Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: August 6, 2021

Y. Nishida
Amazon Web Services

February 2, 2021

Aggregated Option for SYN Option Space Extension
draft-nishida-tcpm-agg-syn-ext-00

Abstract

TCP option space is scarce resource as its max 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 provide a feature to compress TCP options in SYN segments and provide more option space. The proposed scheme does not update the format of TCP header nor transmit any additional SYN or SYN-like segments so that it has lower risks for middlebox interventions. In addition, by combining another proposal for option space extension, it is possible to provide further option space.

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 August 6, 2021.

Table of Contents

1. Introduction 2
2. Conventions and Terminology 4
3. Basic Design 4
3.1. Aggregated Option 4
3.2. Additional Information Exchange Using 3rd ACK Segments . 6
3.3. Examples of Option Exchanges with Aggregated Options . . 7
4. Option Bits Registration 9
5. Discussions 9
5.1. Incremental Deployments for Aggregated Option 10
5.2. Middlebox Considerations 10
5.3. Which Options are Aggregatable 10
5.4. Further Extensions 11
6. Security Considerations 11
7. IANA Considerations 11
7.1. Aggregated Option 11
7.2. Option Bits Registry for Aggregated Option 12
8. References 12
8.1. Normative References 12
8.2. Informative References 12
Author's Address 14

1. Introduction

TCP option space is scarce resource as its max length is limited to 40 bytes. This limitation is a nominal issue especially for SYN segment. This is because although a TCP endpoint can send only one SYN segment to the peer, SYN segments need to contain all options expected to be used for the connection. As a result, the current SYN option space tends to be congested. Many TCP connections use MSS [RFC0793], Timestamp and Window Scale [RFC7323], SACK Permitted options [RFC2018] which already consume 19 bytes (10 + 4 + 3 + 2). In addition to them, 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] consumes more option space as it requires additional 16 bits or 32 bits EXID field than the standard option format. Hence, if an endpoint is willing to activate some of extra features in addition to common options, 40 bytes space may not be sufficient. If SYN segment cannot

accommodate all required options, it means endpoints need to give up using some features. This issue affects the extensibility of TCP.

There have been various proposals in order to extend option space in SYN Segments [I-D.eddy-tcp-loo] [I-D.yourtchenko-tcp-loic] [I-D.touch-tcpm-tcp-syn-ext-opt] [I-D.briscoe-tcpm-inner-space] [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 tries to use multiple SYN segment or additional segment that can be treated as a SYN segment. (E.g sending another SYN with wrong checksum or from different source port)

However, these approach tend to require a certain complexity and have potential risks for middlebox interventions. In this document, in order to reduce the risk for middlebox intervention we propose to extend option space in SYN segments through the following approach which utilizes "3rd segments". In this document, we define "3rd segment" as the segments sent from initiator that contains acknowledge for the SYN ACK segment from responder.

- Aggregated Option Format: we propose a new option format that can aggregate multiple TCP options into a single option format so that it can create more space than conventional option formats.

- Using 3rd segments for supplemental negotiation: we propose to use 3rd segment and its acknowledge for additional feature information exchanges.

One important characteristic of this approach is that it does not require any changes in SYN header format nor using multiple SYN or SYN-like segments. In stead, it utilizes 3rd segments which should looks legitimate segments from middleboxes' points of view. MPTCP [RFC8684] specifications already requires the use of 3rd segment for further information exchanges. Hence, in a sense, the proposal in this document may look using the scheme in MPTCP for more generic purposes. Given that the maturity and the deployment status of MPTCP implementations, we believe that this approach has lower risks for middlebox interventions without introducing lots of complexity in TCP architecture.

2. Conventions and Terminology

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. Basic Design

The proposed scheme in this document has the following two design criteria.

1. All feature negotiations are archived during a single SYN exchange. No extra SYN or SYN-like segments are utilized.

2. No change is required for TCP header format. Use current option space in TCP header and not use payload for option space.

These requirements ensure that the scheme can have simple architecture and can traverse middleboxes without problems. In order to fulfill this requirements, For this purpose, we propose a new option format and a scheme for additional information exchange as follows.

3.1. Aggregated Option

This aggregated option can be used only to indicate that an endpoint want to enable the specified features in the option. This option uses one bit to express one TCP option. This is because this option is used only to indicate that an endpoint wants to use specified features. The receiver of the option can only specify whether it agrees to use the requested features or not. The format of aggregated option format is shown in Figure 1. The option can contains 0-3 Aggregated Blocks which have 1 byte length. The length of the option format is varied by the number of Aggregated blocks in the option.

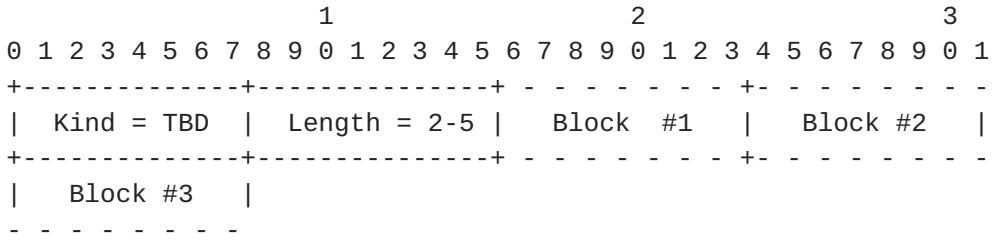

Figure 1: Aggregated Option format

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 is split into 4 groups, such as an option defined in a standard RFC belongs to group 1 or group 2, an option in an experimental RFC belongs to group 3, etc. (Note: how to allocate option bits will be discussed later) 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 one group, the aggregated option will need to contain just one Aggregated Block. However, if the option will need to contain multiple blocks when an endpoint wants to use options in multiple groups.

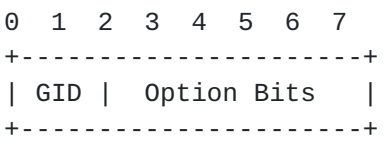

Figure 2: Aggregated Block format

GID field in Aggregated Block indicate the group ID that option bits in the block belongs to. Aggregated Blocks are appeared 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 their response segments. These mappings is used not to be confused by such cases. Table Table 1 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	Extensions

Table 1: 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 shown in Table 1, we currently propose to allocate 3 GID for options and 1 GID is reserved for future extensions. Consequently, the maximum number of options that can be aggregated with Aggregated Option will be limited to 18. We believe this is sufficient number for the time being based on the current usage of option code points. In addition, Section Section 5.4 describes some possibilities to overcome the limitation in case we need to support more options to be supported.

Aggregated Option that contains Aggregated Blocks MUST be only used in SYN segments. When an endpoint received SYN segments with Aggregated Option, it checks Aggregated Blocks in the option. If the option does not contain Aggregated Blocks, the segment MUST be 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 SYN ACK to the initiator, it uses original formats of the options or Aggregated options depends on remaining options space or other criteria.

3.2. Additional Information Exchange Using 3rd ACK Segments

Aggregated Option is used to negotiate the use of features that an endpoint would like to activate. However, it does not provide a way to negotiate additional information. This suffices for options that do not require additional parameters such as SACK-Permit option. However, if options need additional parameter exchanges, these parameters will need to be sent through 3rd segment. When options are sent in the 3rd segments, original formats of the options are used. In this case, in order to solicit acknowledgement for this segments, an initiator MUST include 2 bytes length Aggregated Option to 3rd segment with the format shown in Figure 3.

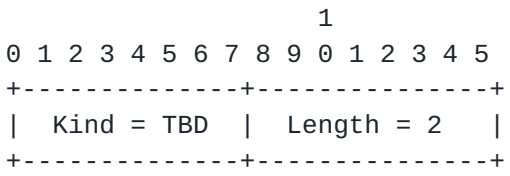

Figure 3: 2-Byte Aggregated Option format

In addition, it MUST start retransmission timer even if it is a pure ACK and MUST retransmit the same segment when the timer is expired. Note that the 3rd segment does not have to be a pure ACK segment. If an endpoint has data to be sent, it can send it with the segment.

When a responder receives 2 bytes length Aggregated Option in the 3rd segment, it MUST send back Aggregated Option to acknowledge the arrival of the option. If responder does not need to receive acknowledgement from initiator, it MUST send back a segment with Aggregated Option with the format shown in Figure 4 to indicate the end of option exchanges. If responder need to send additional options that requires acknowledgement, it MUST send 2-bytes Aggregated Option shown in Figure 3 and MUST start retransmission timer for retransmission. An endpoint that receives Fin Aggregated Option MUST not use Aggregated Options in subseqent segments. Similarly, an endpoint that sends Fin Aggregated Option MUST not use Aggregated Options once the segment is acknowledged.

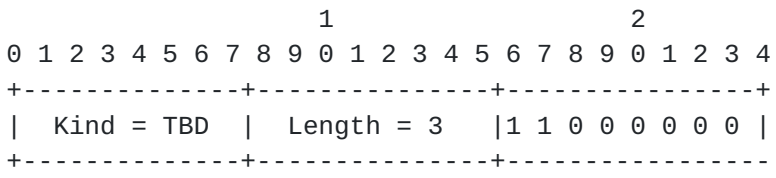

Figure 4: Fin Aggregated Option format

When an endpoint sends a segment with 2 byte Aggregated Option, it MUST retransmit the segment until it receives an ACK for the segment with Aggregated Option (2 byte or Fin Aggregated Option). When it receives an ACK for the segment without Aggregated Option, it MUST ignore and discard the ACK.

3.3. Examples of Option Exchanges with Aggregated Options

We show 3 patterns of option negotiations with Aggregated Option in the following examples: Figure Figure 5 , Figure Figure 6 and Figure Figure 7. In these examples, we use hypothetical option A and B. Option A is an option that does not require additional information such as SACK Permit option. While Option B are an option that needs to exchanges additional parameters. "Non-Agg options" in the examples represent the options that are not aggregated, which are sent with their original formats.

As shown in Figure 5, as option A does not require additional information, exchanging SYN and SYN ACK segments with Aggregated Options are enough for feature negotiation. In this case, initiator sends back 3rd segment with Fin Aggregated Option only to indicate the end of option negotiation.

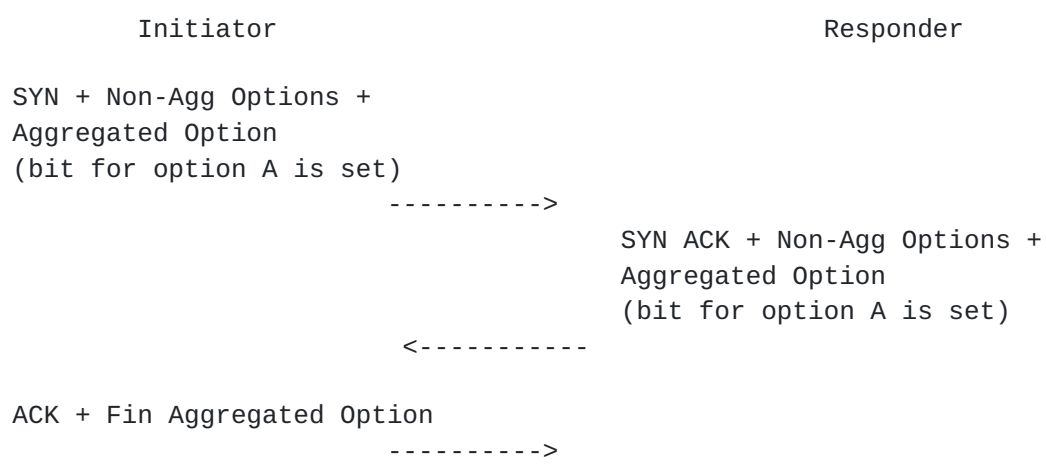

Figure 5: Example of Aggregated Option #1

Figure 6 is an example that uses option B which require additional information exchange. In this case, After exchanging Aggregated Options during SYN exchanges, initiator sends 3rd ACK segment with 2-Byte Aggregated Options in addition to the original form of option B. Similarly, responder sends back the ACK for the segment with 2-Bytes Aggregated Options in addition to the original form of Option B. After that, initiator sends an ACK with Fin Aggregated Option only to indicate the end of option negotiation.

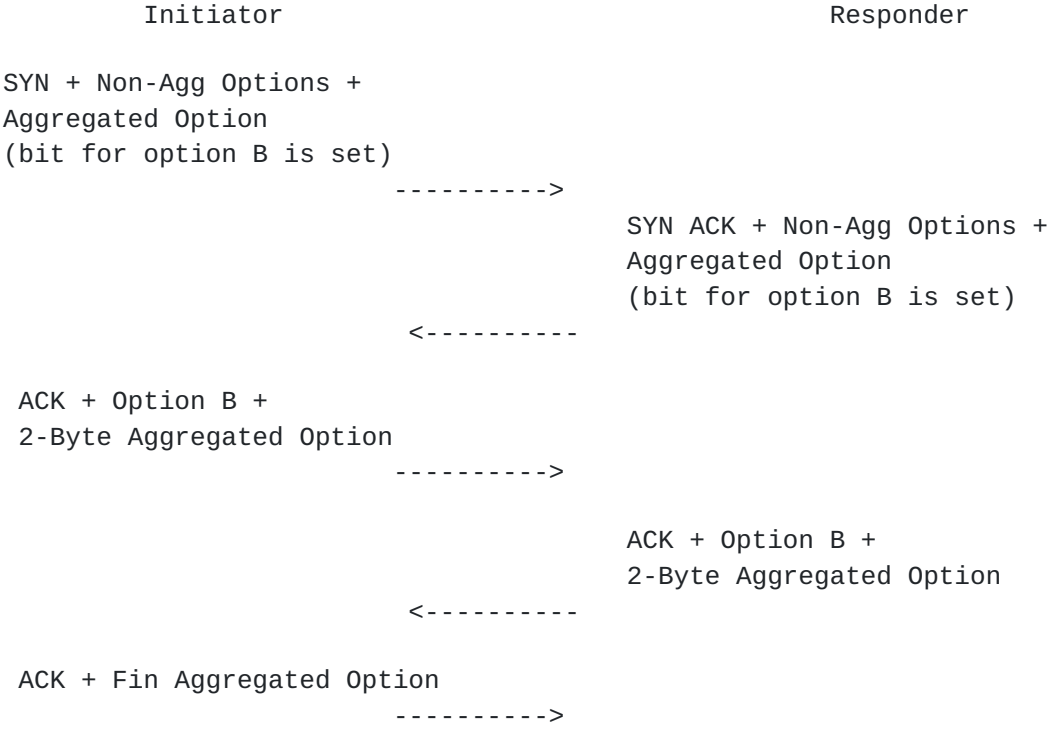

Figure 6: Example of Aggregated Option #2

Figure 7 also uses option B as Figure 6, however, initiator only sends two segments for option negotiations. In this case, when responder sends SYN-ACK, it sends back the segment with original form of Option B. In this case, when responder sends back the ACK for 3rd segment, it sends the segment with Fin Aggregated Option to indicate the end of option negotiation. In this case, while sender utilizes extra option space with Aggregated Option in 3rd segment, responder uses option space only in SYN ACK segment. This use case can be useful when only option space in SYN segment needs to be extended.

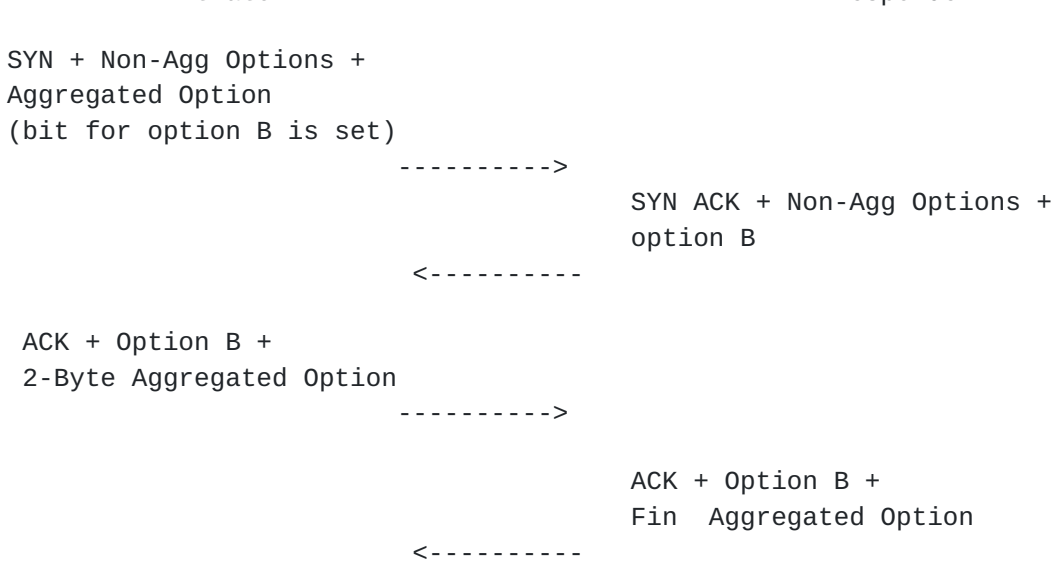

Figure 7: Example of Aggregated Option #3

4. Option Bits Registration

As described in Section Section 3, Aggregated Option has 18 Option bits space and each bit represents a single option ID. The allocation of the Option Bits in Aggregated Option is maintained by IANA [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 a option already has assigned TCP Option Kind Number, one can only request Option bit which will represent the option kind.

5. Discussions

5.1. Incremental Deployments for Aggregated Option

An endpoint MAY send the original form of an option and Aggregated Option for the option in SYN segments if option space is available. This may look redundant, but it can be useful when an endpoint sends a SYN segment to a new destination. In this case, if the peers support the option and Aggregated Option, they can send back Aggregated Option in addition to the response for normal option format. The endpoint cache this information and when it opens another connection to the the same destination with a short interval, it can send only Aggregated Option this time. If an endpoint did not receive SYN ACK when it sent Aggregated Option to a new destination, the retransmitted SYN segment SHOULD NOT include Aggregated Option.

5.2. Middlebox Considerations

As Aggregated Option will be a new option, there is a potential risk that the option will be removed from the segment or the segment that carries the option will be discarded by the middleboxes. While we believe the risk for this risk is low from the past studies [HONDA11], if endpoints try to aggregate well-known TCP options such as SACK-Permit, MSS, TimeStamp to new destinations, it is recommended to follow incremental deployment approach described in the previous section. On the other hand, the risk for the removal of Aggregated Option and new options can be considered mostly in the same level. In this case, endpoints MAY send only the aggregated option to new destinations.

If a middlebox strips Aggregated options in SYN segments but keeps other options unchanged, the proposed scheme can behave safely. When Aggregated Option in SYN segment from initiator is removed on outgoing path, both endpoint will not activate any features in the Aggregated Option. When Aggregated Option is removed on return path, there will be a situation where the responder enables the features in Aggregated Option while initiator disables the features. However, the responder still can aware that the Aggregated Option is removed on the return path when it receives the 3rd segments without Aggregate Option. In this case, the responder can decide whether it can continue the connection or reset it based on the features in the Aggregated Option.

5.3. Which Options are Aggregatable

While we think most of options can be aggregated by using Aggregated Option, there will be some limitations. One example would be TCP AO [RFC5925] as it has to carry security information in SYN segments. If an option must carry additional information for the feature like TCP AO, we cannot aggregate the option. TCP Fast Open [RFC7413]

would be another example as it needs to carry cached cookie information in SYN segments. Sending cookie info in the 3rd segments is not desirable from the objective of TCP Fast Open. However, when initiator sends Fast Open Cookie Request, the request can be aggregated as it does not carry additional information. Hence, we still believe that it will be useful to aggregate TCP Fast Open option in some cases.

5.4. Further Extensions

The proposed approach can aggregate 18 options with Aggregated Options. We believe this is enough number for the time being based on the current usages of TCP options. It is also possible to accommodate more options by using Group id 4 which is currently reserved for future extensions.

Aggregated Option can extend available space for TCP options in SYN segments by utilizing 3rd segments and its acknowledgement. However, there might be a situation where this is still not enough to accommodate all TCP options that an endpoint would like to activate. In this case, one possible approach is to utilize TCP Extended Data Offset Option (EDO) [I-D.ietf-tcpm-tcp-edo]. As EDO supported Option does not carry any information, it can be aggregated in Aggregated Option. Once both endpoints agree to use the feature by SYN exchange, an endpoint can start using EDO to extend option space in the 3rd segments and its acknowledgement. This approach will create more space for options in SYN segments although further discussions will be required for more detailed design.

6. Security Considerations

We believe Aggregated Option maintains the same level of security as other TCP options does, but further discussions will be required.

7. 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.

7.1. 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

Table 2: Aggregated Option Format

7.2. Option Bits Registry for Aggregated Option

This document also requests to create a "Aggregated Option Identifiers" registry in IANA registries. The registry maintains 24 records which are mapped to the TCP Option Kind Number Records in [TCPParameters] The 24 records are divided into 4 groups so that each group contains 6 records.

8. References

8.1. 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/info/rfc2119>.

8.2. Informative References

[HONDA11] Honda, M., "Is it still possible to extend TCP?", IMC 2011 , November 2011.

[I-D.allman-tcpx2-hack] Allman, M., "TCPx2: Don't Fence Me In", draft-allman-tcpx2-hack-00 (work in progress), May 2006.

[I-D.briscoe-tcpm-inner-space] Briscoe, B., "Inner Space for TCP Options", draft-briscoe-tcpm-inner-space-01 (work in progress), October 2014.

[I-D.eddy-tcp-loo] Eddy, W. and A. Langley, "Extending the Space Available for TCP Options", draft-eddy-tcp-loo-04 (work in progress), July 2008.

[I-D.ietf-tcpm-tcp-edo] Touch, J. and W. Eddy, "TCP Extended Data Offset Option", draft-ietf-tcpm-tcp-edo-10 (work in progress), July 2018.

[I-D.touch-tcpm-tcp-syn-ext-opt]
Touch, J. and T. Faber, "TCP SYN Extended Option Space Using an Out-of-Band Segment", draft-touch-tcpm-tcp-syn-ext-opt-09 (work in progress), July 2018.

[I-D.yourtchenko-tcp-loic]
Yourtchenko, A., "Introducing TCP Long Options by Invalid Checksum", draft-yourtchenko-tcp-loic-00 (work in progress), April 2011.

[IANA] "IANA Home Page", <https://www.iana.org/>.

[RFC0793] Postel, J., "Transmission Control Protocol", STD 7, RFC 793, DOI 10.17487/RFC0793, September 1981, <https://www.rfc-editor.org/info/rfc793>.

[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/info/rfc2018>.

[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/info/rfc5925>.

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

[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/info/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/info/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/info/rfc8684>.

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

Author's Address

Yoshifumi Nishida
Amazon Web Services
410 Terry Avenue North
Seattle, WA 98109
USA

Email: nishida@wide.ad.jp