SCTP: A Proposed Standard for Robust Internet Data Transport

The stream control transmission protocol is a new standard for general-purpose transport proposed by the Internet Engineering Task Force. SCTP addresses application and security gaps left open by its predecessors, TCP and UDP.

Armando L. Caro Jr.
Janardhan R. Iyengar
Paul D. Amer
Sourabh Ladha
Gerard J. Heinz II
Keyur C. Shah
University of Delaware

Most Internet protocol-based networks employ either the transmission control protocol (TCP) or the user datagram protocol (UDP) for data transfer. However, these two general-purpose protocols provide disjointed services and do not ideally satisfy all application needs.

The general-purpose stream control transmission protocol is designed to expand the scope beyond TCP and UDP. SCTP evolved from a telephony signaling protocol for IP networks.[1] Today it is a proposed Internet Engineering Task Force standard (RFC 2960).[2] Like TCP, SCTP provides a reliable, full-duplex connection and mechanisms to control network congestion. Unlike both TCP and UDP, however, SCTP offers new delivery options that are particularly desirable for telephony signaling and multimedia applications.

Table 1 compares SCTP's services and features with those of TCP and UDP. An SCTP connection, called an *association*, provides novel services such as *multihoming*, which allows the end points of a single association to have multiple IP addresses, and *multistreaming*, which allows for independent delivery among data streams. SCTP also features a four-way handshake to establish an association, which makes it resistant to blind denial-of-service attacks and thus improves overall protocol security.

PACKET FORMAT

TCP provides a byte-stream data delivery service, whereas SCTP provides a message-oriented data delivery service. Figure 1 illustrates a generalization of the SCTP packet format. The packets always begin with an SCTP common header, a minimal structure that provides three basic functions:

- *Source and destination ports*. Together with the IP addresses in the IP header, the port numbers identify the association to which an SCTP packet belongs.
- *Verification tags*. Vtags ensure that the packet belongs to the current incarnation of an association.
- *Checksum*. This computed value maintains the entire packet's data integrity.

The remainder of an SCTP packet consists of one or more *chunks*, concatenated building blocks that contain either control or data information. This format differs from TCP and UDP packets, which include control information in the header and offer only a single optional data field.

SCTP control chunks transfer information needed for association functionality, while data chunks carry application-layer data. The current specification defines 14 different control chunks for association establishment, association termination, data acknowledgment (ACK), destination failure detection and recovery, explicit congestion notification (ECN), and error reporting. SCTP is extensible, allowing new control chunk types to be defined in the future.

Each chunk has a chunk header that identifies its length, type, and any special flags the type needs. SCTP has the flexibility to concatenate different

Published by the IEEE Computer Society

chunk types into a single data packet. The only restriction is on packet size, which cannot exceed the destination path's maximum transmission unit (MTU) size.

MULTIHOMING

Mission-critical systems rely on redundancy at multiple levels to provide uninterrupted service during resource failures. Such systems often deliver network redundancy by multihoming their hosts. As Figure 2 shows, a multihomed host is accessible through multiple IP addresses. If one of its IP addresses fails—possibly from an interface or link failure, severe congestion, or slow route convergence around path outages—the destination host can still receive data through an alternate source interface.

To benefit from this network-layer redundancy, SCTP supports multihoming at the transport layer. A multihomed SCTP end point can bind to multiple IP addresses when that end point initializes an association.

To contrast SCTP with TCP in multihomed hosts, consider the potential connections in Figure 2:

- A TCP connection uses a single IP address at each end point. Hence, four distinct TCP connections are possible between hosts *A* and *B*: (A_1, B_1), (A_1, B_2), (A_2, B_1), or (A_2, B_2).
- SCTP, on the other hand, allows a single association to span all of the IP addresses at each end point. Hence, an SCTP association between hosts *A* and *B* could consist of two sets of IP addresses: $(\{A_1, A_2\}, \{B_1, B_2\})$.

Currently, SCTP uses multihoming only for redundancy, not for load balancing. Each end point chooses a single primary destination address for sending all new data chunks during normal transmission. An end point sends retransmitted data chunks to an alternate address under the assumption that alternate paths increase the probability of the chunks reaching the peer end point. Continued failure to reach the primary address ultimately results in failure detection, at which point the end point transmits all chunks to an alternate destination until the primary destination becomes reachable again.

If B1 in Figure 2 is the primary destination for host A and becomes unreachable, multihoming keeps the SCTP association alive by allowing host A to send data to alternate destination B2. SCTP's built-in failure detection and recovery system performs failover and allows end points to dynamically send traffic to an alternate peer IP address. In this example, the SCTP association would redirect traffic to B2 until B1 becomes reachable again, potentially transparent to the application.

SCTP keeps track of each destination address's reachability through two mechanisms: ACKs of data chunks and *heartbeat* chunks—control chunks that periodically probe a destination's status. Currently in RFC 2960,[2] if six consecutive timeouts occur on either data or heartbeat chunks to the same destination, the sender concludes that that

Table 1. Comparison of SCTP services and features with those of TCP and UDP.

Services/Features	SCTP	TCP	UDP
Full-duplex data transmission	yes	yes	yes
Connection-oriented	yes	yes	no
Reliable data transfer	yes	yes	no
Partially reliable data transfer	optional	no	no
Ordered data delivery	yes	yes	no
Unordered data delivery	yes	no	yes
Flow and congestion control	yes	yes	no
Explicit congestion notification support	yes	yes	no
Selective acks	yes	optional	no
Preservation of message boundaries	yes	no	yes
Path maximum transmission unit discovery	yes	yes	no
Application data fragmentation/bundling	yes	yes	no
Multistreaming	yes	no	no
Multihoming	yes	no	no
Protection against SYN flooding attack	yes	no	n/a
Half-closed connections	no	yes	n/a

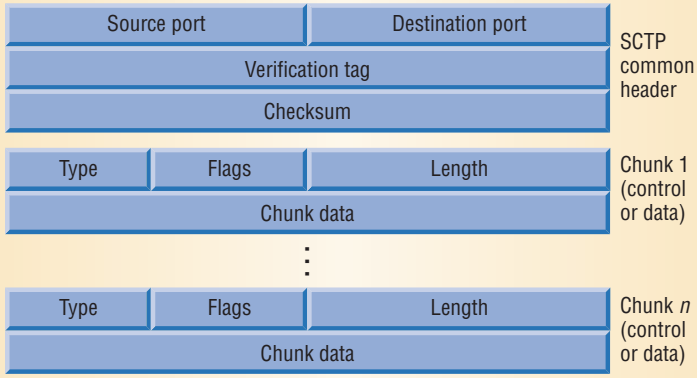

Figure 1. SCTP packet format. The common header is followed by one or more concatenated chunks containing either control or data information.

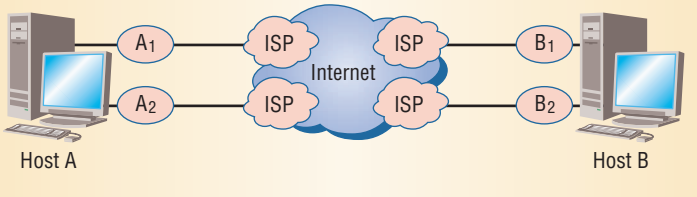

Figure 2. Multihomed hosts. A_1 and A_2 represent two IP addresses for the end point host A. B_1 and B_2 represent two IP addresses for host B.

Related SCTP Multihoming Research

Researchers in the University of Delaware's Protocol Engineering Laboratory (www.cis.udel.edu/) are investigating innovative transport protocols in current projects that apply to SCTP.

Retransmissions on multihomed hosts

SCTP's policy specifying an alternate peer IP address for retransmissions assumes that packet loss is due to congestion and that retransmitting the chunks increases the probability that they will reach the peer end point. Under certain conditions, however, the round-trip time (RTT) measurements to alternate destinations are insufficient to determine appropriate retransmission time-outs. The result is an overly conservative RTT that decreases throughput. Furthermore, a retransmission sent to an alternate destination cannot credit the primary destination's congestion window, even though the primary destination receives most of the data transferred. Again, the result is decreased throughput.

Mechanisms that either increase the number of RTT measurements to the alternate destinations or reduce the number of time-outs might provide better performance with SCTP's current retransmission policy, but these solutions do not address the congestion window issue. The work at PEL shows that changing the retransmission policy to retransmit lost chunks to the same destination as the original transmission addresses both issues.[1] This research also suggests a retransmission policy for avoiding performance degradation during failure.

Adaptive failover mechanism

SCTP's current failover mechanism is static, whereas network dynamics vary greatly and applications also have different failover requirements. For example, telephony signaling applications require that failovers take no longer than 800 ms, but a file transfer may be more concerned with time taken to complete delivery. In the first case, failover must occur sooner but at the potential expense of performing failover when no failure has occurred and possibly degraded throughput.

Early work at PEL argued for network fault tolerance to cope with dynamic network conditions and varying application needs.[2] Subsequent work developed and formally specified a two-level threshold failover mechanism,[3] which serves as a generic framework for an adaptive SCTP failover algorithm.

Robust changeover

SCTP provides for application-initiated changeover by letting the sending application change the sender's primary destination address, thus moving outgoing traffic to a different path. PEL researchers identified a problem in the current SCTP specification that results in unnecessary retransmissions and overgrowth of the sender's congestion window during certain changeover conditions. They have proposed algorithms for making SCTP robust to the negative effects of a single changeover.[4]

Concurrent multipath transfer

Currently, multihomed SCTP end points may only transmit new data to a single destination at any time. PEL researchers are currently working on mechanisms to simultaneously send data across multiple end-to-end paths to accomplish end-to-end load balancing, or *concurrent multipath transfer*.[5] This work identified three negative side effects of reordering introduced by CMT that must be managed to achieve the full performance gains of parallel transfer. The PEL work proposed algorithms to counter these side effects and is now investigating CMT's effects on network congestion and failover performance. ■

References

1. A. Caro et al., "Retransmission Policies with Transport Layer Multihoming," to be published in *Proc. 11th IEEE Int'l Conf. on Networks* (ICON 2003), IEEE Press, 2003.
2. A. Caro et al., "Using SCTP Multihoming for Fault Tolerance and Load Balancing," *ACM Computer Comm. Rev.*, July 2002, p. 17.
3. A. Caro et al., "A Two-Level Threshold Recovery Mechanism for SCTP," *Proc. World Multiconference on Systemics, Cybernetics, and Informatics* (SCI 2002), vol. X, Int'l Inst. Informatics and Systemics, 2002.
4. J. Iyengar et al., "Making SCTP More Robust to Changeover," presented at the *Int'l Symp. Performance Evaluation of Computer and Telecommunication Systems* (SPECTS 2003), sponsored by the SCS Society for Simulation and Modeling Int'l, 2003; www.cis.udel.edu/~iyengar/ publications/2003.spects.iyengar.pdf.
5. J. Iyengar et al., "Concurrent Multipath Transfer Using SCTP," tech. report TR2004-02, Computer and Information Sciences Dept., Univ. of Delaware, 2003.

destination is unreachable and chooses an alternate destination IP address dynamically.

The "Related SCTP Multihoming Research" sidebar describes current research projects at the University of Delaware's Protocol Engineering Laboratory that study retransmission policies and refine multihoming mechanisms to support adaptive failover mechanisms and concurrent multipath transfer.

MULTISTREAMING

Multistreaming is another novel service SCTP includes at the transport layer. An SCTP stream is a unidirectional logical data flow within an SCTP association. The SCTP end points negotiate application-requested streams during association setup that exist for the life of the association.

The "Related SCTP Multistreaming Research" sidebar describes current research at the University of Delaware's Protocol Engineering Laboratory to enhance and exploit SCTP multistreaming capabilities for different applications.

Figure 3 shows a multistreamed association between hosts *A* and *B*. During this example's association setup, host *A* requested three streams to host *B* (numbered 0 to 2), and host *B* requested only one stream to host *A* (numbered 0).

Related SCTP Multistreaming Research

SCTP streams are independent, so research is under way to support their adaptation to specific application requirements.

FTP over multistreamed SCTP

The classic file transfer protocol (FTP), defined in IETF RFC 959, uses one TCP connection for control information and a separate, nonpersistent TCP connection for each file transfer or directory listing. This out-of-band signaling approach increases latency and creates problems when interacting with network address translators and firewalls.

Protocol Engineering Lab researchers are investigating a single persistent multistreamed association that supports both data and control channels.[1] One stream is dedicated to the control channel, while other streams are used for data transfers.

SCTP multistreaming distinguishes between FTP control and data information in a single SCTP association. Results indicate that SCTP multistreaming significantly improves throughput for multiple file transfers—more than 50 percent, with loss rates between 3 and 10 percent.

Other applications can also exploit SCTP multistreaming benefits. For instance, Web transfers using HTTP may benefit from aggregating multiple transfers in a single SCTP association.

Priority stream scheduling

In other PEL work, researchers are investigating the theoretical and practical implications of adding a priority stream scheduling scheme to SCTP.[2]

Priorities allow the sending end point to give precedence to data specified as critical during periods of increased network delay or decreased throughput. Priority schemes can help applications adapt to periods of heavy network congestion or poor quality of service.

Preferential treatment

Researchers at Telcordia Technologies are investigating ways for applications to indicate QoS per stream. They have modified mechanisms to introduce preferential treatment among streams. These mechanisms use the IP-layer type-of-service header bits to take advantage of QoS support in the underlying network.[3] ■

References

1. S. Ladha and P. Amer, *Improving Multiple File Transfers Using SCTP Multistreaming*, tech. report TR2003-06, Computer and Information Sciences Dept., Univ. of Delaware, 2003.
2. G. Heinz, *Priorities in SCTP Multistreaming*, master's thesis, Computer and Information Sciences Dept., Univ. of Delaware, May 2003.
3. S. Samtani, J. Iyengar, and M. Fecko, "SCTP Multistreaming: Preferential Treatment among Streams," to be published in *Proc. Military Communications Conf.* (MILCOM 2003), ACM Press, 2003.

Within streams, SCTP uses *stream sequence numbers* (SSNs) to preserve the data order and reliability for each data chunk. Between streams, however, no data order is preserved. This approach avoids TCP's head-of-line blocking problem, in which successfully transmitted segments must wait in the receiver's queue until a TCP sending end point retransmits any previously lost segments. TCP's blockage delays delivery of received data to the receiving application until it receives the retransmitted segments, which is unnecessary and sometimes unacceptable in signaling and some multimedia applications.

In SCTP, if data on Stream 1 is lost, only Stream 1 is blocked at the receiver while awaiting retransmissions. The SCTP receiving end point can immediately deliver data arriving without loss on other streams to the application.

While SCTP manages ordering and packet delivery on a per-stream basis, it manages congestion control on a per-destination basis and flow control on a per-association basis. In other words, an SCTP sending end point maintains a separate congestion window for each destination and a single receiver window for the association. A congestion window (or cwnd) constrains the amount of data that an SCTP sender can send, thus controlling the sending rate to avoid congestion in the network. A receiver window (or rwnd) prevents an SCTP sender from sending data too fast to a slower SCTP receiver.

Multistreaming and multihoming are orthogonal services. An end point's multiple streams are logically independent from its multiple interfaces. Data from any stream can potentially travel over any path to any destination address.

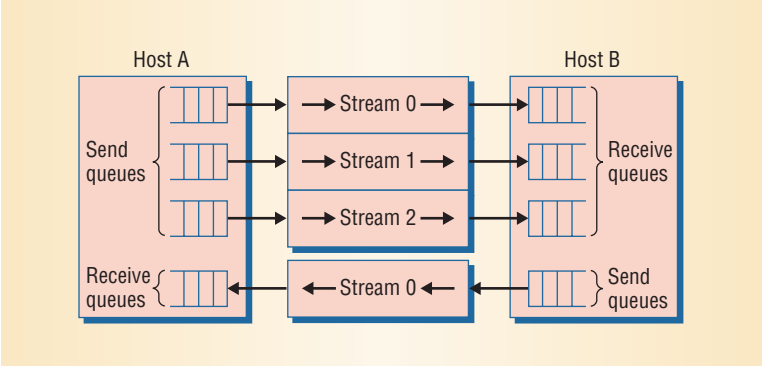

Figure 3. SCTP multistreamed association. Streams are unidirectional logical data flows that the SCTP end points negotiate during association setup.

A congestion manager is an alternative approach for managing multiple streams.[3] In the CM approach, the transport layer (TCP or UDP) manages separate connections between end points, but a sublayer between the transport and network layers aggregates congestion control across these connections. The CM solution, however, still requires end points to open multiple end-to-end connections for handling multiple flows, whereas SCTP uses a single multistreamed association.

Researchers are investigating mechanisms that exploit SCTP to improve performance in wireless and mobile environments.

ASSOCIATION PHASES

As a connection-oriented protocol, an SCTP association has three phases: association establishment, data transfer, and association shutdown.

Association establishment

When a TCP server receives a SYN request to establish a connection during the three-way handshake, the server allocates memory resources, stores state for the SYN received, and replies with a SYN/ACK to the sender. The receiving server maintains this state until it receives an ACK for the SYN/ACK, establishing the connection, or until the SYN/ACK expires with no ACK.

When a malicious user orchestrates a coordinated SYN attack, many other malicious hosts flood a predetermined TCP server with SYNs, causing the server to allocate its full resources to respond. The server then cannot accept valid connection requests. Such SYN attacks are a primary security concern with TCP's three-way handshake.

SCTP uses a four-way handshake in which a cookie mechanism establishes an association that prevents blind SYN attacks. If host *A* initiates an association with host *B*, the following process ensues:

1. Host *A* sends an INIT chunk to host *B*.
2. When host *B* receives the INIT, it returns an INIT-ACK to host *A*. This INIT-ACK contains a cookie composed of information that only *B* can verify. The cookie helps check if host *A* is legitimate. At this point, host *B* does not allocate any memory to maintain state for the requested association. Note that TCP is forced to maintain state at this point in the handshake, making it prone to a blind SYN attack.
3. When host *A* receives the INIT-ACK, it replies with a COOKIE-ECHO chunk. This chunk may contain *A*'s first data, and—as the name indicates—it echoes the cookie that host *B* sent.
4. On receiving the COOKIE-ECHO chunk, host *B* checks the cookie's validity. A valid cookie establishes host *A*'s legitimacy.

Only at this point does SCTP establish the association and allocate resources at host *B*.[1]

In this approach, host *A*—which initiated the association—must maintain state before host *B* does. Although this four-way handshake avoids blind SYN attacks, SCTP does not provide security against an attacker capable of sniffing and replaying traffic between hosts *A* and *B*.

Data transfer

SCTP data chunks are indivisible units. For purposes of reliability, congestion control, and flow control, SCTP assigns each chunk a *transmission sequence number*. A TSN is unique within an association—until the 32-bit number wraps around—and independent of the stream on which the chunk is being sent. Since a chunk is atomic, TSNs only need to be associated with data chunks—as opposed to TCP, which associates a sequence number with each data byte and hence wraps around faster. SCTP peers exchange starting TSN values during association establishment.

Unlike TCP's byte-stream service, SCTP preserves the boundaries of application-layer messages, similar to UDP. When an application has a message larger than the destination path MTU, SCTP fragments the message into multiple data chunks, which can be sent in separate packets.

To assist a receiver in the reassembly process, SCTP assigns the same SSN to all the data chunks associated with a single application message. However, each data chunk has a different TSN, assigned incrementally, to maintain reliability and proper function of the flow and congestion control algorithms.

The data chunk header includes begin and end bits to delimit the fragmented message. Thus, SCTP can fragment a single application message into multiple chunks at the sender for transmission and reassemble them into a single message at the receiver for delivery to the application.

Inversely, SCTP can bundle messages that are smaller than the path MTU into a single packet and unbundle them at the receiver. Since chunks themselves cannot be refragmented, SCTP preserves the original fragmentation boundaries upon retransmission. This differs from TCP, which can concatenate and fragment the previously transmitted data byte stream with different boundaries.

Like TCP, SCTP maintains reliability through acks, retransmissions, and an end-to-end checksum. SCTP uses the CRC-32 checksum to verify packet integrity, as opposed to the 16-bit 1's-complement sum that both TCP and UDP use. SCTP acks carry cumulative (CumAck) and selective (GapAck) information, the latter being similar to TCP's selective ack (SACK) extension. The CumAck indicates the TSNs received in sequence thus far, and the receiver sets the CumAck to the last TSN successfully received in sequence. The GapAck blocks indicate TSNs received out of order beyond the CumAck.

Fast retransmit and time-out mechanisms handle packet loss detection and recovery. SCTP's con-

gestion control algorithms are derived from TCP's, with changes to allow for multihoming. For example, the SCTP sender maintains a separate set of congestion control parameters per destination address.

Association shutdown

SCTP's association shutdown, illustrated in Figure 4, is a three-way handshake, as opposed to TCP's four-way handshake. This three-way process does not allow half-closed connections, in which one end point shuts down while the other end point continues sending new data. SCTP's authors decided that half-closes were not used often enough in practice to warrant the extra complexity in the SCTP shutdown procedure. Either one of the end points engaged in the association can initiate the shutdown.

SCTP STATUS

Many companies and universities are working with SCTP. At least 26 implementations currently exist, and the IETF Transport Area is considering several extensions to SCTP. Current research also addresses wireless and multimedia issues.

Implementations

Kernel implementations of SCTP exist for many mainstream operating systems, including FreeBSD, NetBSD, OpenBSD, Linux, Solaris, AIX, and HP-UX. User-space implementations exist for a larger number of operating systems, including Windows, Mac OS X, and proprietary platforms from Cisco, Nokia, Siemens, and other vendors.

In addition, SCTP modules exist for network simulation tools—ns-2 (http://pel.cis.udel.edu) and Opnet[4]—and for packet-sniffing utilities—tcpdump (www.tcpdump.org) and Ethereal (www.ethereal.com).

Nontelephony applications. Finally, several nontelephony user applications now run SCTP, including the Mozilla Web browser and the Apache Web server (www.sctp.org). Other ported applications include the BSD FTP client and server, an MPEG-4 streamer and player,[5] and a live video streaming application (http://netlab.cis.temple. edu/sctpcam).

Interoperability testing. Since June 2000, six conformance interoperability testing workshops, known as SCTP Interops, have tested implementations. Inconsistencies and ambiguities from the tests are documented in the "SCTP Implementer's Guide," a working draft of changes that the IETF Transport Area working group will merge with RFC 2960 when SCTP moves to draft standard status.

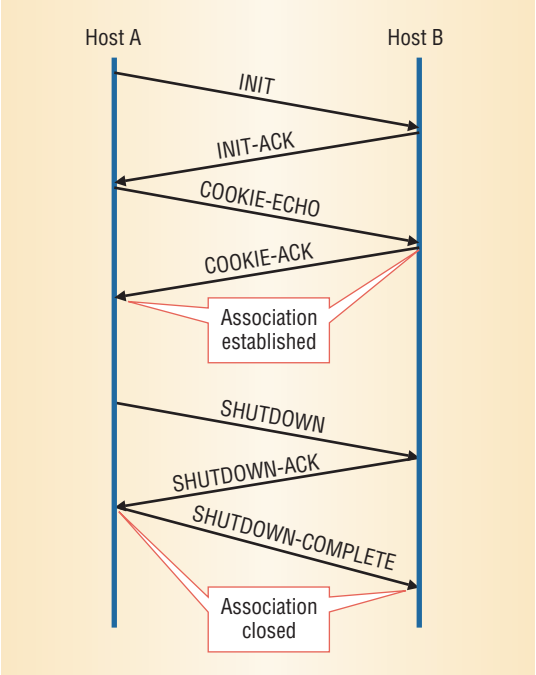

Figure 4. Association establishment and shutdown. SCTP uses a cookie mechanism in a four-way handshake to establish an association. The shutdown process is a three-way handshake.

Proposed extensions

The IETF Transport Area working group is considering two significant extensions to SCTP.

Dynamic address reconfiguration (draft-ietf-tsvwg-addip-sctp) lets SCTP end points reconfigure IP address information on an existing association and set a peer's primary destination. This functionality provides a graceful method for adding IP addresses and deleting them from existing associations—say, for platforms that hot swap network interface cards, or mobile environments that dynamically allocate IP addresses when hosts change IP domains.

Partial reliability (draft-stewart-tsvwg-prsctp) lets a user specify a reliability level on a per-message basis. The reliability level defines how persistent an SCTP sender should be in attempting to communicate a message to the receiver—for example, never retransmit, retransmit up to k times, retransmit until lifetime expires, or retransmit until the association aborts. Partially reliable SCTP (PR-SCTP) introduces the flexibility to provide intermediate reliability levels, in addition to the two extremes that UDP and TCP currently provide.

Wireless and mobile environments

Researchers are investigating mechanisms that exploit SCTP's novel features to improve performance in wireless and mobile environments.

Fixed network with roaming hosts. Current research comparing the performance of SCTP and two variants of the TCP protocol—TCP Reno and TCP

IETF research efforts have transformed SCTP into a general-purpose Internet protocol.

Reno with Eifel—in wireless mobile environments shows that SCTP, like TCP, suffers from spurious time-outs when delay spikes occur.[6]

University of Oklahoma researchers have also shown that when an end point uses mobile IP,[7] SCTP outperforms TCP Reno and TCP SACK during handoffs.[8] Since losses are common during handoff, a robust loss recovery mechanism is a key to improving performance during handoffs. Although the SACK mechanisms of both TCP and SCTP improve throughput, SCTP allows a larger number of SACK blocks and thus outperforms TCP SACK.

Researchers from Siemens AG have submitted an Internet draft for mobile SCTP (draft-riegel-tuexen-mobilesctp), which uses the proposed SCTP extension for dynamic address reconfiguration to manage mobility at the transport layer and avoid some of the performance and deployment drawbacks of the IETF's mobile IP. Mobile SCTP allows a mobile host to maintain SCTP associations without using mobile IP. Instead, a mobile host dynamically adds and deletes IP addresses to existing SCTP associations as needed.

Satellite networks. Research sponsored by the US National Aeronautics and Space Administration to evaluate the performance of SCTP in satellite networks showed that the protocol performs better than TCP SACK.[9]

Boeing researchers investigated the performance of single-homed SCTP, multihomed SCTP, plain TCP SACK, and TCP SACK with an optimized satellite gateway.[10]

For small file transfers, the Boeing researchers found that SCTP's larger packet overheads caused plain TCP SACK to perform better than single-homed SCTP, but multihomed SCTP outperformed TCP SACK even with an optimized satellite gateway—unless the probability of a path outage was greater than 60 percent.

For large file transfers, single-homed SCTP performed better than TCP SACK without an optimized gateway, but worse than TCP SACK with an optimized gateway. Multihomed SCTP had the poorest performance for large file transfers, which the researchers attributed to SCTP's current policy of always using an alternate peer IP address for retransmissions. The "Related SCTP Multihoming Research" sidebar describes research on alternative retransmission policies.

Ad hoc networks. City University of New York researchers have shown that SCTP suffers from the same problems as TCP when used in multihop wireless networks.[11] They found that well-known hidden and exposed node problems cause performance to degrade when the number of hops increases.

MPEG-4 streaming over PR-SCTP

Researchers at the University of California, Los Angeles, used ns-2 simulations to investigate the PR-SCTP extension for streaming video applications.[12] Their results demonstrated the merits of applying different reliability levels to different components of an MPEG-4 video stream. Using PR-SCTP for MPEG-4 video streaming improved video quality and consistency.

Cisco Systems researchers modified existing video streaming client (Cisco MPEG4IP Player) and server (Apple Darwin Streaming Server) applications on FreeBSD machines to use the real-time transport protocol (RTP) over PR-SCTP instead of over UDP.[5] They used video streams of low (200 Kbps) and medium quality (700 Kbps) at loss rates of 0.4 to 12.5 percent.

Preliminary results showed that RTP/PR-SCTP provides a higher peak signal-to-noise ratio than RTP/UDP at lower loss rates. However, RTP/PR-SCTP performs worse than RTP/UDP at higher loss rates—perhaps because RTP/PR-SCTP's congestion control mechanisms respond to loss by reducing the streaming rate at higher loss rates. On the other hand, RTP/UDP is unresponsive to loss and continually streams at a constant rate, which improves application performance at the cost of possibly increased network congestion.

Although SCTP is an evolving protocol, efforts within the IETF have transformed it into a general-purpose Internet protocol that expands transport layer possibilities beyond what TCP and UDP offer. SCTP is expected to have a large deployment base in the telephony world, but its novel features and increasing application support hold the key for larger deployment in nontelephony communities. With support from companies such as Cisco, Nokia, Siemens, IBM, and HP, and implementations for more than a dozen operating systems, SCTP has a promising future. ■

Acknowledgments

This article stems from our collaborative participation in the Communications and Networks Consortium sponsored by the US Army Research Laboratory under the Collaborative Technology Alliance Program, Cooperative Agreement DAAD19-01-2-0011. The US government is

References

1. R. Stewart and Q. Xie, *Stream Control Transmission Protocol (SCTP): A Reference Guide*, Addison-Wesley, 2001.
2. R. Stewart et al., "Stream Control Transmission Protocol," IETF RFC 2960 (standards track), Oct. 2000; www.ietf.org/rfc/rfc2960.txt.
3. H. Balakrishnan, H.S. Rahul, and S. Seshan, "An Integrated Congestion Management Architecture for Internet Hosts," *Proc. ACM SIGComm.*, ACM Press, 1999; www.acm.org/sigcomm/sigcomm99/papers/session5-2.html.
4. R. Brennan and T. Curran, "SCTP Congestion Control: Initial Simulation Studies," *Proc. 17th Int'l Teletraffic Congress*, Elsevier Science, 2001; www.eeng.dcu.ie/~opnet/.
5. M. Molteni and M. Villari, "Using SCTP with Partial Reliability for MPEG-4 Multimedia Streaming," *Proc. European BSD Conf.*, 2002; http://bsdconeurope.org/papers/.
6. S. Fu, M. Atiquzzaman, and W. Ivancic, "Effect of Delay Spike on SCTP, TCP Reno, and Eifel in a Wireless Mobile Environment," *Proc. Int'l Conf. Computer Communications and Networks*, IEEE Press, 2002, pp. 575-578.
7. C. Perkins, "IP Mobility Support," IETF RFC 2002 (standards track), Oct. 1996; www.ietf.org/rfc/rfc2002.txt.
8. S. Fu and M. Atiquzzaman, "Improving End-to-End Throughput of Mobile IP using SCTP," *Proc. 2003 Workshop High Performance Switching and Routing*, IEEE Press, 2003; www.cs.ou.edu/~atiq/papers/03-hpsr-fu-sctp-mip.pdf.
9. R. Alamgir, M. Atiquzzaman, and W. Ivancic, "Effect of Congestion Control on the Performance of TCP and SCTP over Satellite Networks," 2002; www.cs.ou.edu/~atiq/papers/congestioncontrol-2.pdf.
10. M. Duke et al., "Stream Control Transport Protocol (SCTP) Performance over the Land Mobile Satellite Channel," to be published in *Proc. Military Communications Conf.* (MILCOM 2003), ACM Press, 2003.
11. G. Ye, T. Saadawi, and M. Lee, "SCTP Congestion Control Performance in Wireless Multihop Networks," *Proc. Military Communications Conf.* (MILCOM 2002), 2002; http://citeseer.nj.nec.com/ye02sctp.html.
12. A. Balk et al., "Investigation of MPEG-4 Video Streaming over SCTP," *Proc. World Multiconference on Systemics, Cybernetics, and Informatics* (SCI 2002), vol. X, Int'l Inst. of Informatics and Systemics, 2002; http://www.cs.ucla.edu/NRL/hpi/tcpw/tcpw_papers/2002-sci-0.ps.

Armando L. Caro Jr. *is a PhD candidate in the Computer and Information Sciences Department at the University of Delaware. His research interests include transport layer services and protocols, network fault tolerance, mobility, and multimedia communication systems. Caro received an MS in CIS from the University of Delaware. He is a student member of the IEEE and the ACM. Contact him at acaro@acm.org.*

Janardhan R. Iyengar *is a PhD candidate in the Computer and Information Sciences Department at the University of Delaware. His research interests include end-to-end services and currently focus on concurrent multipath transfer at the transport layer. Iyengar received an MS in CIS from the University of Delaware. He is a student member of the ACM. Contact him at iyengar@cis.udel.edu.*

Paul D. Amer *is a professor of computer science at the University of Delaware. His research focuses on innovative transport layer services and protocols and multimedia data compression. Amer received a PhD in CIS from Ohio State University. He is a member of the ACM and an associate member of the IEEE. Contact him at amer@udel.edu.*

Sourabh Ladha *is a graduate student in the Computer and Information Sciences Department at the University of Delaware. His research focuses on transport- and application-layer protocols in computer networks. Sourabh received a BS in computer science from the Indian Institute of Technology, Roorkee. Contact him at ladha@cis.udel.edu.*

Gerard J. Heinz II *is an MS student in the Computer and Information Sciences Department at the University of Delaware. His research interests include application design and development, computer networking, and graphics design. Heinz received a BA in CIS from Temple University. Contact him at heinz@cis.udel.edu.*

Keyur C. Shah *is an MS student in the Computer and Information Sciences Department at the University of Delaware. His research interests include computer networking, specifically transport layer protocols. Shah received a BE in computer engineering from the University of Mumbai, India. Contact him at shah@cis.udel.edu.*