Workgroup: TCPM Working Group
Internet-Draft:
draft-nishida-tcpm-agg-syn-ext-04
Published: 5 July 2023
Intended Status: Standards Track
Expires: 6 January 2024
Authors: Y. Nishida
AWS

Aggregated Option for SYN Option Space Extension

Abstract

TCP option space is scarce resource as its maximum length is limited to 40 bytes. This limitation becomes more significant in SYN segments as all options used in a connection should be exchanged during SYN negotiations. This document proposes a new SYN option negotiation scheme that can aggregate multiple TCP options in SYN segments into a single option so that more options can be negotiate during 3-way handshake. With its simple design, the approach does not require fundamental changes in 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 6 January 2024.

Table of Contents

1. Introduction

TCP option space is scarce resource as its maximum length is limited to 40 bytes because the length of the Data Offset field in the TCP header is 4 bits [RFC9293].

This limitation is a critical issue especially for SYN segments. Because SYN segments need to contain all options expected to be used for the connection, although a TCP endpoint can send only one SYN segment to its peer in a connection. The only exception in the current standards is User Timeout Option [RFC5482]. However, this is because this option provides only advisory information and does not need to be exchanged reliably.

As a result, the current SYN option space tends to be congested. Many TCP connections use MSS [RFC9293], Timestamp and Window Scale [RFC7323], SACK Permitted options [RFC2018] which already consume 19 bytes (4 + 10 + 3 + 2). In addition to these options, if a connection wants to use Multipath TCP [RFC8684], it requires additional 4-12 bytes for MP_CAPABLE or 12-16 bytes for MP_JOIN option. Similarly, TCP Fast Open [RFC7413] and TCP AO [RFC5925] require additional 6-18 bytes and 16 bytes respectively. Moreover, Experimental Option Format defined in [RFC6994] requires 16 bits or 32 bits ExID, which means the length of any experimental options will be 4 bytes or 6 bytes.

If an endpoint is willing to add some of extra options in addition to commonly used options, 40 bytes space may not be sufficient. If a SYN segment cannot accommodate all options that an endpoint wants to use, the endpoint needs to give up using some of them. This problem affects the extensibility of TCP.

There have been various proposals in order to extend option space in SYN Segments such as [I-D.eddy-tcp-loo], [I-D.yourtchenko-tcp-loic], [I-D.touch-tcpm-tcp-syn-ext-opt], [I-D.briscoe-tcpm-inner-space] and [I-D.allman-tcpx2-hack]. These proposals have adopted one or both of the following two types of approach.

*Extending TCP header in SYN segment: This approach tries to accommodate more options in a SYN segment by using payload (e.g override Data Offset field in TCP header).

*Using Multiple SYN or SYN-like segment: This approach uses multiple SYN segments or additional segments that can be treated as a SYN segment (e.g sending another SYN with wrong checksum or from different source port).

However, these kinds of approach induce some complexity as it needs to update fundamental TCP design and have potential risks for middlebox interventions because of it. Instead, we propose a simple alternate approach that can aggregate multiple TCP options into a single options. As this approach does not require drastic changes to TCP SYN negotiation scheme, the risk for middlebox interventions will be minimized. [I-D.boucadair-tcpm-capability-option] also proposes a scheme to aggregate multiple options as many of these options are basically about negotiating support with the peer before actual use of the option. However, our approach requires less option space as it can aggregate and condense some TCP options to create more option space for others. Note that [I-D.boucadair-tcpm-capability-option] specifically target controlled domains to nullify the implications of the presence of middleboxes.

The proposed approach in the draft cannot aggregate all kinds of options. However, we believe it still will be useful especially for newly defined experimental options as it requires at least 4 bytes space in the option field. Also, the proposed approach can be combined with EDO [I-D.draft-ietf-tcpm-tcp-edo] extension or utilize 3rd segments and 4th segments like the feature negotiation scheme for MPTCP if needed.

One example use case for the proposed approach is [I-D.gomez-tcpm-ack-rate-request]. In order to use the feature proposed in the document, endpoints need to exchange a 4-byte TCP option during 3-way handshake so that they can check if the peer is

capable of the feature. However, whether an endpoint supports the feature or not is just 1-bit information. Using 4-byte field to carry 1-bit information looks redundant. On the other hand, Aggregated Option can accommodate up to 18 new TCP options + 3 existing options like into a single TCP option.

Also, even if more than 1-bit information needs to be carried in a TCP option for a certain feature, it is still possible to utilize aggregated options in some cases. In such cases, an endpoint can confirm that the peer supports the feature it wants to use by using the aggregated option. After that, it can continue to negotiate required parameters through 3rd segment and 4th segment. This type of approach is already used in MPTCP [RFC8684]. Hence, we believe this scheme can be applicable to many other TCP extensions.

2. Conventions and Definitions

The key words "**MUST**", "**MUST NOT**", "**REQUIRED**", "**SHALL**", "**SHALL NOT**", "**SHOULD**", "**SHOULD NOT**", "**RECOMMENDED**", "**NOT RECOMMENDED**", "**MAY**", and "**OPTIONAL**" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

This document uses the following terms:

* Original Option format: refers to the option format as defined in [RFC9293]

* Aggregated Option format: refers to indication of support of a TCP option by setting the corresponding bit in the Aggregated TCP Option.

3. Aggregated Option Design

3.1. Option Format

The aggregated option can be used to indicate that an endpoint wants to enable the specified features during SYN segment exchanges. This option uses one bit in the option field for one TCP option. The receiver of the option also uses this option to indicate that it agrees to use the requested features or not. The format of aggregated option format is shown in Figure 1. The option contains 1-byte field named called "Aggregated Block". Aggregated Option can accommodate 1-3 Aggregated Blocks.

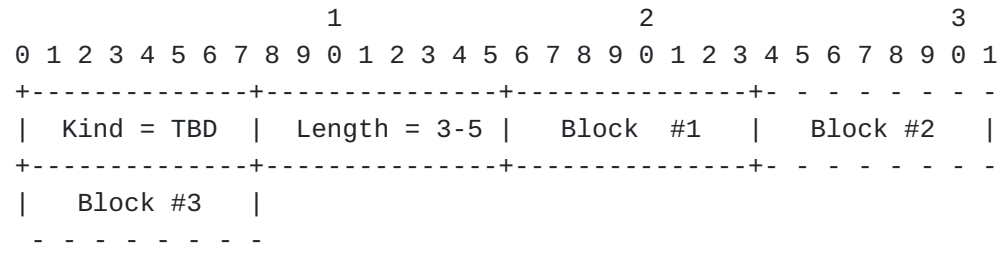

Figure 1: Aggregated Option format

Figure 2 shows the format of Aggregated block. Aggregated block has 1 byte length which consists of 2 bits "Group ID" (GID) field and 6 bits "Option Bits" field. The options supported by Aggregated Option are split into 4 groups. The Group ID field is used to identify the group identifier of the option bits in the Aggregated Option and each single bit in Option Bits field represent a option in the group.

If all options that an endpoint wants to aggregate belong to the same Group ID, the aggregated option needs to contain only one Aggregated Block. Otherwise, it needs to contain multiple blocks as needed.

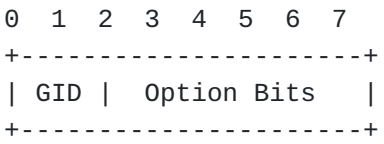

Figure 2: Aggregated Block format

GID field in Aggregated Block indicate the group ID that option bits in the block belongs to. Aggregated Blocks appear in SYN and SYN ACK segments, however, different mappings between GID value in the option and Group ID are used for these two segments. This is because some implementations may copy back unknown options in SYN/ACK segments. These mappings are used not to be confused by such cases. Figure 3 shows the mapping between GID value in the option and Group ID. For example, GID value 1 in SYN represents group 2, while GID value 1 in ACK segment for SYN ACK represents group 1.

GID value in SYN	GID value in SYN ACK	Group ID Description
0	1	group 1
1	2	group 2
2	3	group 3
3	0	group 4

Figure 3: Mapping between GID value and Group ID

The allocation of the bit in Option Bits field in each group will be managed by the registry provided by IANA. Since an aggregated block has 6-bits field to indicate options, one group can have 6 options at most. As a result, the possible maximum number of options with this format will be 4 * 6 = 24. We believe this is sufficient number for the time being based on the current usage of option code points.

The Aggregated Option **MUST** be only used in SYN segments. When an endpoint receives SYN segments with Aggregated Option, it checks Aggregated Blocks in the option. Otherwise, the segment **MUST** be silently discarded. If it contains Aggregated Blocks, the options specified in the blocks **MUST** be processed as well as options in original formats. When a responder sends back a SYN ACK to the initiator, it **SHOULD** send back its response with Aggregated Option. But, it **MAY** uses original format of the options for the response as long as there is enough option space.

3.2. Predefined Aggregated Options

In this proposal, group 1 is used to aggregate commonly used options as predefined aggregated options. Hence, when new aggregated options are registered, they will belong to the rest of groups. Figure 4 shows aggregated option format when group 1 is specified.

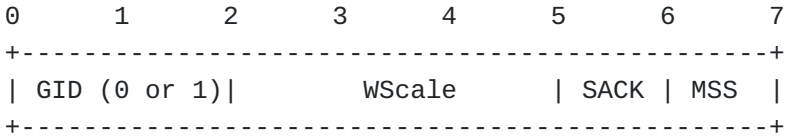

Figure 4

In order to specify group id, the GID field of this format is 0 in SYN and 1 in SYN/ACK. The first 4 bits of 6 bytes Option Bits are used for Window Scale Option. As the value of the shift.cnt in Window Scale Option is 0-14. The shift.cnt values can be stored in the 4 bits as the same format in the original option. When this

value is 15 (all 4 bits are 1), it specifies the window scale option is not aggregated in the segment.

The 5th bit in Option Bits field is used for SACK options. If the bit is set, it indicate the sending endpoint supports Selective Acknowledgement. If the bit is not set, it specifies SACK option is not aggregated in the segment.

The 6th bit in Option Bits field is used for MSS options. If the bit is set, it indicates that the sending endpoint uses 1460 as send MSS which is the most common value used for MSS option. If the bit is not set, it specifies MSS option is not aggregated in the segment.

3.3. Option Bits Registration

The allocation of the Option Bits in Aggregated Option is maintained by IANA. If a new option can be aggregatable, one can request Option Bit in addition to the current procedure, requesting TCP Option Kind Number in [TCPParameters] . If an option already has assigned TCP Option Kind Number, one can request Option bit only which will represent the assigned option kind.

3.4. Utilizing 3rd and 4th Segments for Further Negotiations

Aggregated Option is designed to exchange 1-bit information for each TCP extension that indicate the willingness to use the feature. Hence, if a TCP extension wants to carry more information in the TCP option for the extension, Aggregated Option is basically not applicable for it.

However, it is still possible for these TCP extensions to utilize Aggregated Option in some situations. It is based on the fact that not all TCP extensions will be used right after SYN exchanges. For example, SACK options are only used when there are packet losses. If a TCP extension is not used right after SYN exchange, it is possible to exchange additional parameters by using utilizing 3rd segments and 4th segments. This approach is already used in MPTCP [RFC8684]. As we have a solid precedence, we believe it will not be difficult to implement similar negotiation schemes for other features. However, discussing negotiation schemes with 3rd and 4th segments is out of scope of the document.

4. Security Considerations

We believe Aggregated Option maintains the same level of security as other TCP options does.

5. IANA Considerations

This document requests new TCP option codepoint. In addition, this document requires new registry for the option. They are described in the following subsections.

5.1. Aggregated Option

This document requests to add new option: Aggregated Option to the TCP option space registry which points to this document as follows:

Kind	Length	Meaning	Reference
TBD	N	Aggregated Option	This Document

Figure 5: Aggregated Option Format

5.2. Option Bits Registry for Aggregated Option

This document also requests to create a "Aggregated Option Identifiers" registry in IANA registries. The registry maintains records which are mapped to the TCP Option Kind Number Records in [TCPParameters] These records are divided into 4 groups so that each group contains 6 records.

Acknowledgments

The authors would like to appreciate Mohamed Boucadair for his insightful comments on this document.

Contributors

The contents in this document are the individual contributions from the authors and do not relate to the authors' positions at their affiliations.

References

Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/

RFC2119, March 1997, <https://www.rfc-editor.org/rfc/rfc2119>.

[RFC6994] Touch, J., "Shared Use of Experimental TCP Options", RFC 6994, DOI 10.17487/RFC6994, August 2013, <https://www.rfc-editor.org/rfc/rfc6994>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/rfc/rfc8174>.

[RFC9293] Eddy, W., Ed., "Transmission Control Protocol (TCP)", STD 7, RFC 9293, DOI 10.17487/RFC9293, August 2022, <https://www.rfc-editor.org/rfc/rfc9293>.

Informative References

[I-D.allman-tcpx2-hack] Allman, M., "TCPx2: Don't Fence Me In", Work in Progress, Internet-Draft, draft-allman-tcpx2-hack-00, 8 May 2006, <https://datatracker.ietf.org/doc/html/draft-allman-tcpx2-hack-00>.

[I-D.boucadair-tcpm-capability-option] Boucadair, M. and C. Jacquenet, "TCP Capability Option", Work in Progress, Internet-Draft, draft-boucadair-tcpm-capability-option-01, 8 December 2016, <https://datatracker.ietf.org/doc/html/draft-boucadair-tcpm-capability-option-01>.

[I-D.briscoe-tcpm-inner-space] Briscoe, B., "Inner Space for TCP Options", Work in Progress, Internet-Draft, draft-briscoe-tcpm-inner-space-01, 27 October 2014, <https://datatracker.ietf.org/doc/html/draft-briscoe-tcpm-inner-space-01>.

[I-D.draft-ietf-tcpm-tcp-edo] Touch, J. D. and W. Eddy, "TCP Extended Data Offset Option", Work in Progress, Internet-Draft, draft-ietf-tcpm-tcp-edo-13, 22 October 2022, <https://datatracker.ietf.org/doc/html/draft-ietf-tcpm-tcp-edo-13>.

[I-D.eddy-tcp-loo] Eddy, W. and A. Langley, "Extending the Space Available for TCP Options", Work in Progress, Internet-Draft, draft-eddy-tcp-loo-04, 1 July 2008, <https://datatracker.ietf.org/doc/html/draft-eddy-tcp-loo-04>.

[I-D.gomez-tcpm-ack-rate-request] Gomez, C. and J. Crowcroft, "TCP ACK Rate Request Option", Work in Progress, Internet-Draft, draft-gomez-tcpm-ack-rate-request-06, 12 October

2022, <https://datatracker.ietf.org/doc/html/draft-gomez-tcpm-ack-rate-request-06>.

[I-D.touch-tcpm-tcp-syn-ext-opt] Touch, J. D. and T. Faber, "TCP SYN Extended Option Space Using an Out-of-Band Segment", Work in Progress, Internet-Draft, draft-touch-tcpm-tcp-syn-ext-opt-12, 22 October 2022, <https://datatracker.ietf.org/doc/html/draft-touch-tcpm-tcp-syn-ext-opt-12>.

[I-D.yourtchenko-tcp-loic] Yourtchenko, A., "Introducing TCP Long Options by Invalid Checksum", Work in Progress, Internet-Draft, draft-yourtchenko-tcp-loic-00, 11 April 2011, <https://datatracker.ietf.org/doc/html/draft-yourtchenko-tcp-loic-00>.

[RFC2018] Mathis, M., Mahdavi, J., Floyd, S., and A. Romanow, "TCP Selective Acknowledgment Options", RFC 2018, DOI 10.17487/RFC2018, October 1996, <https://www.rfc-editor.org/rfc/rfc2018>.

[RFC5482] Eggert, L. and F. Gont, "TCP User Timeout Option", RFC 5482, DOI 10.17487/RFC5482, March 2009, <https://www.rfc-editor.org/rfc/rfc5482>.

[RFC5925] Touch, J., Mankin, A., and R. Bonica, "The TCP Authentication Option", RFC 5925, DOI 10.17487/RFC5925, June 2010, <https://www.rfc-editor.org/rfc/rfc5925>.

[RFC7323] Borman, D., Braden, B., Jacobson, V., and R. Scheffenegger, Ed., "TCP Extensions for High Performance", RFC 7323, DOI 10.17487/RFC7323, September 2014, <https://www.rfc-editor.org/rfc/rfc7323>.

[RFC7413] Cheng, Y., Chu, J., Radhakrishnan, S., and A. Jain, "TCP Fast Open", RFC 7413, DOI 10.17487/RFC7413, December 2014, <https://www.rfc-editor.org/rfc/rfc7413>.

[RFC8684] Ford, A., Raiciu, C., Handley, M., Bonaventure, O., and C. Paasch, "TCP Extensions for Multipath Operation with Multiple Addresses", RFC 8684, DOI 10.17487/RFC8684, March 2020, <https://www.rfc-editor.org/rfc/rfc8684>.

[TCPParameters] "Transmission Control Protocol (TCP) Parameters", n.d., <https://www.iana.org/assignments/tcp-parameters/tcp-parameters.xhtml#tcp-parameters-1>.

Author's Address

Yoshifumi Nishida
Amazon Web Services
440 Terry Ave N
Seattle, WA 98109
United States of America

Email: nsd.ietf@gmail.com