Network Working Group
Internet-Draft
Intended status: Experimental
Expires: 28 April 2022

S. McQuistin
V. Band
D. Jacob
C. S. Perkins
University of Glasgow
25 October 2021

Describing TCP with Augmented Packet Header Diagrams
draft-mcquistin-augmented-tcp-example-02

Abstract

This document describes TCP, and a number of its extensions, using Augmented Packet Header Diagrams. This document is an example of the Augmented Packet Header Diagram language: it is not intended as a contribution to any ongoing or future work on maintaining or extending TCP.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on 28 April 2022.

Copyright Notice

Copyright (c) 2021 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (https://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction 2
2. TCP Header 2
3. TCP Options 5
4. Comparison with draft-ietf-tcpm-rfc793bis 8
5. IANA Considerations 8
6. Security Considerations 8
7. Acknowledgements 8
8. Informative References 8
Appendix A. Source code repository 9
Authors' Addresses 9

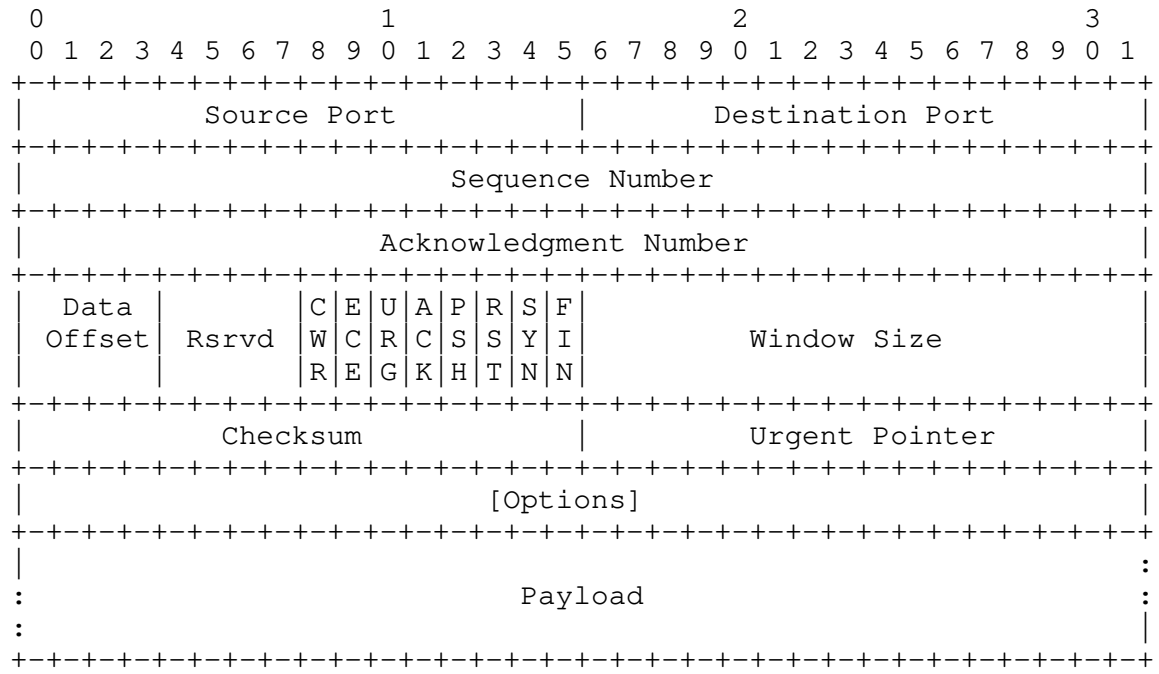
1. Introduction

This document uses Augmented Packet Header Diagrams [AUGMENTED-DIAGRAMS] to describe TCP [RFC793], and is intended to further discussion about the design and implementation of the Augmented Packet Header Diagram language and tooling. Given this purpose, this document is not intended as a contribution to any ongoing or future work on maintaining or extending TCP. Further, this document does not necessarily reflect TCP, and its extensions, as presently standardised.

2. TCP Header

This document describes the TCP protocol. The TCP protocol uses TCP Headers.

A TCP Header is formatted as follows:



where:

Source Port: 16 bits. The source port number.

Destination Port: 16 bits. The destination port number.

Sequence Number: 32 bits. The sequence number of the first data octet in this segment (except when the SYN flag is set). If SYN is set the sequence number is the initial sequence number (ISN) and the first data octet is ISN+1.

Acknowledgment Number: 32 bits. If the ACK control bit is set, this field contains the value of the next sequence number the sender of the segment is expecting to receive. Once a connection is established, this is always sent.

Data Offset (DOffset): 4 bits; DOffset >= 5. The number of 32 bit words in the TCP Header. This indicates where the data begins. The TCP header (even one including options) is an integral number of 32 bits long.

Reserved (Rsrvd): 4 bits; Rsrvd == 0. A set of control bits reserved for future use. Must be zero in generated segments and must be ignored in received segments, if corresponding future features are unimplemented by the sending or receiving host.

CWR: 1 bit. Congestion Window Reduced

ECE: 1 bit. ECN-Echo

URG: 1 bit. Urgent Pointer field significant

ACK: 1 bit. Acknowledgment field significant.

PSH: 1 bit. Push Function (see the Send Call description)

RST: 1 bit. Reset the connection

SYN: 1 bit. Synchronize sequence numbers

FIN: 1 bit; (FIN == 0) || (SYN == 0). No more data from sender.

Window Size: 16 bits. The number of data octets beginning with the one indicated in the acknowledgment field that the sender of this segment is willing to accept.

The window size MUST be treated as an unsigned number, or else large window sizes will appear like negative windows and TCP will not work (MUST-1). It is RECOMMENDED that implementations will reserve 32-bit fields for the send and receive window sizes in the connection record and do all window computations with 32 bits (REC- 1).

Checksum: 16 bits. The checksum field is the 16 bit one's complement of the one's complement sum of all 16 bit words in the header and text. The checksum computation needs to ensure the 16-bit alignment of the data being summed. If a segment contains an odd number of header and text octets, alignment can be achieved by padding the last octet with zeros on its right to form a 16 bit word for checksum purposes. The pad is not transmitted as part of the segment. While computing the checksum, the checksum field itself is replaced with zeros.

Urgent Pointer: 16 bits. This field communicates the current value of the urgent pointer as a positive offset from the sequence number in this segment. The urgent pointer points to the sequence number of the octet following the urgent data. This field is only be interpreted in segments with the URG control bit set.

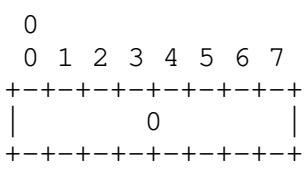
Options: [TCP Option]; size(Options) == (DOffset-5)*32; present only when DOffset > 5. Options may occupy space at the end of the TCP header and are a multiple of 8 bits in length. All options are included in the checksum.

Payload. Payload.

3. TCP Options

A TCP Option is one of: a EOL Option, a NOOP Option, a Maximum Segment Size Option, a Window Scale Factor Option, a Timestamp Option, a SACK Permitted Option, or a SACK Range Option.

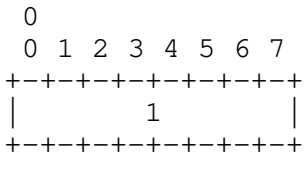
An EOL Option is formatted as follows:



where:

Option Kind (Kind): 1 byte; Kind == 0. This option code indicates the end of the option list.

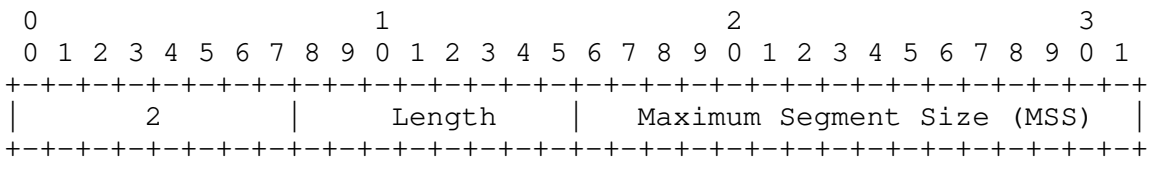
A NOOP Option is formatted as follows:



where:

Option Kind (Kind): 1 byte; Kind == 1. This option code can be used between options, for example, to align the beginning of a subsequent option on a word boundary.

A Maximum Segment Size Option is formatted as follows:



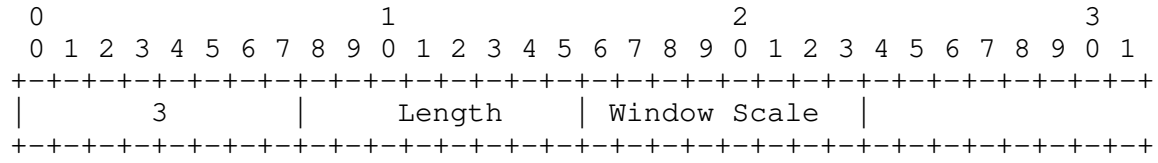
where:

Option Kind (Kind): 1 byte; Kind == 2. If this option is present, then it communicates the maximum receive segment size at the TCP endpoint that sends this segment.

Option Length (Length): 1 byte; Length == 4. Option length.

Maximum Segment Size (MSS): 2 bytes. The maximum segment size allowed.

A Window Scale Factor Option is formatted as follows:



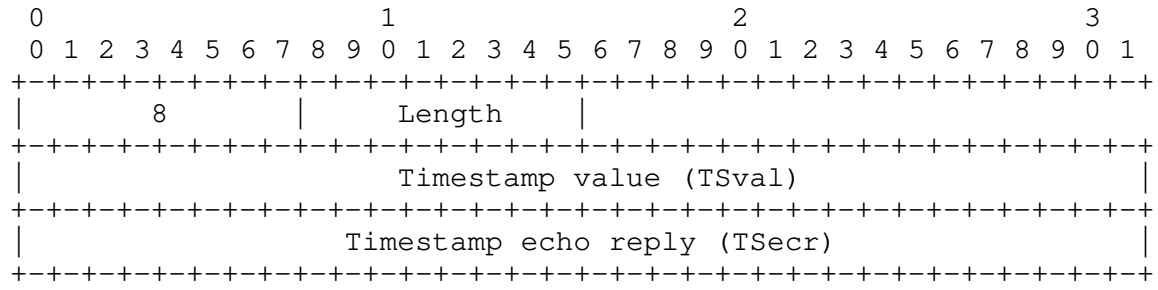
where:

Option Kind (Kind): 1 byte; Kind == 3. If present, this option carries the window scale factor.

Option Length (Length): 1 byte; Length == 3. Option length.

Window Scale: 1 byte. Window scale factor.

A Timestamp Option is formatted as follows:



where:

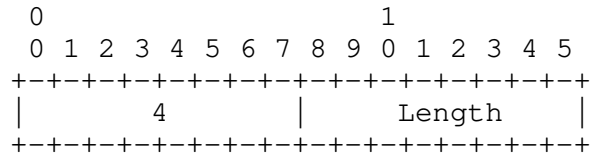
Option Kind (Kind): 1 byte; Kind == 8. If present, this option carries a timestamp and an echoed timestamp.

Option Length (Length): 1 byte; Length == 10. Option length.

Timestamp value (TSval): 4 bytes. TSval.

Timestamp echo reply (TSecr): 4 bytes. TSecr.

A SACK Permitted Option is formatted as follows:

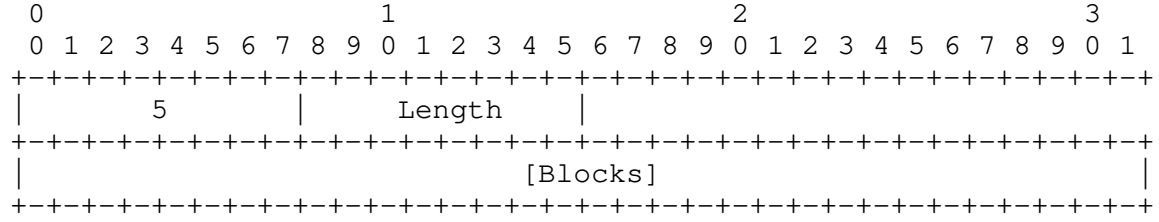


where:

Option Kind (Kind): 1 byte; Kind == 4. If present, this option indicates that SACK is permitted.

Option Length (Length): 1 byte; Length == 2. Option length.

A SACK Range Option is formatted as follows:



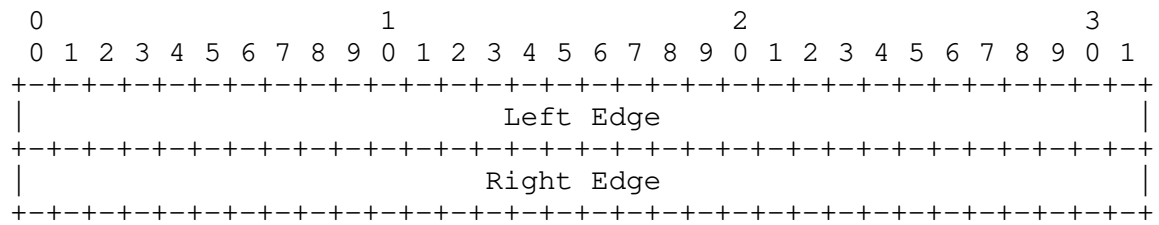
where:

Option Kind (Kind): 1 byte; Kind == 5. If present, this option contains one or more SACK blocks.

Option Length (Length): 1 byte. Option length.

Blocks: (Length-2)/8 SACK Blocks. SACK blocks.

A SACK Block is formatted as follows:



where:

Left Edge: 4 bytes. This is the first sequence number of this block.

Right Edge: 4 bytes. This is the sequence number immediately following the last sequence number of this block.

4. Comparison with draft-ietf-tcpm-rfc793bis

The Augmented Packet Header Diagram format used in this example has also been adopted within [draft-ietf-tcpm-rfc793bis]. This example goes beyond [draft-ietf-tcpm-rfc793bis], and includes a number of TCP options that are not defined as part of that document, including the Window Scale Factor, Timestamp, SACK permitted, and SACK block options. In addition, the definition of the TCP header (Section 2) in this document includes a number of field constraints that are not specified in [draft-ietf-tcpm-rfc793bis].

The purpose of this document is to give an example use of the Augmented Packet Header Diagrams, and not to contribute to ongoing or future TCP standardisation efforts. We include additional constraints and TCP options to demonstrate the ease with which these can be expressed in our format, and that these are then supported by the generated parser code.

5. IANA Considerations

This document contains no actions for IANA.

6. Security Considerations

The security implications of the Augmented Packet Header Diagrams format are considered in [AUGMENTED-DIAGRAMS].

7. Acknowledgements

This work has received funding from the UK Engineering and Physical Sciences Research Council under grant EP/R04144X/1.

8. Informative References

[AUGMENTED-DIAGRAMS]
McQuistin, S., Band, V., Jacob, D., and C. S. Perkins, "Describing Protocol Data Units with Augmented Packet Header Diagrams", Work in Progress, Internet-Draft, draft-mcquistin-augmented-ascii-diagrams-09, 25 October 2021, <http://www.ietf.org/internet-drafts/draft-mcquistin-augmented-ascii-diagrams-09.txt>.

[RFC793] Postel, J., "Transmission Control Protocol", RFC 793, September 1981, <https://www.rfc-editor.org/info/rfc793>.

[draft-ietf-tcpm-rfc793bis]
Eddy, W., "Transmission Control Protocol (TCP) Specification", Work in Progress, Internet-Draft, draft-

ietf-tcpm-rfc793bis-25, 7 September 2021,
<http://www.ietf.org/internet-drafts/draft-ietf-tcpm-rfc793bis-25.txt>.

Appendix A. Source code repository

The source code for tooling that can be used to parse this document, and generate parser code for the protocol it describes, is available from https://github.com/glasgow-ipl/ips-protodesc-code.

Authors' Addresses

Stephen McQuistin
University of Glasgow
School of Computing Science
Glasgow
G12 8QQ
United Kingdom

Email: sm@smcquistin.uk

Vivian Band
University of Glasgow
School of Computing Science
Glasgow
G12 8QQ
United Kingdom

Email: vivianband0@gmail.com

Dejice Jacob
University of Glasgow
School of Computing Science
Glasgow
G12 8QQ
United Kingdom

Email: d.jacob.1@research.gla.ac.uk

Colin Perkins
University of Glasgow
School of Computing Science
Glasgow
G12 8QQ
United Kingdom

Email: csp@csperkins.org