Delivery

Summary

The intent of this paper is to familiarize the reader with both the vision and need for SCTP. While it currently has a very limited use, its capabilities will soon prove too tempting, and SCTP adoption will grow quickly. Within IMS architecture, SCTP provides the call signaling over IP. The key benefits of SCTP, multi-homing and multi-streaming, will allow development of numerous new devices which include the ability to support and carry this new protocol.

Appendix 1. Feature/Service Explanations

Feature/Service	Explanation
Allow half-closed connections	Half-closed connections can occur when one side of the conversation believes the connection is closed, but the other believes it is still open.
Application PDU bundling	Application Protocol Data Unit (PDU) packets are bundled together prior to transmission.
Application PDU fragmentation	Application Protocol Data Unit (PDU) packets are un-bundled together after receipt.
Congestion control	Congestion control means that network is checked prior to transmission to help avoid congesting the links.
Connection-oriented	Connection-orient is based on both ends of a communications understanding that they are passing data.
ECN capable	Explicit Congestion Notification (ECN) notifies on entry or exit to a connection of congestion.
Flow control	Ability to adjust data transmission, specifically the quantity. See MTU for size/volume.
Full duplex	The ability to send and receive simultaneously.
Multi-homing	Ability to have multiple IP addresses and use either/both of them simultaneously as needed. This is like having a multiple line capable telephone, but that you can handle multiple calls simultaneously.
Multi-streaming	The ability to send/receive multiple data streams within a single connection.
Ordered data delivery	This means that when packet # 1, 2, and 3 are sent, that they need to be processed in order, even if packet 3 arrives before packet 2.
Partial-reliable data transfer	Allows the user to specify, on a per message basis, the rules governing how persistent the transport service should be in attempting to send the message to the receiver.
Path MTU discovery	Maximum Transmission Unit (MTU). The largest size of packet that can be sent from the sender to receiver without needing to be fragmented.
Preserve message boundaries	Message boundaries are defined by the application. This is whether or not the transmission protocol maintains the boundaries.
Protect against SYN flooding attacks	SYN flooding attacks occur when multiple systems send requests to establish connections, but do no further data transmissions. It is an attack methodology.
Pseudo-header for checksum	An additional header containing a checksum of the data within the packet. These were added to TCP and UDP.
Reachability check	Ability to confirm whether a remote device is alive or dead, prior to sending data.
Reliable data transfer	Reassurance that the data sent, in multiple pieces, was all received uncorrupted.
Selective Acknowledgements	Ability to notify sender of the receiver missing, duplicate, and mis-ordered packets.
Time wait state	After a connection is closed, this is the amount of time that a system waits before reusing the ephemeral port and IP address.
Unordered data delivery	The ability to receive packets out of the order they were sent.

[i] Randall Stewart, Paul D. Amer. "Why is SCTP needed given TCP and UDP are widely available?" ISOC MEMBER BRIEFING #17, June 2004.