                    Path MTU Discovery for IP version 6

Abstract

   This document describes Path MTU Discovery (PMTUD) for IP version 6.
   It is largely derived from RFC 1191, which describes Path MTU
   Discovery for IP version 4. It obsoletes RFC 1981.

Status of This Memo

   This is an Internet Standards Track document.

   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). Further information on
   Internet Standards is available in Section 2 of RFC 7841.

   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/rfc8201.

Copyright Notice

Table of Contents

1. Introduction

   When one IPv6 node has a large amount of data to send to another
   node, the data is transmitted in a series of IPv6 packets. These
   packets can have a size less than or equal to the Path MTU (PMTU).
   Alternatively, they can be larger packets that are fragmented into a
   series of fragments each with a size less than or equal to the PMTU.

   It is usually preferable that these packets be of the largest size
   that can successfully traverse the path from the source node to the
   destination node without the need for IPv6 fragmentation. This
   packet size is referred to as the Path MTU, and it is equal to the
   minimum link MTU of all the links in a path. This document defines a
   standard mechanism for a node to discover the PMTU of an arbitrary
   path.

   IPv6 nodes should implement Path MTU Discovery in order to discover
   and take advantage of paths with PMTU greater than the IPv6 minimum
   link MTU [RFC8200]. A minimal IPv6 implementation (e.g., in a boot
   ROM) may choose to omit implementation of Path MTU Discovery.

   Nodes not implementing Path MTU Discovery must use the IPv6 minimum
   link MTU defined in [RFC8200] as the maximum packet size. In most
   cases, this will result in the use of smaller packets than necessary,
   because most paths have a PMTU greater than the IPv6 minimum link
   MTU. A node sending packets much smaller than the Path MTU allows is
   wasting network resources and probably getting suboptimal throughput.

   Nodes implementing Path MTU Discovery and sending packets larger than
   the IPv6 minimum link MTU are susceptible to problematic connectivity
   if ICMPv6 [ICMPv6] messages are blocked or not transmitted. For
   example, this will result in connections that complete the TCP three-
   way handshake correctly but then hang when data is transferred. This
   state is referred to as a black-hole connection [RFC2923]. Path MTU
   Discovery relies on ICMPv6 Packet Too Big (PTB) to determine the MTU
   of the path.

   An extension to Path MTU Discovery defined in this document can be
   found in [RFC4821]. RFC 4821 defines a method for Packetization
   Layer Path MTU Discovery (PLPMTUD) designed for use over paths where
   delivery of ICMPv6 messages to a host is not assured.

   Note: This document is an update to [RFC1981] that was published
   prior to [RFC2119] being published. Consequently, although RFC 1981
   used the "should/must" style language in upper and lower case, this
   document does not cite the RFC 2119 definitions and only uses lower
   case for these words.

2. Terminology

    node   a device that implements IPv6.

    router   a node that forwards IPv6 packets not explicitly
                         addressed to itself.

    host   any node that is not a router.

    upper layer   a protocol layer immediately above IPv6.
                         Examples are transport protocols such as TCP and
                         UDP, control protocols such as ICMPv6, routing
                         protocols such as OSPF, and internet-layer or
                         lower-layer protocols being "tunneled" over
                         (i.e., encapsulated in) IPv6 such as Internetwork
                         Packet Exchange (IPX), AppleTalk, or IPv6 itself.

    link   a communication facility or medium over which
                         nodes can communicate at the link layer, i.e.,
                         the layer immediately below IPv6. Examples are
                         Ethernets (simple or bridged); PPP links; X.25,
                         Frame Relay, or ATM networks; and internet-layer
                         or higher-layer "tunnels", such as tunnels over
                         IPv4 or IPv6 itself.

    interface   a node's attachment to a link.

    address   an IPv6-layer identifier for an interface or a
                         set of interfaces.

    packet   an IPv6 header plus payload. The packet can have
                         a size less than or equal to the PMTU.
                         Alternatively, this can be a larger packet that
                         is fragmented into a series of fragments each
                         with a size less than or equal to the PMTU.

    link MTU   the maximum transmission unit, i.e., maximum
                         packet size in octets, that can be conveyed in
                         one piece over a link.

    path   the set of links traversed by a packet between a
                         source node and a destination node.

    path MTU   the minimum link MTU of all the links in a path
                         between a source node and a destination node.

    PMTU   path MTU.

Path MTU Discovery   the process by which a node learns the PMTU of a
                     path.

EMTU_S   Effective MTU for sending; used by upper-layer
                     protocols to limit the size of IP packets they
                     queue for sending [RFC6691] [RFC1122].

EMTU_R   Effective MTU for receiving; the largest packet
                     that can be reassembled at the receiver
                     [RFC1122].

flow   a sequence of packets sent from a particular
                     source to a particular (unicast or multicast)
                     destination for which the source desires special
                     handling by the intervening routers.

flow id   a combination of a source address and a non-zero
                     flow label.

3. Protocol Overview

This memo describes a technique to dynamically discover the PMTU of a
path. The basic idea is that a source node initially assumes that
the PMTU of a path is the (known) MTU of the first hop in the path.
If any of the packets sent on that path are too large to be forwarded
by some node along the path, that node will discard them and return
ICMPv6 Packet Too Big messages. Upon receipt of such a message, the
source node reduces its assumed PMTU for the path based on the MTU of
the constricting hop as reported in the Packet Too Big message. The
decreased PMTU causes the source to send smaller packets or change
EMTU_S to cause the upper layer to reduce the size of IP packets it
sends.

The Path MTU Discovery process ends when the source node's estimate
of the PMTU is less than or equal to the actual PMTU. Note that
several iterations of the packet-sent/Packet-Too-Big-message-received
cycle may occur before the Path MTU Discovery process ends, as there
may be links with smaller MTUs further along the path.

Alternatively, the node may elect to end the discovery process by
ceasing to send packets larger than the IPv6 minimum link MTU.

The PMTU of a path may change over time, due to changes in the
routing topology. Reductions of the PMTU are detected by Packet Too
Big messages. To detect increases in a path's PMTU, a node
periodically increases its assumed PMTU. This will almost always
result in packets being discarded and Packet Too Big messages being

generated, because in most cases the PMTU of the path will not have
changed. Therefore, attempts to detect increases in a path's PMTU
should be done infrequently.

Path MTU Discovery supports multicast as well as unicast
destinations. In the case of a multicast destination, copies of a
packet may traverse many different paths to many different nodes.
Each path may have a different PMTU, and a single multicast packet
may result in multiple Packet Too Big messages, each reporting a
different next-hop MTU. The minimum PMTU value across the set of
paths in use determines the size of subsequent packets sent to the
multicast destination.

Note that Path MTU Discovery must be performed even in cases where a
node "thinks" a destination is attached to the same link as itself,
as it might have a PMTU lower than the link MTU. In a situation such
as when a neighboring router acts as proxy [ND] for some destination,
the destination can appear to be directly connected, but it is in
fact more than one hop away.

4. Protocol Requirements

   As discussed in Section 1, IPv6 nodes are not required to implement
   Path MTU Discovery. The requirements in this section apply only to
   those implementations that include Path MTU Discovery.

   Nodes should appropriately validate the payload of ICMPv6 PTB
   messages to ensure these are received in response to transmitted
   traffic (i.e., a reported error condition that corresponds to an IPv6
   packet actually sent by the application) per [ICMPv6].

   If a node receives a Packet Too Big message reporting a next-hop MTU
   that is less than the IPv6 minimum link MTU, it must discard it. A
   node must not reduce its estimate of the Path MTU below the IPv6
   minimum link MTU on receipt of a Packet Too Big message.

   When a node receives a Packet Too Big message, it must reduce its
   estimate of the PMTU for the relevant path, based on the value of the
   MTU field in the message. The precise behavior of a node in this
   circumstance is not specified, since different applications may have
   different requirements, and since different implementation
   architectures may favor different strategies.

   After receiving a Packet Too Big message, a node must attempt to
   avoid eliciting more such messages in the near future. The node must
   reduce the size of the packets it is sending along the path. Using a
   PMTU estimate larger than the IPv6 minimum link MTU may continue to
   elicit Packet Too Big messages. Because each of these messages (and

the dropped packets they respond to) consume network resources, nodes
using Path MTU Discovery must detect decreases in PMTU as fast as
possible.

Nodes may detect increases in PMTU, but because doing so requires
sending packets larger than the current estimated PMTU, and because
the likelihood is that the PMTU will not have increased, this must be
done at infrequent intervals. An attempt to detect an increase (by
sending a packet larger than the current estimate) must not be done
less than 5 minutes after a Packet Too Big message has been received
for the given path. The recommended setting for this timer is twice
its minimum value (10 minutes).

A node must not increase its estimate of the Path MTU in response to
the contents of a Packet Too Big message. A message purporting to
announce an increase in the Path MTU might be a stale packet that has
been floating around in the network, a false packet injected as part
of a denial-of-service (DoS) attack, or the result of having multiple
paths to the destination, each with a different PMTU.

5. Implementation Issues

   This section discusses a number of issues related to the
   implementation of Path MTU Discovery. This is not a specification,
   but rather a set of notes provided as an aid for implementers.

   The issues include:

   - What layer or layers implement Path MTU Discovery?

   - How is the PMTU information cached?

   - How is stale PMTU information removed?

   - What must transport and higher layers do?

5.1. Layering

   In the IP architecture, the choice of what size packet to send is
   made by a protocol at a layer above IP. This memo refers to such a
   protocol as a "packetization protocol". Packetization protocols are
   usually transport protocols (for example, TCP) but can also be
   higher-layer protocols (for example, protocols built on top of UDP).

   Implementing Path MTU Discovery in the packetization layers
   simplifies some of the inter-layer issues but has several drawbacks:
   the implementation may have to be redone for each packetization
   protocol, it becomes hard to share PMTU information between different

packetization layers, and the connection-oriented state maintained by
some packetization layers may not easily extend to save PMTU
information for long periods.

It is therefore suggested that the IP layer store PMTU information
and that the ICMPv6 layer process received Packet Too Big messages.
The packetization layers may respond to changes in the PMTU by
changing the size of the messages they send. To support this
layering, packetization layers require a way to learn of changes in
the value of MMS_S, the "maximum send transport-message size"
[RFC1122].

MMS_S is a transport message size calculated by subtracting the size
of the IPv6 header (including IPv6 extension headers) from the
largest IP packet that can be sent, EMTU_S. MMS_S is limited by a
combination of factors, including the PMTU, support for packet
fragmentation and reassembly, and the packet reassembly limit (see
"Fragment Header", Section 4.5 of [RFC8200]). When source
fragmentation is available, EMTU_S is set to EMTU_R, as indicated by
the receiver using an upper-layer protocol or based on protocol
requirements (1500 octets for IPv6). When a message larger than PMTU
is to be transmitted, the source creates fragments, each limited by
PMTU. When source fragmentation is not desired, EMTU_S is set to
PMTU, and the upper-layer protocol is expected to either perform its
own fragmentation and reassembly or otherwise limit the size of its
messages accordingly.

However, packetization layers are encouraged to avoid sending
messages that will require source fragmentation (for the case against
fragmentation, see [FRAG]).

5.2. Storing PMTU Information

Ideally, a PMTU value should be associated with a specific path
traversed by packets exchanged between the source and destination
nodes. However, in most cases a node will not have enough
information to completely and accurately identify such a path.
Rather, a node must associate a PMTU value with some local
representation of a path. It is left to the implementation to select
the local representation of a path. For nodes with multiple
interfaces, Path MTU information should be maintained for each IPv6
link.

In the case of a multicast destination address, copies of a packet
may traverse many different paths to reach many different nodes. The
local representation of the "path" to a multicast destination must
represent a potentially large set of paths.

Minimally, an implementation could maintain a single PMTU value to be
used for all packets originated from the node. This PMTU value would
be the minimum PMTU learned across the set of all paths in use by the
node. This approach is likely to result in the use of smaller
packets than is necessary for many paths. In the case of multipath
routing (e.g., Equal-Cost Multipath Routing (ECMP)), a set of paths
can exist even for a single source and destination pair.

An implementation could use the destination address as the local
representation of a path. The PMTU value associated with a
destination would be the minimum PMTU learned across the set of all
paths in use to that destination. This approach will result in the
use of optimally sized packets on a per-destination basis. This
approach integrates nicely with the conceptual model of a host as
described in [ND]: a PMTU value could be stored with the
corresponding entry in the destination cache.

If flows [RFC8200] are in use, an implementation could use the flow
id as the local representation of a path. Packets sent to a
particular destination but belonging to different flows may use
different paths, as with ECMP, in which the choice of path might
depend on the flow id. This approach might result in the use of
optimally sized packets on a per-flow basis, providing finer
granularity than PMTU values maintained on a per-destination basis.

For source-routed packets (i.e. packets containing an IPv6 Routing
header [RFC8200]), the source route may further qualify the local
representation of a path.

Initially, the PMTU value for a path is assumed to be the (known) MTU
of the first-hop link.

When a Packet Too Big message is received, the node determines which
path the message applies to based on the contents of the Packet Too
Big message. For example, if the destination address is used as the
local representation of a path, the destination address from the
original packet would be used to determine which path the message
applies to.

   Note: if the original packet contained a Routing header, the
   Routing header should be used to determine the location of the
   destination address within the original packet. If Segments Left
   is equal to zero, the destination address is in the Destination
   Address field in the IPv6 header. If Segments Left is greater
   than zero, the destination address is the last address
   (Address[n]) in the Routing header.

The node then uses the value in the MTU field in the Packet Too Big
message as a tentative PMTU value or the IPv6 minimum link MTU if
that is larger, and compares the tentative PMTU to the existing PMTU.
If the tentative PMTU is less than the existing PMTU estimate, the
tentative PMTU replaces the existing PMTU as the PMTU value for the
path.

The packetization layers must be notified about decreases in the
PMTU. Any packetization layer instance (for example, a TCP
connection) that is actively using the path must be notified if the
PMTU estimate is decreased.

   Note: even if the Packet Too Big message contains an Original
   Packet Header that refers to a UDP packet, the TCP layer must be
   notified if any of its connections use the given path.

Also, the instance that sent the packet that elicited the Packet Too
Big message should be notified that its packet has been dropped, even
if the PMTU estimate has not changed, so that it may retransmit the
dropped data.

   Note: An implementation can avoid the use of an asynchronous
   notification mechanism for PMTU decreases by postponing
   notification until the next attempt to send a packet larger than
   the PMTU estimate. In this approach, when an attempt is made to
   SEND a packet that is larger than the PMTU estimate, the SEND
   function should fail and return a suitable error indication. This
   approach may be more suitable to a connectionless packetization
   layer (such as one using UDP), which (in some implementations) may
   be hard to "notify" from the ICMPv6 layer. In this case, the
   normal timeout-based retransmission mechanisms would be used to
   recover from the dropped packets.

It is important to understand that the notification of the
packetization layer instances using the path about the change in the
PMTU is distinct from the notification of a specific instance that a
packet has been dropped. The latter should be done as soon as
practical (i.e., asynchronously from the point of view of the
packetization layer instance), while the former may be delayed until
a packetization layer instance wants to create a packet.

5.3. Purging Stale PMTU Information

Internetwork topology is dynamic; routes change over time. While the
local representation of a path may remain constant, the actual
path(s) in use may change. Thus, PMTU information cached by a node
can become stale.

If the stale PMTU value is too large, this will be discovered almost
immediately once a large enough packet is sent on the path. No such
mechanism exists for realizing that a stale PMTU value is too small,
so an implementation should "age" cached values. When a PMTU value
has not been decreased for a while (on the order of 10 minutes), it
should probe to find if a larger PMTU is supported.

   Note: an implementation should provide a means for changing the
   timeout duration, including setting it to "infinity". For
   example, nodes attached to a link with a large MTU that is then
   attached to the rest of the Internet via a link with a small MTU
   are never going to discover a new non-local PMTU, so they should
   not have to put up with dropped packets every 10 minutes.

5.4. Packetization Layer Actions

   A packetization layer (e.g., TCP) must use the PMTU for the path(s)
   in use by a connection; it should not send segments that would result
   in packets larger than the PMTU, except to probe during PMTU
   Discovery (this probe packet must not be fragmented to the PMTU). A
   simple implementation could ask the IP layer for this value each time
   it created a new segment, but this could be inefficient. An
   implementation typically caches other values derived from the PMTU.
   It may be simpler to receive asynchronous notification when the PMTU
   changes, so that these variables may be also updated.

   A TCP implementation must also store the Maximum Segment Size (MSS)
   value received from its peer, which represents the EMTU_R, the
   largest packet that can be reassembled by the receiver, and must not
   send any segment larger than this MSS, regardless of the PMTU.

   The value sent in the TCP MSS option is independent of the PMTU; it
   is determined by the receiver reassembly limit EMTU_R. This MSS
   option value is used by the other end of the connection, which may be
   using an unrelated PMTU value. See Section 5, "Packet Size Issues",
   and Section 8.3, "Maximum Upper-Layer Payload Size", of [RFC8200] for
   information on selecting a value for the TCP MSS option.

   Reception of a Packet Too Big message implies that a packet was
   dropped by the node that sent the ICMPv6 message. A reliable upper-
   layer protocol will detect this loss by its own means, and recover it
   by its normal retransmission methods. The retransmission could
   result in delay, depending on the loss detection method used by the
   upper-layer protocol. If the Path MTU Discovery process requires
   several steps to find the PMTU of the full path, this could finally
   delay the retransmission by many round-trip times.

Alternatively, the retransmission could be done in immediate response
to a notification that the Path MTU was decreased, but only for the
specific connection specified by the Packet Too Big message. The
packet size used in the retransmission should be no larger than the
new PMTU.

   Note: A packetization layer that determines a probe packet is lost
   needs to adapt the segment size of the retransmission. Using the
   reported size in the last Packet Too Big message, however, can
   lead to further losses as there might be smaller PMTU limits at
   the routers further along the path. This would lead to loss of
   all retransmitted segments and therefore cause unnecessary
   congestion as well as additional packets to be sent each time a
   new router announces a smaller MTU. Any packetization layer that
   uses retransmission is therefore also responsible for congestion
   control of its retransmissions [RFC8085].

A loss caused by a PMTU probe indicated by the reception of a Packet
Too Big message must not be considered as a congestion notification,
and hence the congestion window may not change.

5.5. Issues for Other Transport Protocols

Some transport protocols are not allowed to repacketize when doing a
retransmission. That is, once an attempt is made to transmit a
segment of a certain size, the transport cannot split the contents of
the segment into smaller segments for retransmission. In such a
case, the original segment can be fragmented by the IP layer during
retransmission. Subsequent segments, when transmitted for the first
time, should be no larger than allowed by the Path MTU.

Path MTU Discovery for IPv4 [RFC1191] used NFS as an example of a
UDP-based application that benefits from PMTU Discovery. Since then,
[RFC7530] states that the supported transport layer between NFS and
IP must be an IETF standardized transport protocol that is specified
to avoid network congestion; such transports include TCP, Stream
Control Transmission Protocol (SCTP) [RFC4960], and the Datagram
Congestion Control Protocol (DCCP) [RFC4340]. In this case, the
transport is responsible for ensuring that transmitted segments
(except probes) conform to the Path MTU, including supporting PMTU
Discovery probe transmissions as needed.

5.6. Management Interface

   It is suggested that an implementation provides a way for a system
   utility program to:

   - Specify that Path MTU Discovery not be done on a given path.

   - Change the PMTU value associated with a given path.

   The former can be accomplished by associating a flag with the path;
   when a packet is sent on a path with this flag set, the IP layer does
   not send packets larger than the IPv6 minimum link MTU.

   These features might be used to work around an anomalous situation or
   by a routing protocol implementation that is able to obtain Path MTU
   values.

   The implementation should also provide a way to change the timeout
   period for aging stale PMTU information.

6. Security Considerations

   This Path MTU Discovery mechanism makes possible two DoS attacks,
   both based on a malicious party sending false Packet Too Big messages
   to a node.

      In the first attack, the false message indicates a PMTU much
      smaller than reality. In response, the victim node should never
      set its PMTU estimate below the IPv6 minimum link MTU. A sender
      that falsely reduces to this MTU would observe suboptimal
      performance.

      In the second attack, the false message indicates a PMTU larger
      than reality. If believed, this could cause temporary blockage as
      the victim sends packets that will be dropped by some router.
      Within one round-trip time, the node would discover its mistake
      (receiving Packet Too Big messages from that router), but frequent
      repetition of this attack could cause lots of packets to be
      dropped. A node, however, must not raise its estimate of the PMTU
      based on a Packet Too Big message, so it should not be vulnerable
      to this attack.

   Both of these attacks can cause a black-hole connection, that is, the
   TCP three-way handshake completes correctly but the connection hangs
   when data is transferred.

A malicious party could also cause problems if it could stop a victim
from receiving legitimate Packet Too Big messages, but in this case
there are simpler DoS attacks available.

If ICMPv6 filtering prevents reception of ICMPv6 Packet Too Big
messages, the source will not learn the actual path MTU.
"Packetization Layer Path MTU Discovery" [RFC4821] does not rely upon
network support for ICMPv6 messages and is therefore considered more
robust than standard PMTUD. It is not susceptible to "black-holed"
connections caused by the filtering of ICMPv6 messages. See
[RFC4890] for recommendations regarding filtering ICMPv6 messages.

7. IANA Considerations

   This document does not require any IANA actions.

8. References

8.1. Normative References

   [ICMPv6]   Conta, A., Deering, S., and M. Gupta, Ed., "Internet
              Control Message Protocol (ICMPv6) for the Internet
              Protocol Version 6 (IPv6) Specification", STD 89,
              RFC 4443, DOI 10.17487/RFC4443, March 2006,
              <http://www.rfc-editor.org/info/rfc4443>.

   [RFC8200] Deering, S. and R. Hinden, "Internet Protocol, Version 6
              (IPv6) Specification", STD 86, RFC 8200,
              DOI 10.17487/RFC8200, July 2017,
              <http://www.rfc-editor.org/info/rfc8200>.

8.2. Informative References

   [FRAG]   Kent, C. and J. Mogul, "Fragmentation Considered Harmful",
              In Proc. SIGCOMM '87 Workshop on Frontiers in Computer
              Communications Technology, DOI 10.1145/55483.55524, August
              1987.

   [ND]   Narten, T., Nordmark, E., Simpson, W., and H. Soliman,
              "Neighbor Discovery for IP version 6 (IPv6)", RFC 4861,
              DOI 10.17487/RFC4861, September 2007,
              <http://www.rfc-editor.org/info/rfc4861>.

   [RFC1122] Braden, R., Ed., "Requirements for Internet Hosts -
              Communication Layers", STD 3, RFC 1122,
              DOI 10.17487/RFC1122, October 1989,
              <http://www.rfc-editor.org/info/rfc1122>.

   [RFC1191] Mogul, J. and S. Deering, "Path MTU discovery", RFC 1191,
              DOI 10.17487/RFC1191, November 1990,
              <http://www.rfc-editor.org/info/rfc1191>.

   [RFC1981] McCann, J., Deering, S., and J. Mogul, "Path MTU Discovery
              for IP version 6", RFC 1981, DOI 10.17487/RFC1981, August
              1996, <http://www.rfc-editor.org/info/rfc1981>.

   [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate
              Requirement Levels", BCP 14, RFC 2119,
              DOI 10.17487/RFC2119, March 1997,
              <http://www.rfc-editor.org/info/rfc2119>.

   [RFC2923] Lahey, K., "TCP Problems with Path MTU Discovery",
              RFC 2923, DOI 10.17487/RFC2923, September 2000,
              <http://www.rfc-editor.org/info/rfc2923>.

   [RFC4340] Kohler, E., Handley, M., and S. Floyd, "Datagram
              Congestion Control Protocol (DCCP)", RFC 4340,
              DOI 10.17487/RFC4340, March 2006,
              <http://www.rfc-editor.org/info/rfc4340>.

   [RFC4821] Mathis, M. and J. Heffner, "Packetization Layer Path MTU
              Discovery", RFC 4821, DOI 10.17487/RFC4821, March 2007,
              <http://www.rfc-editor.org/info/rfc4821>.

   [RFC4890] Davies, E. and J. Mohacsi, "Recommendations for Filtering
              ICMPv6 Messages in Firewalls", RFC 4890,
              DOI 10.17487/RFC4890, May 2007,
              <http://www.rfc-editor.org/info/rfc4890>.

   [RFC4960] Stewart, R., Ed., "Stream Control Transmission Protocol",
              RFC 4960, DOI 10.17487/RFC4960, September 2007,
              <http://www.rfc-editor.org/info/rfc4960>.

   [RFC6691] Borman, D., "TCP Options and Maximum Segment Size (MSS)",
              RFC 6691, DOI 10.17487/RFC6691, July 2012,
              <http://www.rfc-editor.org/info/rfc6691>.

   [RFC7530] Haynes, T., Ed. and D. Noveck, Ed., "Network File System
              (NFS) Version 4 Protocol", RFC 7530, DOI 10.17487/RFC7530,
              March 2015, <http://www.rfc-editor.org/info/rfc7530>.

   [RFC8085] Eggert, L., Fairhurst, G., and G. Shepherd, "UDP Usage
              Guidelines", BCP 145, RFC 8085, DOI 10.17487/RFC8085,
              March 2017, <http://www.rfc-editor.org/info/rfc8085>.

Appendix A. Comparison to RFC 1191

   RFC 1981 (obsoleted by this document) was based in large part on RFC
   1191, which describes Path MTU Discovery for IPv4. Certain portions
   of RFC 1191 were not needed in RFC 1981:

   router specification Packet Too Big messages and corresponding
                         router behavior are defined in [ICMPv6]

   Don't Fragment bit   there is no DF bit in IPv6 packets

   TCP MSS discussion   selecting a value to send in the TCP MSS option
                         is discussed in [RFC8200]

   old-style messages   all Packet Too Big messages report the MTU of
                         the constricting link

   MTU plateau tables   not needed because there are no old-style
                         messages

Appendix B. Changes Since RFC 1981

   This document is based on RFC 1981 and has the following changes from
   RFC 1981:

   o Clarified in Section 1, "Introduction", that the purpose of PMTUD
      is to reduce the need for IPv6 fragmentation.

   o Added text to Section 1, "Introduction", about the effects on
      PMTUD when ICMPv6 messages are blocked.

   o Added a "Note" to the introduction to document that this
      specification doesn't cite RFC 2119 and only uses lower case
      "should/must" language. Changed all upper case "should/must" to
      lower case.

   o Added a short summary to Section 1, "Introduction", about PLPMTUD
      and a reference to RFC 4821 that defines it.

   o Aligned text in Section 2, "Terminology", to match current
      packetization layer terminology.

   o Added clarification in Section 4, "Protocol Requirements", that
      nodes should validate the payload of ICMP PTB messages per RFC
      4443, and that nodes should detect decreases in PMTU as fast as
      possible.

   o Removed a "Note" from Section 4, "Protocol Requirements", about a
      Packet Too Big message reporting a next-hop MTU that is less than
      the IPv6 minimum link MTU because this was removed from [RFC8200].

   o Added clarification in Section 5.2, "Storing PMTU Information", to
      discard an ICMPv6 Packet Too Big message if it contains an MTU
      less than the IPv6 minimum link MTU.

   o Added clarification in Section 5.2, "Storing PMTU Information",
      that for nodes with multiple interfaces, Path MTU information
      should be stored for each link.

   o Removed text in Section 5.2, "Storing PMTU Information", about
      Routing Header type 0 (RH0) because it was deprecated by RFC 5095.

   o Removed text about obsolete security classification from
      Section 5.2, "Storing PMTU Information".

   o Changed the title of Section 5.4 to "Packetization Layer Actions"
      and changed the text in the first paragraph to generalize this
      section to cover all packetization layers, not just TCP.

   o Clarified text in Section 5.4, "Packetization Layer Actions", to
      use normal packetization layer retransmission methods.

   o Removed text in Section 5.4, "Packetization Layer Actions", that
      described 4.2 BSD because it is obsolete, and removed reference to
      TP4.

   o Updated text in Section 5.5, "Issues for Other Transport
      Protocols", about NFS, including adding a current reference to NFS
      and removing obsolete text.

   o Added a paragraph to Section 6, "Security Considerations", about
      black-hole connections if PTB messages are not received and
      comparison to PLPMTUD.

   o Updated "Acknowledgements".

   o Editorial Changes.

Authors' Addresses

   Jack McCann
   Digital Equipment Corporation

   Stephen E. Deering
   Retired
   Vancouver, British Columbia
   Canada

   Jeffrey Mogul
   Digital Equipment Corporation

   Robert M. Hinden (editor)
   Check Point Software
   959 Skyway Road
   San Carlos, CA 94070
   United States of America

   Email: bob.hinden@gmail.com