Internet Engineering Task Force (IETF)
Request for Comments: 7053
Updates: 4960
Category: Standards Track
ISSN: 2070-1721

M. Tuexen
I. Ruengeler
Muenster Univ. of Appl. Sciences
R. Stewart
Adara Networks

November 2013

SACK-IMMEDIATELY Extension for the Stream Control Transmission Protocol

Abstract

This document updates RFC 4960 by defining a method for the sender of a DATA chunk to indicate that the corresponding Selective Acknowledgment (SACK) chunk should be sent back immediately and should not be delayed. It is done by specifying a bit in the DATA chunk header, called the (I)mmediate bit, which can get set by either the Stream Control Transmission Protocol (SCTP) implementation or the application using an SCTP stack. Since unknown flags in chunk headers are ignored by SCTP implementations, this extension does not introduce any interoperability problems.

Status of This Memo

This is an Internet Standards Track document.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Further information on Internet Standards is available in Section 2 of RFC 5741.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at http://www.rfc-editor.org/info/rfc7053.

Table of Contents

1. Introduction

According to [RFC4960], the receiver of a DATA chunk should use delayed SACKs. This delay is completely controlled by the receiver of the DATA chunk and remains the default behavior.

In specific situations, the delaying of SACKs results in reduced performance of the protocol:

1. If such a situation can be detected by the receiver, the corresponding SACK can be sent immediately. For example, [RFC4960] recommends immediately sending the SACK if the receiver has detected message loss or message duplication.

2. However, if the situation can only be detected by the sender of the DATA chunk, [RFC4960] provides no method of avoiding a delay in sending the SACK. Examples of these situations include ones that require interaction with the application (e.g., applications using the SCTP_SENDER_DRY_EVENT, see Section 4.1) and ones that can be detected by the SCTP stack itself (e.g., closing the association, hitting window limits, or resetting streams, see Section 4.2).

To overcome the limitation described in the second case, this document describes a simple extension of the SCTP DATA chunk by defining a new flag, the "I bit". By setting this bit, the sender of a DATA chunk indicates that the corresponding SACK chunk should not be delayed.

2. Conventions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

3. The (I)mmediate Bit in the DATA Chunk Header

Figure 1 shows the extended DATA chunk.

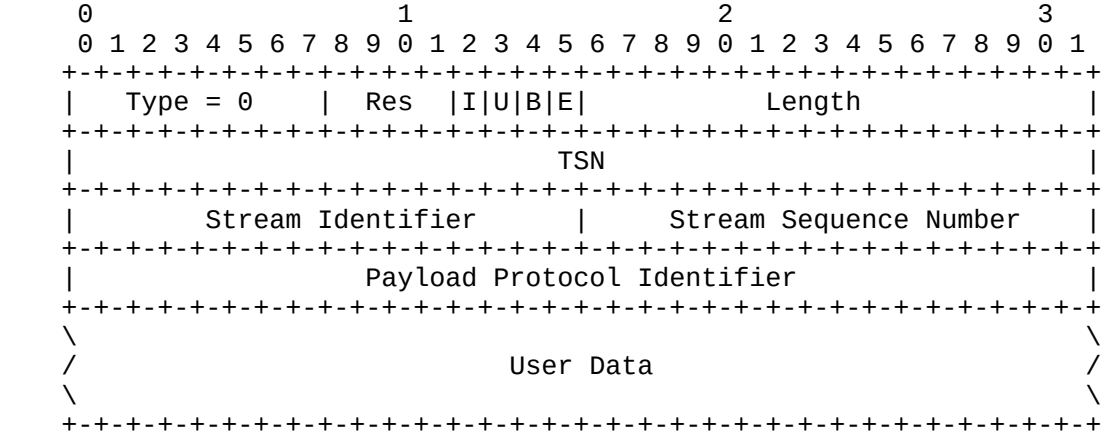

Figure 1: Extended DATA chunk format

The only difference between the DATA chunk in Figure 1 and the DATA chunk defined in [RFC4960] is the addition of the I bit in the flags field of the DATA chunk header.

[RFC4960] defines the Reserved field and specifies that these bits should be set to 0 by the sender and ignored by the receiver.

4. Use Cases

The setting of the I bit can either be triggered by the application using SCTP or by the SCTP stack itself. The following two subsections provide a non-exhaustive list of examples of how the I bit may be set.

4.1. Triggering at the Application Level

One example of a situation in which it may be desirable for an application to trigger the setting of the I bit involves the SCTP_SENDER_DRY_EVENT in the SCTP socket API [RFC6458]. Upper layers of SCTP that use the socket API as defined in [RFC6458] may subscribe to the SCTP_SENDER_DRY_EVENT to be notified as soon as no user data is outstanding. To avoid an unnecessary delay, the application can request that the I bit be set when sending the last user message before waiting for the event. This results in setting the I bit of the last DATA chunk corresponding to the user message; this is possible using the extension of the socket API described in Section 7.

4.2. Triggering at the SCTP Level

There are also situations in which the SCTP implementation can set the I bit without interacting with the upper layer.

If the association is in the SHUTDOWN-PENDING state, setting the I bit reduces the number of simultaneous associations for a busy server handling short-lived associations.

Another case is where the sending of a DATA chunk fills the congestion or receiver window. Setting the I bit in these cases improves the throughput of the transfer.

If an SCTP association supports the SCTP Stream Reconfiguration extension defined in [RFC6525], the performance can be improved by setting the I bit when there are pending reconfiguration requests that require that there be no outstanding DATA chunks.

5. Procedures

5.1. Sender-Side Considerations

Whenever the sender of a DATA chunk can benefit from the corresponding SACK chunk being sent back without delay, the sender MAY set the I bit in the DATA chunk header. Please note that why the sender has set the I bit is irrelevant to the receiver.

Reasons for setting the I bit include, but are not limited to (see Section 4 for the benefits):

- The application requests to set the I bit of the last DATA chunk of a user message when providing the user message to the SCTP implementation (see Section 7).
- The sender is in the SHUTDOWN-PENDING state.
- The sending of a DATA chunk fills the congestion or receiver window.
- The sending of an Outgoing SSN Reset Request Parameter or an SSN/TSN Reset Request Parameter is pending, if the association supports the Stream Reconfiguration extension defined in [RFC6525].

5.2. Receiver Side Considerations

Upon receipt of an SCTP packet containing a DATA chunk with the I bit set, the receiver SHOULD NOT delay the sending of the corresponding SACK chunk, i.e., the receiver SHOULD immediately respond with the corresponding SACK chunk.

6. Interoperability Considerations

According to [RFC4960], the receiver of a DATA chunk with the I bit set should ignore this bit when it does not support the extension described in this document. Since the sender of the DATA chunk is able to handle this case, there is no requirement for negotiating the support of the feature described in this document.

7. Socket API Considerations

This section describes how the socket API defined in [RFC6458] is extended to provide a way for the application to set the I bit.

Please note that this section is informational only.

A socket API implementation based on [RFC6458] needs to be extended to allow the application to set the I bit of the last DATA chunk when sending each user message.

This can be done by setting a flag called SCTP_SACK_IMMEDIATELY in the snd_flags field of the struct sctp_sndinfo structure when using sctp_sendv() or sendmsg(). If the deprecated struct sctp_sndrcvinfo structure is used instead when calling sctp_send(), sctp_sendx(), or sendmsg(), the SCTP_SACK_IMMEDIATELY flag can be set in the sinfo_flags field. When using the deprecated function sctp_sendmsg(), the SCTP_SACK_IMMEDIATELY flag can be in the flags parameter.

8. IANA Considerations

Following the chunk flag registration procedure defined in [RFC6096], IANA has registered a new bit, the I bit, for the DATA chunk.

The "Chunk Flags" registry for SCTP has been updated as described in the following table.

DATA Chunk Flags

Chunk Flag Value	Chunk Flag Name	Reference
0x01	E bit	[RFC4960]
0x02	B bit	[RFC4960]
0x04	U bit	[RFC4960]
0x08	I bit	[RFC7053]
0x10	Unassigned	
0x20	Unassigned	
0x40	Unassigned	
0x80	Unassigned	

9. Security Considerations

See [RFC4960] for general security considerations for SCTP. In addition, a malicious sender can force its peer to send packets containing a SACK chunk for each received packet containing DATA chunks instead of every other received packet containing DATA chunks. This could impact the network, resulting in more packets sent on the network, or the peer, because the generating and sending of the packets has some processing cost. However, the additional packets can only contain the simplest SACK chunk (no gap reports, no duplicate TSNs), since in cases of packet drops or reordering in the network a SACK chunk would be sent immediately anyway. Therefore, this does not introduce a significant additional processing cost on the receiver side. This also does not result in more traffic in the network, because a receiver sending a SACK for every packet is already permitted.

10. Acknowledgments

The authors wish to thank Mark Allmann, Brian Bidulock, David Black, Anna Brunstrom, Gorry Fairhurst, Janardhan Iyengar, Kacheong Poon, and Michael Welzl for their invaluable comments.

11. References

11.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC4960] Stewart, R., "Stream Control Transmission Protocol", RFC 4960, September 2007.

[RFC6096] Tuexen, M. and R. Stewart, "Stream Control Transmission Protocol (SCTP) Chunk Flags Registration", RFC 6096, January 2011.

11.2. Informative References

[RFC6458] Stewart, R., Tuexen, M., Poon, K., Lei, P., and V. Yasevich, "Sockets API Extensions for the Stream Control Transmission Protocol (SCTP)", RFC 6458, December 2011.

[RFC6525] Stewart, R., Tuexen, M., and P. Lei, "Stream Control Transmission Protocol (SCTP) Stream Reconfiguration", RFC 6525, February 2012.

Authors' Addresses

Michael Tuexen
Muenster University of Applied Sciences
Stegerwaldstr. 39
48565 Steinfurt
DE

EMail: tuexen@fh-muenster.de

Irene Ruengeler
Muenster University of Applied Sciences
Stegerwaldstr. 39
48565 Steinfurt
DE

EMail: i.ruengeler@fh-muenster.de

Randall R. Stewart
Adara Networks
Chapin, SC 29036
US

EMail: randall@lakerest.net