SCTP Sendbuffer Advertising

CS4089 Project

End Semester Report

Arpan Kapoor (B120555CS)
Deepak Sirone J (B120097CS)
K Prasad Krishnan (B120128CS)
Guided By: Dr. Vinod Pathari

November 9, 2015

Abstract

Advertising sendbuffer, i.e. the amount of backlogged data present in the sender's buffer has been proposed for TCP and experiments show improvements in application limited flows. This project proposes to advertise the same in SCTP.

1 Introduction

Stream Control Transport Protocol (SCTP) is a reliable transport protocol designed to transport Public Switched Telephone Network (PSTN) signaling messages over IP networks, but is capable of broader applications. Unlike TCP, SCTP offers sequenced delivery of user messages within multiple unidirectional logical channels called streams. Each SCTP endpoint is represented as a set of destination transport addresses, one of which is the primary address. If the primary address becomes unreachable SCTP chooses another destination transport address to route the messages thereafter. This provides network-level fault tolerance and is called multi-homing. It also employs a security cookie mechanism during association initialization to provide resistance to flooding and masquerade attacks.

Advertising the amount of backlogged data present in the sender's buffer can help network operators evaluate the end-to-end performance of a connection in a better way than that with the existing passive measurements. This information can also be used to infer whether a connection is limited by the network or the application.

2 Problem Statement

To propose a scheme to advertise sendbuffer occupancy information, implement it in the Linux kernel and study the performance and security implications of the same.

3 Literature Survey

RFC 3286 [3] provides a high level introduction to the capabilities supported by SCTP, while RFC 4960 [4] describes the complete protocol. Agache and Raiciu [1] propose a scheme to advertise sendbuffer occupancy in TCP. [2] was used to study the state machine employed in the Linux SCTP implementation. It was also used to understand the SCTP packet flow within the kernel.

4 Work Done

Initially, we wrote a file transfer utility that uses SCTP as the transport protocol. We modified a kernel module called `sctp_probe` to measure and plot the sendbuffer size at regular intervals during a file transfer performed using our userspace program.

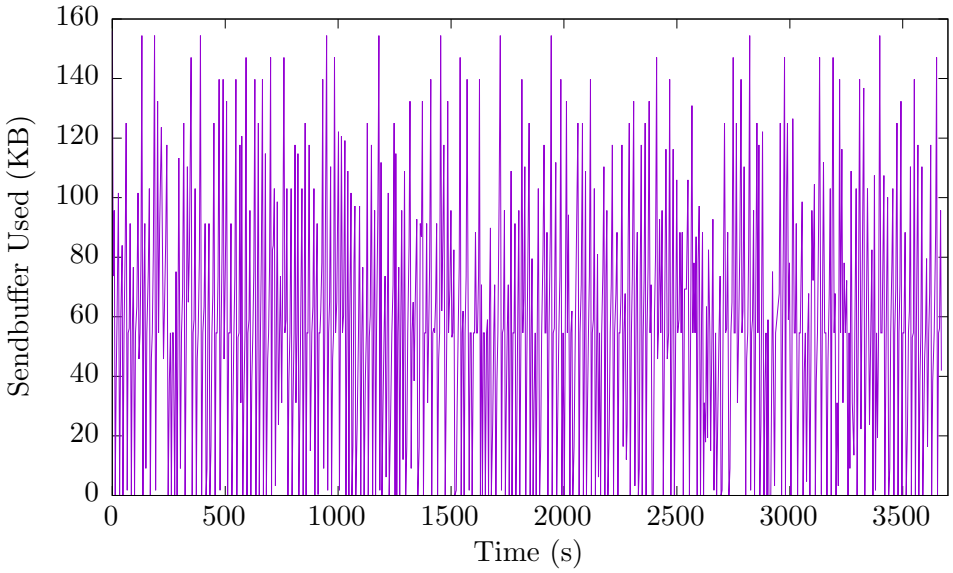

Figure 1: Sendbuffer variation during a file transfer.

We explored the Linux kernel SCTP implementation to understand how the data from userspace is transformed into a SCTP packet and sent to the network layer. The data structures related to the state information, specifically the out queue were studied in detail. The parameter corresponding to the sendbuffer information, which is to be advertised was identified.

4.1 Prerequisite Terms

- **SCTP Chunk** is a unit of information within an SCTP packet, consisting of a chunk header and chunk-specific content.

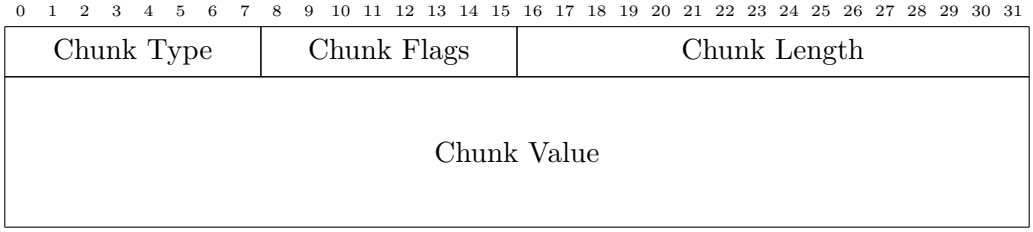

Figure 2: SCTP Chunk Format

- **SCTP Packet** consists of a common header followed by one or more chunks.

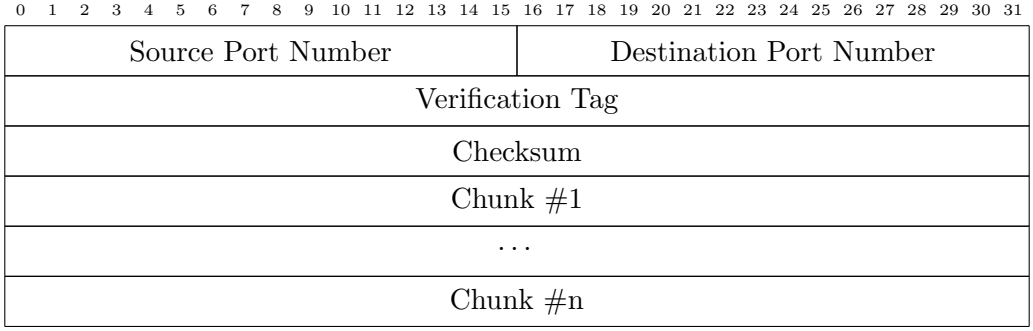

Figure 3: SCTP Packet Format

- **SCTP Stream** referes to a sequence of user messages that are to be delivered to the upper layer protocol in order with respect to other messages within the same stream. This is unlike TCP, where it refers to a sequence of bytes.

4.2 Design

Each chunk is formatted with a Chunk Type field, a chunk-specific Flag field, a Chunk Length field, and a Value field. Chunk Type, which takes a value from 0 to 254, identifies the type of information contained in the Chunk Value field. The Chunk Type values from 15 to 62, 64 to 126, 128 to 190 and 192 to 254 are not used presently in the SCTP implementation. For advertising the sendbuffer occupancy,a new Chunk Type value is selected from 128 to 190. RFC 4960 [4] lays out the actions when the processing endpoint does not recognize the Chunk Type (as is the case here) depending on the highest order 2 bits. RFC 4960 [4] specifies that if the chunk is not recognised by the endpoint and the highest order bits are 10, the chunk will be skipped without sending an Unrecognized Chunk Type error chunk. This ensures that proposed addition of a new Chunk Type would not affect hosts running the unmodified SCTP linux implementation.

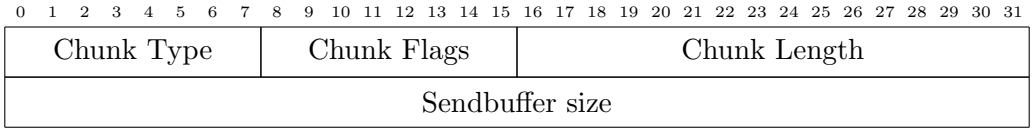

Figure 4: Proposed Chunk for sendbuffer advertisement

The overhead of 8 bytes which can come up in the proposed solution is considerable as it is just 0.53% of a 1500 byte packet.

5 Future Work and Conclusions

To design a working prototype of sendbuffer advertising for SCTP in the Linux kernel and test it in a simulated network. Security implications of the prototype will also be studied.

SDN (Software Defined Networking) based traffic engineering is an emerging area where sendbuffer advertisement can be used for performance optimization. Sendbuffer backlog data can be used by an SDN enabled network switch to assign priorities to packets depending on whether the flow is network limited or application limited, to offer better quality of service to the flow.

The other direction is to use sendbuffer backlog for the identification of congested links by SDN based traffic controllers. Controllers like Hedera use the assumption that flows occupying a bandwidth higher than a certain percentage of the host NIC's speed can be scheduled to occupy an idle link with the assumption that the link will be filled up. Experiments show that this does not always lead to an optimal path allocation. The flow maybe application limited and may not be capable of utilising the entire idle link bandwidth. Hence determining if a flow is network limited or application limited can be done accurately using sendbuffer advertisement.

References

[1] A. Agache and C. Raiciu. *TCP Sendbuffer Advertising*. Internet-Draft draft-agache-tcpm-sndbufadv-00.txt. IETF Secretariat, July 20, 2015.

[2] Karthik Budigere. "Linux Implementation Study of Stream Control Transmission Protocol". In: *Proceedings of Seminar on Network Protocols in Operating Systems*, p. 22.

[3] L. Ong and J. Yoakum. *An Introduction to the Stream Control Transmission Protocol (SCTP)*. RFC 3286. RFC Editor, May 2002, pp. 1–10. URL: `http://www.rfc-editor.org/rfc/rfc3286.txt`.

[4] R. Stewart. *Stream Control Transmission Protocol*. RFC 4960. RFC Editor, Sept. 2007, pp. 1–152. URL: `http://www.rfc-editor.org/rfc/rfc4960.txt`.