Extended Option Space for TCP

Status of this Memo

This document is an Internet-Draft.

By submitting this Internet-Draft, we certify that any applicable patent or other IPR claims of which we are aware have been disclosed, or will be disclosed, and any of which we become aware will be disclosed, in accordance with RFC 3668 (BCP 79).

By submitting this Internet-Draft, we accept the provisions of Section 3 of RFC 3667 (BCP 78).

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

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 a "work in progress."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/1id-abstracts.html

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html

Copyright Notice

Copyright (C) The Internet Society (2004). All Rights Reserved.

Abstract

This memo describes a reinterpretation of the TCP Data Offset field, affecting the previously illegal code points 0-4, that allows
endpoints to fit more than 40 bytes of option into TCP segments.
1. Introduction

The TCP datagram format has space for up to 40 bytes of TCP options [RFC 793]. Although this is adequate in most cases, a combination of options such as TCP MD5 [RFC 2385], SACK (Selective Acknowledgement) [RFC 2018], and Timestamp [RFC 1323] will not fit in the currently available option space. In fact, SACK alone could take up more space than is available, given a sufficiently complex loss pattern. A mechanism supporting larger option space might support currently illegal option combinations, simplify the deployment of any future TCP options, and discourage kludges that try to fit too much data into too little option space. Further motivation and discussion TBA.

The amount of space used for options is determined by the TCP header’s 4-bit Data Offset field, or DO. This number equals the offset of application data relative to the start of the TCP header, measured in 32-bit words. The fixed portion of the TCP header is 20 bytes long, so 5 is the smallest legal value for DO; it indicates the absence of options. The largest possible value, 15, indicates a data offset of 60 bytes, and thus 40 bytes of option space. The values 0 through 4 are currently illegal. The proposed mechanism uses these code points to indicate extended option space, taking more than 40 bytes.

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 [RFC 2119].

2. Mechanism

A TCP implementing this Internet-Draft MUST interpret the TCP header’s DO field according to the following table. The interpretation of values 5-15 is identical to that of [RFC 793].

<table>
<thead>
<tr>
<th>DO</th>
<th>Data Offset</th>
<th>Option Space</th>
<th>Min. TCP Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>68</td>
<td>48</td>
<td>68</td>
</tr>
<tr>
<td>1</td>
<td>84</td>
<td>64</td>
<td>84</td>
</tr>
<tr>
<td>2</td>
<td>148</td>
<td>128</td>
<td>148</td>
</tr>
<tr>
<td>3</td>
<td>276</td>
<td>256</td>
<td>276</td>
</tr>
<tr>
<td>4</td>
<td>infinity</td>
<td>whole packet</td>
<td>20</td>
</tr>
<tr>
<td>5-15</td>
<td>DO*4</td>
<td>(DO*4)-20</td>
<td>DO*4</td>
</tr>
</tbody>
</table>

A segment’s TCP length MUST equal or exceed the Min. TCP Length value indicated by its DO field. A receiving TCP MUST ignore any segment that is too short.
TCP segments with DO between 0 and 4 are called extended segments.

2.1. Requesting Extended Segments with SYN

Extended segments MUST NOT be sent unless their use was approved during the TCP three-way handshake. Approval happens when an extended segment (here, the SYN) is acknowledged by another extended segment (the SYNACK).

An endpoint performing active open indicates its desire to use extended segments by sending an extended SYN, that is, a SYN with DO < 5. If an extended SYNACK arrives in response, the endpoint will send an ACK and continue, using extended and nonextended segments as appropriate. If the connection attempt fails (through a timeout, ICMP destination unreachable, or received TCP RST), or the received SYNACK is not extended, the active endpoint MUST try again with a non-extended SYN. Unless the connection attempt failed through a RST, the active endpoint MUST clean up any remote state before retrying, by sending a RST and waiting at least a short interval (roughly 1 round-trip time, or 100 ms, if no RTT is available) to discourage packet reordering.

A listening endpoint receiving an extended SYN MUST either respond with an extended SYNACK (to allow the use of extended segments), or reset the connection with a non-extended RST (to prevent their use).

Requiring a full handshake to approve the use of extended segments has the side effect of ensuring that any middleboxes on both parts of the path can handle extended segments (or at least won't drop them).

The procedures described in this section can delay connection establishment, or definitive connection refusal, by up to a SYN timeout (on the order of 3 seconds).

2.2. Requesting Extended Segments with SYNACK

A passive, listening endpoint MAY also request the use of extended segments, by sending an extended SYNACK in response to a non-extended SYN. Approval is granted if the response ACK is extended. This procedure is riskier than requesting extended segments on the SYN, however. An active endpoint with a "legacy" implementation might reset the connection in response to the extended SYNACK, and not retry. Furthermore, a listening endpoint implementing this procedure must distinguish SYN transmissions from retransmissions, preventing the use of SYN cookies [SYNCOOKIES].
A listening endpoint receiving a non-extended SYN MAY respond with an extended SYNACK to request the use of extended segments. If an extended ACK arrives in response, the endpoint will continue using extended and nonextended segments as appropriate. If the extended SYNACK transmission fails (a timeout occurs, a retransmitted non-extended SYN is received, or a non-extended RST is received), it MUST try again with a non-extended SYNACK. If a non-extended ACK is received, it MUST send a non-extended SYNACK retransmission; the hope is that the active endpoint will use any options specified on the retransmission.

An active-open endpoint that sent a non-extended SYN, but received an extended SYNACK, MUST either respond with an extended ACK (to allow the use of extended segments), or reset the connection with a non-extended RST (to prevent their use).

3. Stability Considerations

Existing "legacy" TCP implementations -- both those in end hosts, and those in middleboxes such as firewalls -- clearly will not process extended segments according to this memo. On encountering an extended segment, legacy implementations might drop the segment as erroneous, act as if the segment had no options, reset the connection, or even conceivably crash. Even if endpoints were able to complete an extended-segment handshake, a path change (perhaps induced by mobility) might introduce a legacy middlebox into the connection, leading to possible connection reset. For these reasons, TCP connections SHOULD NOT use extended segments, or the extended segment handshake, unless it is considered required. APIs SHOULD let applications allow the use of extended segments; this API SHOULD be off by default.

Legacy endpoints that treat extended segments as if they have DO 5 are particularly problematic. The risk is that any options on the packet, including the mandatory MSS option, will be ignored; and that any options on retransmitted SYN or SYNACK packets will likewise be ignored. This risk should be investigated further. Modern open-source operating systems, at least, appear to drop extended segments.

4. Security Considerations

TCP implementations that follow this document will respond more slowly to some received RSTs, specifically those sent in response to extended SYNs and SYNACKs. Endpoints that implement the algorithm in Section 2.2 cannot use SYN cookies to protect against SYN-flood denial-of-service attacks. (Others?)
5. Acknowledgements

This mechanism was developed in conversation with Mark Allman, following conversation with Wes Eddy.

Normative References


Informative References


Full Copyright Statement

Copyright (C) The Internet Society 2004. This document is subject to the rights, licenses and restrictions contained in BCP 78, and except as set forth therein, the authors retain all their rights.

This document and the information contained herein are provided on an "AS IS" basis and THE CONTRIBUTOR, THE ORGANIZATION HE/SHE REPRESENTS OR IS SPONSORED BY (IF ANY), THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Intellectual Property

The IETF has been notified of intellectual property rights claimed in regard to some or all of the specification contained in this document. For more information consult the online list of claimed rights.

The IETF takes no position regarding the validity or scope of any Intellectual Property Rights or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; nor does it represent that it has made any independent effort to identify any such rights. Information on the procedures with respect to rights in RFC documents can be found in BCP 78 and BCP 79.

Copies of IPR disclosures made to the IETF Secretariat and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this specification can be obtained from the IETF on-line IPR repository at http://www.ietf.org/ipr.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights that may cover technology that may be required to implement this standard. Please address the information to the IETF at ietf-