Internet Engineering Task Force (IETF)
Request for Comments: 6691
Updates: 879, 2385
Category: Informational
ISSN: 2070-1721

D. Borman
Quantum Corporation

July 2012

# TCP Options and Maximum Segment Size (MSS)

## Abstract

This memo discusses what value to use with the TCP Maximum Segment Size (MSS) option, and updates RFC 879 and RFC 2385.

## Status of This Memo

This document is not an Internet Standards Track specification; it is published for informational purposes.

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). Not all documents approved by the IESG are a candidate for any level of Internet Standard; see 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/rfc6691.

## 1. Introduction

There has been some confusion as to what value to use with the TCP MSS option when using IP and TCP options. RFC 879 [RFC879] states:

> The MSS counts only data octets in the segment, it does not count the TCP header or the IP header.

This statement is unclear about what to do about IP and TCP options, since they are part of the headers. RFC 1122 [RFC1122] clarified the MSS option, which is discussed in Appendix A, but there still seems to be some confusion.

### 1.1. 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 RFC 2119 [RFC2119].

## 2. The Short Statement

When calculating the value to put in the TCP MSS option, the MTU value SHOULD be decreased by only the size of the fixed IP and TCP headers and SHOULD NOT be decreased to account for any possible IP or TCP options; conversely, the sender MUST reduce the TCP data length to account for any IP or TCP options that it is including in the packets that it sends. The rest of this document just expounds on that statement, and the goal is to avoid IP-level fragmentation of TCP packets.

The size of the fixed TCP header is 20 bytes [RFC793], the size of the fixed IPv4 header is 20 bytes [RFC791], and the size of the fixed IPv6 header is 40 bytes [RFC2460]. The determination of what MTU value should be used, especially in the case of multi-homed hosts, is beyond the scope of this document.

## 3. MSS in Other RFCs

### 3.1. RFC 879

RFC 879 [RFC879] discusses the MSS option and other topics. In the Introduction, it states:

> THE TCP MAXIMUM SEGMENT SIZE IS THE IP MAXIMUM DATAGRAM SIZE MINUS FORTY.

In Section 13, it states:

> The definition of the MSS option can be stated:
>
> > The maximum number of data octets that may be received by the sender of this TCP option in TCP segments with no TCP header options transmitted in IP datagrams with no IP header options.

These are both correct. However, in the next paragraph -- in Section 14 -- it then confuses this by stating that the consequence is:

> When TCP is used in a situation when either the IP or TCP headers are not minimum and yet the maximum IP datagram that can be received remains 576 octets then the TCP Maximum Segment Size option must be used to reduce the limit on data octets allowed in a TCP segment.
>
> > For example, if the IP Security option (11 octets) were in use and the IP maximum datagram size remained at 576 octets, then the TCP should send the MSS with a value of 525 (536-11).

That is incorrect. The simpler and more correct statement would be:

> When TCP is used in a situation where either the IP or TCP headers are not minimum, the sender must reduce the amount of TCP data in any given packet by the number of octets used by the IP and TCP options.

## 3.2. RFC 2385

RFC 2385 [RFC2385] incorrectly states, in Section 4.3:

> As with other options that are added to every segment, the size of the MD5 option must be factored into the MSS offered to the other side during connection negotiation. Specifically, the size of the header to subtract from the MTU (whether it is the MTU of the outgoing interface or IP’s minimal MTU of 576 bytes) is now at least 18 bytes larger.

This is incorrect. The value for the MSS option is only adjusted by the fixed IP and TCP headers.

## 4. Clarification from the TCP Large Windows Mailing List

The initial clarification was sent to the TCP Large Windows mailing list in 1993 [Borman93]; this section is based on that message.

The MSS value to be sent in an MSS option should be equal to the effective MTU minus the fixed IP and TCP headers. By ignoring both IP and TCP options when calculating the value for the MSS option, if there are any IP or TCP options to be sent in a packet, then the sender must decrease the size of the TCP data accordingly. The reason for this can be seen in the following table:

| | MSS is adjusted to include options | MSS isn't adjusted to include options |
|---|---|---|
| Sender adjusts packet length for options | Packets are too short | Packets are the correct length |
| Sender doesn't adjust packet length for options | Packets are the correct length | Packets are too long |

The goal is to not send IP datagrams that have to be fragmented, and packets sent with the constraints in the lower right of this grid will cause IP fragmentation. Since the sender doesn't know if the received MSS option was adjusted to include options, the only way to guarantee that the packets are not too long is for the data sender to decrease the TCP data length by the size of the IP and TCP options. It follows, then, that since the sender will be adjusting the TCP data length when sending IP and TCP options, there is no need to include the IP and TCP option lengths in the MSS value.

Another argument against including IP or TCP options in the determination of the MSS value is that the MSS is a fixed value, and by their very nature the lengths of IP and TCP options are variable, so the MSS value can never accurately reflect all possible IP and TCP option combinations. The only one that knows for sure how many IP and TCP options are in any given packet is the sender; hence, the sender should be doing the adjustment to the TCP data length to account for any IP and TCP options.

# 5. Additional Considerations

## 5.1. Path MTU Discovery

The TCP MSS option specifies an upper bound for the size of packets that can be received. Hence, setting the value in the MSS option too small can impact the ability for Path MTU Discovery to find a larger path MTU. For more information on Path MTU Discovery, see:

- "Path MTU Discovery" [RFC1191]
- "TCP Problems with Path MTU Discovery" [RFC2923]
- "Packetization Layer Path MTU Discovery" [RFC4821]

## 5.2. Interfaces with Variable MSS Values

The effective MTU can sometimes vary, as when used with variable compression, e.g., RObust Header Compression (ROHC) [RFC5795]. It is tempting for TCP to want to advertise the largest possible MSS, to support the most efficient use of compressed payloads. Unfortunately, some compression schemes occasionally need to transmit full headers (and thus smaller payloads) to resynchronize state at their endpoint compressors/decompressors. If the largest MTU is used to calculate the value to advertise in the MSS option, TCP retransmission may interfere with compressor resynchronization.

As a result, when the effective MTU of an interface varies, TCP SHOULD use the smallest effective MTU of the interface to calculate the value to advertise in the MSS option.

## 5.3. IPv6 Jumbograms

In order to support TCP over IPv6 jumbograms, implementations need to be able to send TCP segments larger than 64K. RFC 2675 [RFC2675] defines that a value of 65,535 is to be treated as infinity, and Path MTU Discovery [RFC1981] is used to determine the actual MSS.

## 5.4. Avoiding Fragmentation

Packets that are too long will either be fragmented or dropped. If packets are fragmented, intermediary firewalls or middle boxes may drop the fragmented packets. In either case, when packets are dropped, the connection can fail; hence, it is best to avoid generating fragments.

## 6. Security Considerations

This document clarifies how to determine what value should be used for the MSS option and does not introduce any new security concerns.

## 7. References

### 7.1. Normative References

[RFC791] Postel, J., "Internet Protocol", STD 5, RFC 791, September 1981.

[RFC793] Postel, J., "Transmission Control Protocol", STD 7, RFC 793, September 1981.

[RFC879] Postel, J., "The TCP Maximum Segment Size and Related Topics", RFC 879, November 1983.

[RFC1122] Braden, R., Ed., "Requirements for Internet Hosts - Communication Layers", STD 3, RFC 1122, October 1989.

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

[RFC2460] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", RFC 2460, December 1998.

[RFC2675] Borman, D., Deering, S., and R. Hinden, "IPv6 Jumbograms", RFC 2675, August 1999.

### 7.2. Informative References

[Borman93] Borman, D., "TCP MSS & Timestamps", Message to the tcplw mailing list, January 7, 1993.

[RFC1191] Mogul, J. and S. Deering, "Path MTU discovery", RFC 1191, November 1990.

[RFC1981] McCann, J., Deering, S., and J. Mogul, "Path MTU Discovery for IP version 6", RFC 1981, August 1996.

[RFC2385] Heffernan, A., "Protection of BGP Sessions via the TCP MD5 Signature Option", RFC 2385, August 1998.

[RFC2923] Lahey, K., "TCP Problems with Path MTU Discovery", RFC 2923, September 2000.

[RFC4821] Mathis, M. and J. Heffner, "Packetization Layer Path MTU Discovery", RFC 4821, March 2007.

[RFC5795] Sandlund, K., Pelletier, G., and L-E. Jonsson, "The RObust Header Compression (ROHC) Framework", RFC 5795, March 2010.

## Appendix A. Details from RFC 793 and RFC 1122

RFC 793 [RFC793] defines the MSS option as follows:

> Maximum Segment Size Option Data: 16 bits
>
> > If this option is present, then it communicates the maximum receive segment size at the TCP which sends this segment. This field must only be sent in the initial connection request (i.e., in segments with the SYN control bit set). If this option is not used, any segment size is allowed.

RFC 1122 [RFC1122] provides additional clarification in Section 4.2.2.6, on pages 85-86. First, it changes the default behavior when the MSS option is not present:

> If an MSS option is not received at connection setup, TCP MUST assume a default send MSS of 536 (576-40) [TCP:4].

Then, it clarifies how to determine the value to use in the MSS option:

> The MSS value to be sent in an MSS option must be less than or equal to:
>
> > MMS_R - 20
>
> where MMS_R is the maximum size for a transport-layer message that can be received (and reassembled). TCP obtains MMS_R and MMS_S from the IP layer; see the generic call GET_MAXSIZES in Section 3.4.

What is implied in RFC 1122, but not explicitly stated, is that the 20 is the size of the fixed TCP header. The definition of MMS_R is found in Section 3.3.2, on page 57:

> MMS_R is given by:
>
> > MMS_R = EMTU_R - 20
>
> since 20 is the minimum size of an IP header.

and on page 56 (also Section 3.3.2):

> We designate the largest datagram size that can be reassembled by EMTU_R ("Effective MTU to receive"); this is sometimes called the "reassembly buffer size". EMTU_R MUST be greater than or equal to 576, SHOULD be either configurable or indefinite, and SHOULD be greater than or equal to the MTU of the connected network(s).

What should be noted here is that EMTU_R is the largest datagram size that can be reassembled, not the largest datagram size that can be received without fragmentation. Taking this literally, since most modern TCP/IP implementations can reassemble a full 64K IP packet, implementations should be using 65535 - 20 - 20, or 65495, for the MSS option. But there is more to it than that. RFC 1122 also states, on page 86:

> The choice of TCP segment size has a strong effect on performance. Larger segments increase throughput by amortizing header size and per-datagram processing overhead over more data bytes; however, if the packet is so large that it causes IP fragmentation, efficiency drops sharply if any fragments are lost [IP:9].

Since it is guaranteed that any IP datagram that is larger than the MTU of the connected network will have to be fragmented to be received, implementations ignore the "greater than or" part of "SHOULD be greater than or equal to the MTU of the connected network(s)". Thus, the MSS value to be sent in an MSS option must be less than or equal to:

> EMTU_R - FixedIPhdrsize - FixedTCPhdrsize

where FixedTCPhdrsize is 20, and FixedIPhdrsize is 20 for IPv4 and 40 for IPv6.

## Author’s Address

David Borman
Quantum Corporation
Mendota Heights, MN 55120

EMail: david.borman@quantum.com