Internet Engineering Task Force
Internet-Draft
Updates: 4821 (if approved)
Intended status: Standards Track
Expires: January 3, 2019

G. Fairhurst
T. Jones
University of Aberdeen
M. Tuexen
I. Ruengeler
Muenster University of Applied Sciences

July 02, 2018

# Packetization Layer Path MTU Discovery for Datagram Transports
# draft-ietf-tsvwg-datagram-plpmtud-03

Abstract

This document describes a robust method for Path MTU Discovery (PMTUD) for datagram Packetization layers. The document describes an extension to RFC 1191 and RFC 8201, which specifies ICMP-based Path MTU Discovery for IPv4 and IPv6. The method allows a Packetization Layer (PL), or a datagram application that uses a PL, to discover whether a network path can support the current size of datagram. This can be used to detect and reduce the message size when a sender encounters a network black hole (where packets are discarded, and no ICMP message is received). The method can also probe a network path with progressively larger packets to find whether the maximum packet size can be increased. This allows a sender to determine an appropriate packet size, providing functionally for datagram transports that is equivalent to the Packetization layer PMTUD specification for TCP, specified in RFC4821.

The document also provides implementation notes for incorporating Datagram PMTUD into IETF Datagram transports or applications that use transports.

When published, this specification updates RFC4821.

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 January 3, 2019.



Table of Contents

## 1. Introduction

The IETF has specified datagram transport using UDP, SCTP, and DCCP, as well as protocols layered on top of these transports (e.g., SCTP/UDP, DCCP/UDP) and directly over the IP network layer. This document describes a robust method for Path MTU Discovery (PMTUD) that may be used with these transport protocols (or the applications that use their transport service) to discover an appropriate size of packet to use across an Internet path.

### 1.1. Classical Path MTU Discovery

Classical Path Maximum Transmission Unit Discovery (PMTUD) can be used with any transport that is able to process ICMP Packet Too Big (PTB) messages (e.g., [RFC1191] and [RFC8201]). The term PTB message

is applied to both IPv4 ICMP Unreachable messages (type 3) that carry the error Fragmentation Needed (Type 3, Code 4) and ICMPv6 packet too big messages (Type 2). When a sender receives a PTB message, it reduces the effective MTU to the value reported as the Link MTU in the PTB message, and a method that from time-to-time increases the packet size in attempt to discover an increase in the supported PMTU. The packets sent with a size larger than the current effective PMTU are known as probe packets.

Packets not intended as probe packets are either fragmented to the current effective PMTU, or the attempt to send fails with an error code. Applications are sometimes provided with a primitive to let them read the maximum packet size, derived from the current effective PMTU.

Classical PMTUD is subject to protocol failures. One failure arises when traffic using a packet size larger than the actual PMTU is black-holed (all datagrams sent with this size, or larger, are silently discarded without the sender receiving ICMP PTB messages). This could arise when the PTB messages are not delivered back to the sender for some reason [RFC2923]). For example, ICMP messages are increasingly filtered by middleboxes (including firewalls) [RFC4890]. A stateful firewall could be configured with a policy to block incoming ICMP messages, which would prevent reception of PTB messages to endpoints behind this firewall. Other examples include cases where PTB messages are not correctly processed/generated by tunnel endpoints.

Another failure could result if a node that is not on the network path sends a PTB message that attempts to force the sender to change the effective PMTU [RFC8201]. A sender can protect itself from reacting to such messages by utilising the quoted packet within a PTB message payload to validate that the received PTB message was generated in response to a packet that had actually originated from the sender. However, there are situations where a sender would be unable to provide this validation.

Examples where validation of the PTB message is not possible include:

- When the router issuing the ICMP message is acting on a tunneled packet, the ICMP message will be directed to the tunnel endpoint. This tunnel endpoint is responsible for forwardiung the ICMP message and also processing the quoted packet within the payload field to remove the effect of the tunnel, and return a correctly fromatted ICMP message to the sender. Failure to do this results in black-holing.

- When a router issuing the ICMP message implements RFC792 [RFC0792], it is only required the to include the first 64 bits of the IP payload of the packet within the quoted payload.This may be insufficient to perfom the tunnel processing described in the previous bullet. Even if the decapsulated message is processed by the tunnel endpoint, there could be insufficient bytes remaining for the sender to interpret the quoted transport information. RFC1812 [RFC1812] requires routers to return the full packet if possible, often the case for IPv4 when used the path includes tunnels; or where the packet has been encapsulated/tunneled over an encrypted transport and it is not possible to determine the original transport header ).

- Even when the PTB message includes sufficient bytes of the quoted packet, the network layer could lack sufficient context to validate the message, because this depends on information about the active transport flows at an endpoint node (e.g., the socket/ address pairs being used, and other protocol header information).

## 1.2. Packetization Layer Path MTU Discovery

The term Packetization Layer (PL) has been introduced to describe the layer that is responsible for placing data blocks into the payload of IP packets and selecting an appropriate Maximum Packet Size (MPS). This function is often performed by a transport protocol, but can also be performed by other encapsulation methods working above the transport.

In contrast to PMTUD, Packetization Layer Path MTU Discovery (PLPMTUD) [RFC4821] does not rely upon reception and validation of PTB messages. It is therefore more robust than Classical PMTUD. This has become the recommended approach for implementing PMTU discovery with TCP.

It uses a general strategy where the PL sends probe packet to search for the largest size of unfragmented datagram that can be sent over a path. The probe packets are sent with a progressively larger packet size. If a probe packet is successfully delivered (as determined by the PL), then the PLPMTU is raised to the size of the successful probe. If no response is received to a probe packet, the method reduces the probe size. This PLPMTU is used to set the application MPS.

PLPMTUD introduces flexibility in the implementation of PMTU discovery. At one extreme, it can be configured to only perform PTB black hole detection and recovery to increase the robustness of Classical PMTUD, or at the other extreme, all PTB processing can be disabled and PLPMTUD can completely replace Classical PMTUD.

PLPMTUD can also include additional consistency checks without increasing the risk of increased black-holing. For instance,the information available at the PL, or higher layers, makes PTB validation more straight forward.

### 1.3. Path MTU Discovery for Datagram Services

Section 4 of this document presents a set of algorithms for datagram protocols to discover the largest size of unfragmented datagram that can be sent over a path. The method described relies on features of the PL Section 3 and apply to transport protocols operating over IPv4 and IPv6. It does not require cooperation from the lower layers, although it can utilise ICMP PTB messages when these received messages are made available to the PL.

The UDP Usage Guidelines [RFC8085] state "an application SHOULD either use the Path MTU information provided by the IP layer or implement Path MTU Discovery (PMTUD)", but does not provide a mechanism for discovering the largest size of unfragmented datagram than can be used on a path. Prior to this document, PLPMTUD had not been specified for UDP.

Section 10.2 of [RFC4821] recommends a PLPMTUD probing method for the Stream Control Transport Protocol (SCTP). SCTP utilises heartbeat messages as probe packets, but RFC4821 does not provide a complete specification. This document provides the details to complete that specification.

The Datagram Congestion Control Protocol (DCCP) [RFC4340] requires implementations to support Classical PMTUD and states that a DCCP sender "MUST maintain the MPS allowed for each active DCCP session". It also defines the current congestion control MPS (CCMPS) supported by a path. This recommends use of PMTUD, and suggests use of control packets (DCCP-Sync) as path probe packets, because they do not risk application data loss. The method defined in this specification could be used with DCCP.

Section 5 specifies the method for a set of transports, and provides information to enables the implementation of PLPMTUD with other datagram transports and applications that use datagram transports.

## 2. 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].

Other terminology is directly copied from [RFC4821], and the definitions in [RFC1122].

Black-Holed: When the sender is unaware that packets are not delivered to the destination endpoint (e.g., when the sender transmits packets of a particular size with a previously known effective PMTU (also refered to as the PLPMTU), but is unaware of a change to the path that resulted in a smaller PLPMTU).

Classical Path MTU Discovery: Classical PMTUD is a process described in [RFC1191] and [RFC8201], in which nodes rely on PTB messages to learn the largest size of unfragmented datagram than can be used across a path.

Datagram: A datagram is a transport-layer protocol data unit, transmitted in the payload of an IP packet.

Effective PMTU: The current estimated value for PMTU that is used by a PMTUD. This is equivalent to the PLPMTU derived by PLPMTUD.

EMTU_S: The Effective MTU for sending (EMTU_S) is defined in [RFC1122] as "the maximum IP datagram size that may be sent, for a particular combination of IP source and destination addresses...".

EMTU_R: The Effective MTU for receiving (EMTU_R) is designated in [RFC1122] as the largest datagram size that can be reassembled by EMTU_R ("Effective MTU to receive").

Link: A communication facility or medium over which nodes can communicate at the link layer, i.e., a layer below the IP layer. Examples are Ethernet LANs and Internet (or higher) layer and tunnels.

Link MTU: The Maximum Transmission Unit (MTU) is the size in bytes of the largest IP packet, including the IP header and payload, that can be transmitted over a link. Note that this could more properly be called the IP MTU, to be consistent with how other standards organizations use the acronym MT. This includes the IP header, but excludes link layer headers and other framing that is not part of IP or the IP payload. Other standards organizations generally define link MTU to include the link layer headers.

MPS: The Maximum Packet Size (MPS) is the largest size of application data block that can be sent unfragmented across a path. In DPLPMTUD this quantity is derived from PLPMTU by taking into consideration the size of the application and lower protocol layer headers.

Packet: An IP header plus the IP payload.

Packetization Layer (PL): The layer of the network stack that places data into packets and performs transport protocol functions.

Path: The set of link and routers traversed by a packet between a source node and a destination node by a particular flow.

Path MTU (PMTU): The minimum of the Link MTU of all the links forming a path between a source node and a destination node.

PLPMTU: The estimate of the actual PMTU provided by the DPLPMTUD algorithm.

PLPMTUD: Packetization Layer Path MTU Discovery, the method described in this document for datagram PLs, which is an extension to Classical PMTU Discovery.

Probe packet: A datagram sent with a purposely chosen size (typically larger than the current PLPMTU) to detect if packets of this size can be successfully sent end-toend across the network path.

## 3. Features Required to Provide Datagram PLPMTUD

TCP PLPMTUD has been defined using standard TCP protocol mechanisms. All of the requirements in [RFC4821] also apply to use of the technique with a datagram PL. Unlike TCP, some datagram PLs require additional mechanisms to implement PLPMTUD.

There are eight requirements for performing the datagram PLPMTUD method described in this specification:

1. PMTU parameters: A DPLPMTUD sender is RECOMMENDED to provide information about the maximum size of packet that can be transmitted by the sender on the local link (the local Link MTU). It MAY utilize similar information about the receiver when this is supplied (note this could be less than EMTU_R). This avoids implementations trying to send probe packets that can not be transmited by the local link. Too high a value may reduce the efficiency of the search algorithm. Some applications also have a maximum transport protocol data unit (PDU) size, in which case there is no benefit from probing for a size larger than this (unless a transport allows multiplexing multiple applications PDUs into the same datagram).

2. PLPMTU: A datagram application MUST be able to choose the size of datagrams sent to the network, up to the PLPMTU, or a smaller

value (such as the MPS) derived from this. This value is managed by the DPLPMTUD method. The PLPMTU (specified as the effective PMTU in Section 1 of [RFC1191]) is equivalent to the EMTU_S (specified in [RFC1122]).

3. Probe packets: On request, a PLPMTUD sender is REQUIRED to be able to transmit a packet larger than the PLMPMTU. This can be uses to send a probe packet. In IPv4, a probe packet MUST be sent with the Don't Fragment (DF) bit set in the IP header, and without network layer endpoint fragmentation. In IPv6, a probe packet is always sent without source fragmentation (as specified in section 5.4 of [RFC8201]).

4. Processing PTB messages: A DPLPMTUD sender MAY optionally utilize PTB messages received from the network layer to help identify when a path does not support the current size of packet probe. Any received PTB message MUST be validated before it is used to update the PLPMTU discovery information [RFC8201]. This validation confirms that the PTB message was sent in response to a packet originating by the sender, and needs to be performed before the PLPMTU discovery method reacts to the PTB message. When the router link MTU is indicated in the PTB message this MAY be used by DPLPMTUD to reduce the probe size but MUST NOT be used to increase the PLPMTU ([RFC8201]). This validation SHOULD utilise information that can not be simply determined by an off-path attacker, for example, by checking the value of a protocol header field known only to the two PL endpoints. (Some datagram applications use well-known source and destination ports and therefore this check needs to rely on other information.)

5. Reception feedback: The destination PL endpoint is REQUIRED to provide a feedback method that indicates to the DPLPMTUD sender when a probe packet has been received by the destination PL endpoint. The local PL endpoint at the sending node is REQUIRED to pass this feedback to the sender-side DPLPMTUD method.

6. Probing and congestion control: The isolated loss of a probe packet SHOULD NOT be treated as an indication of congestion and its loss SHOULD NOT directly trigger a congestion control reaction [RFC4821].

7. Probe loss recovery: If the data block carried by a probe packet needs to be sent reliably, the PL (or layers above) MUST arrange retransmission/repair of any resulting loss. This method MUST be robust in the case where probe packets are lost due to other reasons (including link transmission error, congestion). The DPLPMTUD method treats isolated loss of a probe packet (with or

without an PTB message) as a potential indication of a PMTU limit on the path, but not as an indictaion of congestion Paragraph 6.

8. Shared PLPMTU state: The PLPMTU value could also be stored with the corresponding entry in the destination cache and used by other PL instances. The specification of PLPMTUD [RFC4821] states: "If PLPMTUD updates the MTU for a particular path, all Packetization Layer sessions that share the path representation (as described in Section 5.2 of [RFC4821]) SHOULD be notified to make use of the new MTU and make the required congestion control adjustments". Such methods need to be robust to the wide variety of underlying network forwarding behaviours, PLPMTU adjustments based on shared PLPMTU values should be incorporated in the search algorithms. Section 5.2 of [RFC8201] provides guidance on the caching of PMTU information and also the relation to IPv6 flow labels.

In addition, the following principles are stated for design of a DPLPMTUD method:

- MPS: A method MUST signal appropriate MPS to the higher layer using the PL. This may change following a change to the path. The method SHOULD avoid forcing an application to use an arbitrary small MPS (PLPMTU) for transmission while the method is searching for the currently supported PLPMTU. Datagram PLs do not necessarily support fragmentation of PDUs larger than the PLPMTU. A reduced MPS can adversely impact the performance of a datagram application.

- Path validation: A method MUST be robust to path changes that could have occurred since the path characteristics were last confirmed, and to the possibility of inconsistent path information being received.

- Datagram reordering: A method MUST be robust to the possibility that a flow encounters reordering, or has the traffic (including probe packets) is divided over more than one network path.

- When to probe: A method SHOULD determine whether the path capacity has increased since it last measured the path. This determines when the path should again be probed.

## 3.1. PLPMTU Probe Packets

The DPLPMTUD method relies upon the PL sender being able to generate probe packets with a specific size. TCP is able to generate these probe packets by choosing to appropriately segment data being sent [RFC4821].

In contrast, a datagram PL that needs to construct a probe packet has to either request an application to send a data block that is larger than that generated by an application, or to utilise padding functions to extend a datagram beyond the size of the application data block. Protocols that permit exchange of control messages (without an application data block) could alternatively prefer to generate a probe packet by extending a control message with padding data.

When the method fails to validate the PLPMTU, it may be required to send a probe packet with a size less than the size of the data block generated by an application. In this case, the PL could provide a way to fragment a datagram at the PL, or could instead utilise a control packet with padding.

A receiver needs to be able to distinguish an in-band data block from any added padding. This is needed to ensure that any added padding is not passed on to an application at the receiver.

This results in three possible ways that a sender can create a probe packet listed in order of preference:

Probing using padding data: A probe packet that contains only control information together with any padding needed to inflate the packet to the size required for the probe packet. Since these probe packets do not carry an application-supplied data block,they do not typically require retransmission, although they do still consume network capacity and incur endpoint processing.

Probing using appication data and padding data: A probe packet that contains a data block supplied by an application that is combined with padding to inflate the length of the datagram to the size required for the probe packet. If the application/transport needs protection from the loss of this probe packet, the application/ transport may perform transport-layer retransmission/repair of the data block (e.g., by retransmission after loss is detected or by duplicating the data block in a datagram without the padding data).

Probing using appication data: A probe packet that contains a data block supplied by an application that matches the size required for the probe packet. This method requests the application to issue a data block of the desired probe size. If the application/ transport needs protection from the loss of an unsuccessful probe packet, the application/transport needs then to perform transport-layer retransmission/repair of the data block (e.g., by retransmission after loss is detected).

A PL that uses a probe packet carrying an application data block, could need to retransmit this application data block if the probe fails. This could need the PL to re-fragment the data block to a smaller packet size that is expected to traverse the end-to-end path (which could utilise network-layer or PL fragmentation when these are available).

DLPMTUD MAY choose to use only one of these methods to simplify the implementation.

## 3.2. Validation of Probe Packet Size

The PL needs a method to determine when probe packets have been successfully received end-to-end across a network path.

Transport protocols can include end-to-end methods that detect and report reception of specific datagrams that they send (e.g., DCCP and SCTP provide keep-alive/heartbeat features). When supported, this mechanism SHOULD also be used by DPLPMTUD to acknowledge reception of a probe packet.

A PL that does not acknowledge data reception (e.g., UDP and UDP-Lite) is unable to detect when the packets that it sends are discarded because their size is greater than the actual PMTU. These PLs need to either rely on an application protocol to detect this loss, or make use of an additional transport method such as UDP-Options [I-D.ietf-tsvwg-udp-options]. In addition, they might need to send reachability probes (e.g., periodically solicit a response from the destination) to determine whether the last successfully probed PLPMTU is still supported by the network path.

Section Section 4 specifies this function for a set of IETF-specified protocols.

## 3.3. Reducing the PLPMTU: Confirming Path Characteristics

If the DPLPMTUD method detects that a packet with the PLPMTU size is no supported by the network path, then the DLPMTUD method needs to validate the PLPMTU. This can happen when a validated PTB message is received, or another event that indicates the network path no longer sustains this packet size, such as a loss report from the PL

All implementations of DPLPMTUD are REQUIRED to provide support that reduces the PLPMTU when the actual PMTU supported by a network path is less than the PLPMTU.

### 3.4. Increasing the PLPMTU: Supporting Path Changes

An implementation that only reduces the PLPMTU to a suitable size is sufficient to ensure reliable operation, but may be very inefficient when the actual PMTU changes or when the method (for whatever reason) makes a suboptimal choice for the PLPMTU.

A full implementation of the DPLPMTUD method is RECOMMENDED to provide a way for the sending PL endpoint to detect when the PLPMTU is smaller than the actual PMTU size. This allows the sender to increase the PLPMTU following a change in the characteristics of the path, such as when a link is reconfigured with a larger MTU, or when there is a change in the set of links traversed by an end-to-end flow (e.g. after a routing or fail-over decision).

### 3.5. Robustness to inconsistent Path information

The decision to increase the PLPMTU needs to be robust to the possibility that information learned about the path is inconsistent (this could happen when probe packets are lost due to other reasons, or some of the packets in a flow are forwarded along a portion of the path that supports a different PMTU).

Frequent path changes could occur due to unexpected "flapping" - where some packets from a flow pass along one path, but other packets follow a different path with different properties. DPLPMTUD can be made robust to these anomalies by introducing hysteresis into the decision to increase the Maximum Packet Size.

XXX A future revision of this section will include recommend appropriate methods to provide robustness. XXX

## 4. Datagram Packetization Layer PMTUD

This section specifies Datagram PLPMTUD (DPLPMTUD). This method can be introduced at various points in the IP protocol stack, to discover the PLPMTU so that the application can use an MPS appropriate to the current network path.

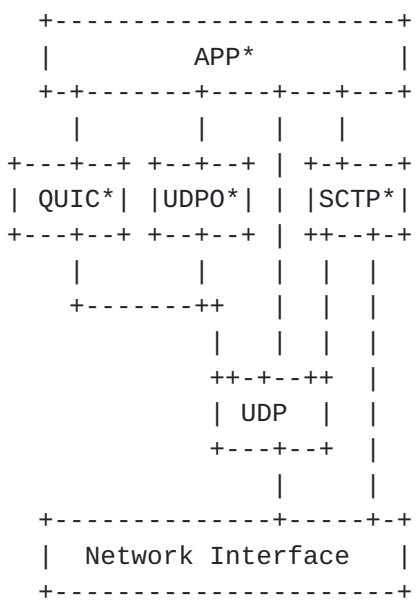



Figure 1: Examples where DPLPMTUD can be implemented

The central idea of DPLPMTUD is probing by a sender. Probe packets are sent to find out the maximum size of user message that is completely transferred across the network path from the sender to the destination.

The are various functions performed by the algorithm:

## 4.1. PROBE_SEARCH: Probing for a larger PLPMTU

The DPLPMTUD method utilises probe packets to confirm that a packet of size PROBED_SIZE can traverse the network path. The PROBE_COUNT is initialised to zero when a probe packet is first sent with a particular size.

A timer is used to trigger the generation of probe packets. The probe_timer is started each time a probe packet is sent to the destination and is cancelled when receipt of the probe packet is acknowledged. The PROBED_SIZE is confirmed, and this value is then assignmed to PLPMTU. The DPLPMTUD method may send subsequent probes of an increasing size. Increasing probes follow a search strategy as discussed in Section 4.7.

Each time the probe_timer expires, the PROBE_COUNT is incremented, the probe_timer is reinitialised, and a probe packet of the same size is retransmitted.

The maximum number of retransmissions for a PROBED_SIZE is configured (MAX_PROBES). If the value of the PROBE_COUNT reaches MAX_PROBES, probing will stop and enters the PROBE_DONE state.

### 4.2. The PROBE_DONE state

When the PL sender completes probing for a larger PLPMTU, it enters the PROBE_DONE state. This starts the PMTU_RAISE_TIMER. While this running, the PLPMTU remains at the value set in the last succesful probe packet.

If the PL is designed in a way that is unable to validate reachability to the destination endpoint after probing has completed, the method uses a REACHABILITY_TIMER to periodically repeat a probe packet for the current PLPMTU size, while the PMTU_RAISE_TIMER is running. If the REACHABILITY_TIMER expires, the method exits the PROBE_DONE state. The done state is also exited when a validated PTB message is received.

If the PMTU_RAISE_TIMER expires, the PL sender also exits the PROBE_DONE state, but in this case resumes probing from the size of the PLPMTU.

### 4.3. Validation and Use of PTB Messages

This section describes processing for both IPv4 ICMP Unreachable messages (type 3) and ICMPv6 packet too big messages.

A PL that receives a PTB message from a router or middlebox, MUST validate the PTB message. The PL checks the protocol information in the quoted payload to validate the message originated from the sending node. The node also checks that the reported link MTU size is less than the size used by packet probes. PTB messages are discarded if they fail to pass these checks, or where there is insufficient ICMP payload to perform these checks. The checks are intended to provide protection from packets that originate from a node that is not on the network path or a node that attempts to report a larger link MTU than the current probe size.

PTB messages that have been validated can be utilised by the DPLPMTUD algorithm. A method that utilises these PTB messages can improve the speed at the which the algorithm detects an appropriate PLPMTU compared to one that relies solely on probing.

### 4.4. Timers

The method in the previous subsections utilises three timers:

PROBE_TIMER: Configured to expire after a period longer than the maximum time to receive an acknowledgment to a probe packet. This value MUST be larger than 1 second, and SHOULD be larger than 15 seconds. Guidance on selection of the timer value are provide in section 3.1.1 of the UDP Usage Guidelines [RFC8085].

If the PL has an RTT estimate and timely acknowedgements the PROBE_TIMER can be derrived from the PL RTT estimate.

PMTU_RAISE_TIMER: Configured to the period a sender ought to continue use the current PLPMTU, after which it re-commences probing for a higher PMTU. This timer has a period of 600 secs, as recommended by DPLPMTUD [RFC4821].

REACHABILITY_TIMER: Configured to the period a sender ought to wait before confirming the current PLPMTU is still supported. This is less than the PMTU_RAISE_TIMER and used to decrease the PLPMTU (e.g. when a black hole is encountered).

DPLPMTUD ought to suspend reachability probes when no application data has been sent since the previous probe packet. Guidance on selection of the timer value are provide in section 3.1.1 of the UDP Usage Guidelines[RFC8085]. DPLPMTUD ought to be suspended or only sent in conjuction with out traffic during periods of dormancy. This PLPMTU validation needs to be frequent enough when data is flowing that the sending PL does not black hole extensive amounts of traffic

An implementation could implement the various timers using a single timer process.

### 4.5. Constants

The following constants are defined:

MAX_PROBES: The maximum value of the PROBE_ERROR_COUNTER. The default value of MAX_PROBES is 10.

MIN_PMTU: The smallest allowed probe packet size. For IPv6, this value is 1280 bytes, as specified in [RFC2460]. For IPv4, the minimum value is 68 bytes. (An IPv4 routed is required to be able to forward a datagram of 68 octets without further fragmentation. This is the combined size of an IPv4 header and the minimum fragment size of 8 octets.)

BASE_PMTU: The BASE_PMTU is a considered a size that ought to work in most cases. The size is equal to or larger than the minimum permitted and smaller than the maximum allowed. In the case of IPv6, this value is 1280 bytes [RFC2460]. When using IPv4, a size of 1200 bytes is RECOMMENDED.

MAX_PMTU: The MAX_PMTU is the largest size of PLPMTU that is probed. This has to be less than or equal to the minimum of the local MTU of the outgoing interface and the destination PLMTU for receiving. An application or PL may reduce this when it knows there is no need to send packets above a specific size.

The figure below illustrates the relationship between some of these variables, in this case when the DPLPMTUD algorithm performs path probing to increase the size of the PLPMTU. The MPS is less than the PLPMTU. A probe packet has been sent of size PROBED_SIZE. When this is acknowledged, the PLPMTU will be raised to PROBED_SIZE allowing the PROBED_SIZE to be increased towards the actual PMTU.

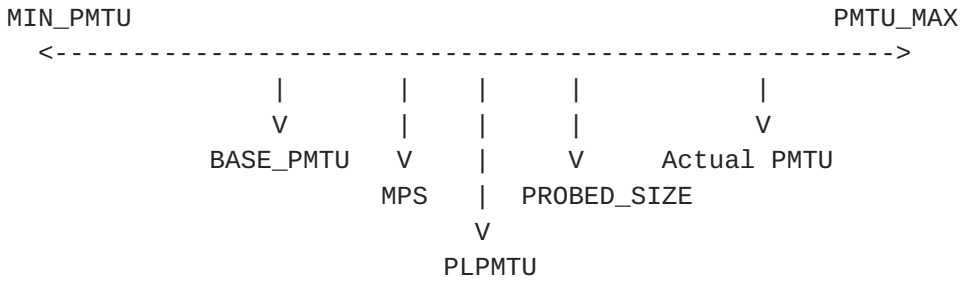



Figure 2: Relationships between probe and packet sizes

## 4.6. Variables

This method utilises a set of variables:

PROBE_TIMER: Configured to expire after a period longer than the maximum time to receive an acknowledgment to a probe packet. This value MUST be larger than 1 second, and SHOULD be larger than 15 seconds. Guidance on selection of the timer value are provide in section 3.1.1 of the UDP Usage Guidelines [RFC8085].

PL with RTT estimates may use values smaller than 1 seconded derrived from their RTT estimate to speed up detection of connectivity issues on the path.

PROBED_SIZE: The PROBED_SIZE is the size of the current probe packet. This is a tentative value for the PLPMTU, which is awaiting confirmation by an acknowledgment.

PROBE_COUNT: This is a count of the number of unsuccessful probe packets that have been sent with size PROBED_SIZE. The value is initialised to zero when a particular size of PROBED_SIZE is first attempted.

PTB_SIZE: The PTB_Size is value returned by a validated PTB message indicating the local MTU size of a router along the path.

## 4.7. Selecting PROBED_SIZE

Implementations discover the search range by validating the minimum path MTU and then using the probe method to select a PROBED_SIZE less than or equal to the maximum PMTU_MAX. Where PMTU_MAX is the minimum of the local link MTU and EMTU_R (learned from the remote endpoint). The PMTU_MAX MAY be constrained by an application that has a maximum to the size of datagrams it wishes to send.

Implementations use a search algorithm to choose probe sizes within the search range.

xxx A future version of this section will detail example methods for selecting probe size values, but does not plan to mandate a single method. xxx

Implementations MAY optimizse the search procedure by selecting step sizes from a table of common PMTU sizes.

Implementations SHOULD select probe sizes to maximise the gain in PLPMTU each search step. Implementations ought to take into consideration useful probe size steps and a minimum useful gain in PLPMTU.

## 4.8. Simple Black Hole Detection

The DPLPMTUD method can be used to provide black hole detection. This enables a reduction of the PLPMTU when a PL sender encounters a path that fails to support the current MPS and also fails to return a PTB message to the sender.

The simple method starts by setting the PLPMTU to the BASE_PMTU. When the method detects that communication is not possible with this size of packet, the PLPMTU is reduced, until an operable message size is reached or the PLPMTU reaches the BASE_MTU size. The method enables a sending PL to inform an application of the reduced MPS and accordingly send smaller packets.

The simple black hole detetction method does not seek to increase the PLPMTU. This makes it vulneable to transient reductions in the

actual PLPMTU, which could result in a PLPMTU lower than the actual PMTU.

The full methiod is specified in Section 4.9.

### 4.8.1. Simple Black Hole Detection State Machine

The PL sender starts with the PLPMTU and PROBED_SIZE set to the BASE_PMTU.

While a PL has a PLPMTU greater than the BASE_MTU, the PL needs to send probe packets at the PROBED_SIZE to revalidate the PLPMTU. Black hole detection is also triggered by lack of reachability at the PL. When the PL sender detects that multiple transmissions of packets of PROBED_SIZE are no longer being acknowledged (e.g., When the number of probe packets sent without receiving an acknowledgement (PROBE_COUNT) becomes greater than the MAX_PROBES), the PL concludes that it has detected a black hole and reduces PLPMTU.

The connectivity check may be performened by the protocol implementing the PL (as in PLPMTUD for TCP [RFC4821]). When the application using the PL does not regularly send packets of size PROBED_SIZE, additional probe packets need to be sent by PL using the reachability timer Section 4.4.

If method does reduces the PLPMTU to the MIN_PMTU, the method concludes the path does not support the MIN_PMTU.

If multihoming is supported, a state machine is needed for each active path.

The state machine for a simple black hole detection mechanism is depicted in Figure 3.

XXX a future version of the simple black hole detection state machine might consider icmp PTB messages XXX

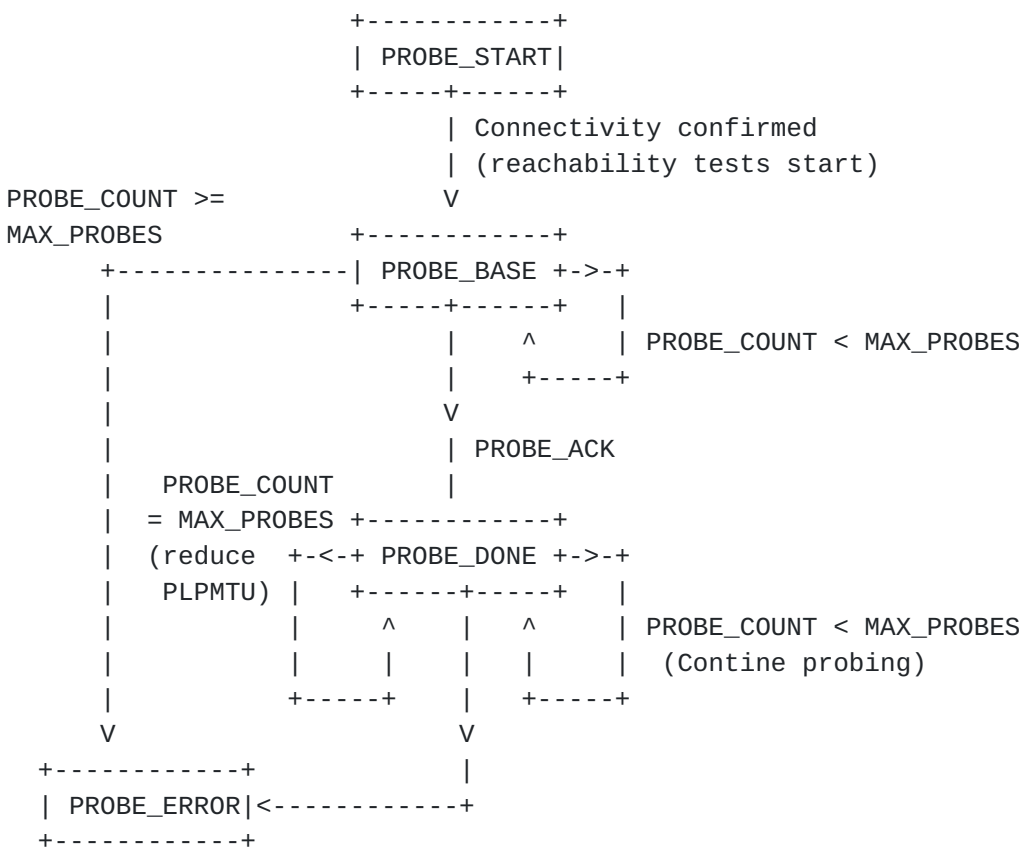



Figure 3: State machine for detecting black holes

## 4.9. Full State Machine

A full state machine for DPLPMTUD is depicted in Figure 4. If multihoming is supported, a state machine is needed for each active path.

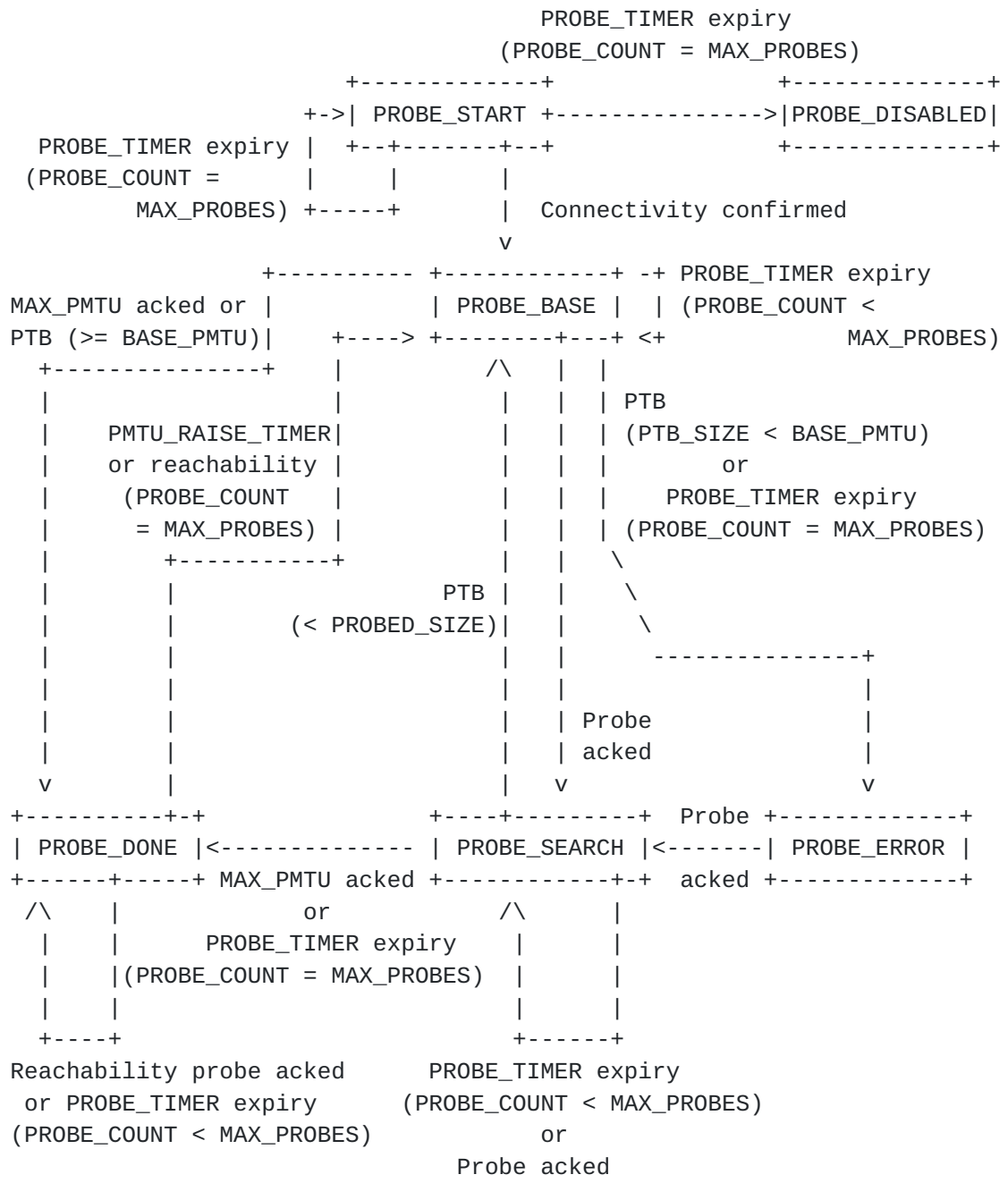



Figure 4: State machine for Datagram PLPMTUD

XXX A future version of this document will update the state machine to describe handling of validated PTB messages. XXX

The following states are defined to reflect the probing process:

PROBE_START: The PROBE_START state is the initial state before probing has started. PLPMTUD is not performed in this state. The

state transitions to PROBE_BASE, when a path has been confirmed, i.e. when a sent packet has been acknowledged on this path. Any transport method may be used to exit PROBE_BASE as long as the send packet is acknowledge by the other side. The PLPMTU is set to the BASE_PMTU size. Probing ought to start immediately after connection setup to prevent the prevent the loss of user data.

PROBE_BASE: The PROBE_BASE state is the starting point for probing with datagram PLPMTUD. It is used to confirm whether the BASE_PMTU size is supported by the network path. On entry, the PROBED_SIZE is set to the BASE_PMTU size and the PROBE_COUNT is set to zero. A probe packet is sent, and the PROBE_TIMER is started. The state is left when the PROBE_COUNT reaches MAX_PROBES; a PTB message is validated, or a probe packet is acknowledged.

PROBE_SEARCH: The PROBE_SEARCH state is the main probing state. This state is entered either when probing for the BASE_PMTU was successful or when there is a successful reachability test in the PROBE_ERROR state. On entry, the PLPMTU is set to the last acknowledged PROBED_SIZE.

The PROBE_COUNT is set to zero when the first probe packet is sent for each probe size. Each time a probe packet is acknowledged, the PLPMTU is set to the PROBED_SIZE, and then the PROBED_SIZE is increased.

When a probe packet is sent and not acknowledged within the period of the PROBE_TIMER, the PROBE_COUNT is incremented and the probe packet is retransmitted. The state is exited when the PROBE_COUNT reaches MAX_PROBES; a PTB message is validated; or a probe of size PMTU_MAX is acknowledged.

PROBE_ERROR: The PROBE_ERROR state represents the case where the network path is not known to support an PLPMTU of at least the BASE_PMTU size. It is entered when either a probe of size BASE_PMTU has not been acknowledged or a validated PTB message indicates a smaller link MTU than the BASE_PMTU. On entry, the PROBE_COUNT is set to zero and the PROBED_SIZE is set to the MIN_PMTU size, and the PLPMTU is reset to MIN_PMTU size. In this state, a probe packet is sent, and the PROBE_TIMER is started. The state transitions to the PROBE_SEARCH state when a probe packet is acknowledged.

PROBE_DONE: The PROBE_DONE state indicates a successful end to a probing phase. DPLPMTUD remains in this state until either the PMTU_RAISE_TIMER expires or a received PTB message is validated.

When PLPMTUD uses an unacknowledged PL and is in the PROBE_DONE state, a REACHABILITY_TIMER periodically resets the PROBE_COUNT and schedules a probe packet with the size of the PLPMTU. If the probe packet fails to be acknowledged after MAX_PROBES attempts, the method enters the PROBE_BASE state. When used with an acknowledged PL (e.g., SCTP), DPLPMTUD SHOULD NOT continue to probe in this state.

PROBE_DISABLED: The PROBE_DISABLED state indicates that connectivity could not be established. DPLPMTUD MUST NOT probe in this state.

Appendix A contains an informative description of key events.

## 5. Specification of Protocol-Specific Methods

This section specifies protocol-specific details for datagram PLPMTUD for IETF-specified transports.

The first subsection provides guidance on how to implement the DPLPMTUD method as a part of an application using UDP or UDP-Lite. The guidance also applies to other datagram services that do not include a specific transport protocol (such as a tunnel encapsulation). The following subsection describe how DPLPMTUD can be implemented as a part of the transport service, allowing applications using the service to benefit from discovery of the PLPMTU without themselves needing to implement this method.

### 5.1. Application support for DPLPMTUD with UDP or UDP-Lite

The current specifications of UDP [RFC0768] and UDP-Lite [RFC3828] do not define a method in the RFC-series that supports PLPMTUD. In particular, the UDP transport does not provide the transport layer features needed to implement datagram PLPMTUD.

The DPLPMTUD method can be implemented as a part of an application built directly or indirectly on UDP or UDP-Lite, but relies on higher-layer protocol features to implement the method [RFC8085].

Some primitives used by DPLPMTUD might not be available via the Datagram API (e.g., the ability to access the PLPMTU cache, or interpret received ICMP PTB messages).

In addition, it is desirable that PMTU discovery is not performed by multiple protocol layers. An application SHOULD avoid implementing DPLPMTUD when the underlying transport system provides this capability. Using a common method for manging the PLPMTU has benefits, both in the ability to share state between different processes and opportunities to coordinate probing.

#### 5.1.1. Application Request

An application needs an application-layer protocol mechanism (such as a message acknowledgement method) that solicits a response from a destination endpoint. The method SHOULD allow the sender to check the value returned in the response to provide additional protection from off-path insertion of data [RFC8085], suitable methods include a parameter known only to the two endpoints, such as a session ID or initialised sequence number.

#### 5.1.2. Application Response

An application needs an application-layer protocol mechanism to communicate the response from the destination endpoint. This response may indicate successful reception of the probe across the path, but could also indicate that some (or all packets) have failed to reach the destination.

#### 5.1.3. Sending Application Probe Packets

A probe packet that may carry an application data block, but the successful transmission of this data is at risk when used for probing. Some applications may prefer to use a probe packet that does not carry an application data block to avoid disruption to normal data transfer.

#### 5.1.4. Validating the Path

An application that does not have other higher-layer information confirming correct delivery of datagrams SHOULD implement the REACHABILITY_TIMER to periodically send probe packets while in the PROBE_DONE state.

#### 5.1.5. Handling of PTB Messages

An application that is able and wishes to receive PTB messages MUST perform ICMP validation as specified in Section 5.2 of [RFC8085]. This requires that the application to check each received PTB messages to validate it is received in response to transmitted traffic and that the reported link MTU is less than the current probe size. A validated PTB message MAY be used as input to the DPLPMTUD algorithm, but MUST NOT be used directly to set the PLPMTU.

### 5.2. DPLPMTUD with UDP Options

UDP-Options [I-D.ietf-tsvwg-udp-options] can supply the additional functionality required to implement DPLPMTUD within the UDP transport

service. This avoids the need for applications to implement the DPLPMTUD method.

This enables padding to be added to UDP datagrams and can be used to provide feedback acknowledgement of received probe packets.

The specification also defines two UDP Options to support DPLMTUD.

Section 5.6 of [I-D.ietf-tsvwg-udp-options] defines the MSS option which allows the local sender to indicate the EMTU_R to the peer. This option can be used to initialise PMTU_MAX. An application wishing to avoid the effects of MSS-Clamping (where a middlebox changes the advertised TCP maximum sending size) ought to use a cryptographic method to encrypt this parameter.

### 5.2.1. UDP Request Option

The Request Option allows a sending endpoint to solicit a response from a destination endpoint.

The Request Option carries a four byte token set by the sender. This token can be set to a value that is likely to be known only to the sender (and becomes known to nodes along the end-to-end path). The sender can then check the value returned in the response to provide additional protection from off-path insertion of data [RFC8085].

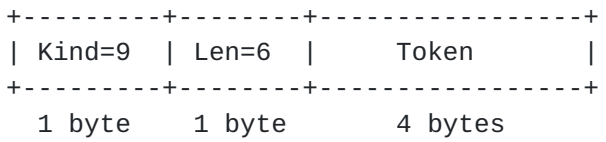



Figure 5: UDP REQ Option Format

### 5.2.2. UDP Response Option

The Response Option is generated by the PL in response to reception of a previously received Echo Request. The Token field associates the response with the Token value carried in the most recently-received Echo Request. The rate of generation of UDP packets carrying a Response Option MAY be rate-limited.

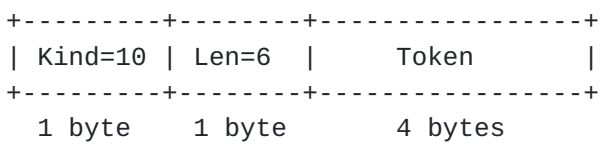



Figure 6: UDP RES Option Format

### 5.3. DPLPMTUD for SCTP

Section 10.2 of [RFC4821] specifies a recommended PLPMTUD probing method for SCTP. It recommends the use of the PAD chunk, defined in [RFC4820] to be attached to a minimum length HEARTBEAT chunk to build a probe packet. This enables probing without affecting the transfer of user messages and without interfering with congestion control. This is preferred to using DATA chunks (with padding as required) as path probes.

XXX Future versions of this document might define a parameter contained in the INIT and INIT ACK chunk to indicate the remote peer MTU to the local peer. However, multihoming makes this a bit complex, so it might not be worth doing. XXX

#### 5.3.1. SCTP/IP4 and SCTP/IPv6

The base protocol is specified in [RFC4960]. This provides an acknowledged PL. A sender can therefore enter the PROBE_BASE state as soon as connectivity has been confirmed.

##### 5.3.1.1. Sending SCTP Probe Packets

Probe packets consist of an SCTP common header followed by a HEARTBEAT chunk and a PAD chunk. The PAD chunk is used to control the length of the probe packet. The HEARTBEAT chunk is used to trigger the sending of a HEARTBEAT ACK chunk. The reception of the HEARTBEAT ACK chunk acknowledges reception of a successful probe.

The HEARTBEAT chunk carries a Heartbeat Information parameter which should include, besides the information suggested in [RFC4960], the probe size, which is the size of the complete datagram. The size of the PAD chunk is therefore computed by reducing the probing size by the IPv4 or IPv6 header size, the SCTP common header, the HEARTBEAT request and the PAD chunk header. The payload of the PAD chunk contains arbitrary data.

To avoid fragmentation of retransmitted data, probing starts right after the handshake, before data is sent. Assuming normal behaviour (i.e., the PMTU is smaller than or equal to the interface MTU), this process will take a few round trip time periods depending on the number of PMTU sizes probed. The Heartbeat timer can be used to implement the PROBE_TIMER.

#### 5.3.1.2. Validating the Path with SCTP

Since SCTP provides an acknowledged PL, a sender does MUST NOT implement the REACHABILITY_TIMER while in the PROBE_DONE state.

#### 5.3.1.3. PTB Message Handling by SCTP

Normal ICMP validation MUST be performed as specified in Appendix C of [RFC4960]. This requires that the first 8 bytes of the SCTP common header are quoted in the payload of the PTB message, which can be the case for ICMPv4 and is normally the case for ICMPv6.

When a PTB message has been validated, the router Link MTU indicated in the PTB message SHOULD be used with the DPLPMTUD algorithm, providing that the reported Link MTU is less than the current probe size.

### 5.3.2. DPLPMTUD for SCTP/UDP

The UDP encapsulation of SCTP is specified in [RFC6951].

#### 5.3.2.1. Sending SCTP/UDP Probe Packets

Packet probing can be performed as specified in Section 5.3.1.1. The maximum payload is reduced by 8 bytes, which has to be considered when filling the PAD chunk.

#### 5.3.2.2. Validating the Path with SCTP/UDP

Since SCTP provides an acknowledged PL, a sender does MUST NOT implement the REACHABILITY_TIMER while in the PROBE_DONE state.

#### 5.3.2.3. Handling of PTB Messages by SCTP/UDP

Normal ICMP validation MUST be performed for PTB messages as specified in Appendix C of [RFC4960]. This requires that the first 8 bytes of the SCTP common header are contained in the PTB message, which can be the case for ICMPv4 (but note the UDP header also consumes a part of the quoted packet header) and is normally the case for ICMPv6. When the validation is completed, the router Link MTU size indicated in the PTB message SHOULD be used with the DPLPMTUD providing that the reported link MTU is less than the current probe size.

#### 5.3.3. DPLPMTUD for SCTP/DTLS

The Datagram Transport Layer Security (DTLS) encapsulation of SCTP is specified in [RFC8261]. It is used for data channels in WebRTC implementations.

##### 5.3.3.1. Sending SCTP/DTLS Probe Packets

Packet probing can be done as specified in Section 5.3.1.1.

##### 5.3.3.2. Validating the Path with SCTP/DTLS

Since SCTP provides an acknowledged PL, a sender does MUST NOT implement the REACHABILITY_TIMER while in the PROBE_DONE state.

##### 5.3.3.3. Handling of PTB Messages by SCTP/DTLS

It is not possible to perform normal ICMP validation as specified in [RFC4960], since even if the ICMP message payload contains sufficient information, the reflected SCTP common header would be encrypted. Therefore it is not possible to process PTB messages at the PL.

### 5.4. DPLPMTUD for QUIC

Quick UDP Internet Connection (QUIC) [I-D.ietf-quic-transport] is a UDP-based transport that provides reception feedback.

Section 9.2 of [I-D.ietf-quic-transport] describes the path considerations when sending QUIC packets. It recommends the use of PADDING frames to build the probe packet. This enables probing the without affecting the transfer of other QUIC frames.

This provides an acknowledged PL. A sender can therefore enter the PROBE_BASE state as soon as connectivity has been confirmed.

#### 5.4.1. Sending QUIC Probe Packets

A probe packet consists of a QUIC Header and a payload containing only PADDING Frames. PADDING Frames are a single octet (0x00) and several of these can be used to create a probe packet of size PROBED_SIZE. QUIC provides an acknowledged PL. A sender can therefore enter the PROBE_BASE state as soon as connectivity has been confirmed.

The current specification of QUIC sets the following:

- BASE_PMTU: 1200. A QUIC sender needs to pad initial packets to 1200 bytes to validate the path can support packets of a useful size.

- MIN_PMTU: 1200 bytes. A QUIC sender that determines the PMTU has fallen below 1200 bytes MUST immediately stop sending on the affected path.

### 5.4.2. Validating the Path with QUIC

QUIC provides an acknowledged PL. A sender therefore MUST NOT implement the REACHABILITY_TIMER while in the PROBE_DONE state.

### 5.4.3. Handling of PTB Messages by QUIC

QUIC operates over the UDP transport, and the guidelines on ICMP validation as specified in Section 5.2 of [RFC8085] therefore apply. Although QUIC does not currently specify a method for validating ICMP responses, it does provide some guidelines to make it harder for an off-path attacker to inject ICMP messages.

- Set the IPv4 Don't Fragment (DF) bit on a small proportion of packets, so that most invalid ICMP messages arrive when there are no DF packets outstanding, and can therefore be identified as spurious.

- Store additional information from the IP or UDP headers from DF packets (for example, the IP ID or UDP checksum) to further authenticate incoming Datagram Too Big messages.

- Any reduction in PMTU due to a report contained in an ICMP packet is provisional until QUIC's loss detection algorithm determines that the packet is actually lost.

XXX The above list was pulled whole from quic-transport - input is invited from QUIC contributors. XXX

## 6. Acknowledgements

This work was partially funded by the European Union's Horizon 2020 research and innovation programme under grant agreement No. 644334 (NEAT). The views expressed are solely those of the author(s).

## 7. IANA Considerations

This memo includes no request to IANA.

XXX If new UDP Options are specified in this document, a request to IANA will be included here. XXX

If there are no requirements for IANA, the section will be removed during conversion into an RFC by the RFC Editor.

## 8. Security Considerations

The security considerations for the use of UDP and SCTP are provided in the references RFCs. Security guidance for applications using UDP is provided in the UDP Usage Guidelines [RFC8085].

There are cases where PTB messages are not delivered due to policy, configuration or equipment design (see Section 1.1), this method therefore does not rely upon PTB messages being received, but is able to utilise these when they are received by the sender. PTB messages could potentially be used to cause a node to inappropriately reduce the PLPMTU. A node supporting DPLPMTUD MUST therefore appropriately validate the payload of PTB messages to ensure these are received in response to transmitted traffic (i.e., a reported error condition that corresponds to a datagram actually sent by the path layer.

Parallel forwarding paths may need to be considered. Section 3.5 identifies the need for robustness in the method when the path information may be inconsistent.

A node performing DPLPMTUD could experience conflicting information about the size of supported probe packets. This could occur when there are multiple paths are concurrently in use and these exhibit a different PMTU. If not considered, this could result in data being black holed when the PLPMTU is larger than the smallest PMTU across the current paths.

An on-path attacker could forge PTB messages to drive down the PLPMTU

## 9. References

### 9.1. Normative References

[I-D.ietf-quic-transport]
Iyengar, J. and M. Thomson, "QUIC: A UDP-Based Multiplexed and Secure Transport", draft-ietf-quic-transport-13 (work in progress), June 2018.

[I-D.ietf-tsvwg-udp-options]
Touch, J., "Transport Options for UDP", draft-ietf-tsvwg-udp-options-04 (work in progress), July 2018.

[RFC0768] Postel, J., "User Datagram Protocol", STD 6, RFC 768, DOI 10.17487/RFC0768, August 1980, <https://www.rfc-editor.org/info/rfc768>.

[RFC0792] Postel, J., "Internet Control Message Protocol", STD 5, RFC 792, DOI 10.17487/RFC0792, September 1981, <https://www.rfc-editor.org/info/rfc792>.

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

[RFC1812] Baker, F., Ed., "Requirements for IP Version 4 Routers", RFC 1812, DOI 10.17487/RFC1812, June 1995, <https://www.rfc-editor.org/info/rfc1812>.

[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>.

[RFC2460] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", RFC 2460, DOI 10.17487/RFC2460, December 1998, <https://www.rfc-editor.org/info/rfc2460>.

[RFC3828] Larzon, L-A., Degermark, M., Pink, S., Jonsson, L-E., Ed., and G. Fairhurst, Ed., "The Lightweight User Datagram Protocol (UDP-Lite)", RFC 3828, DOI 10.17487/RFC3828, July 2004, <https://www.rfc-editor.org/info/rfc3828>.

[RFC4820] Tuexen, M., Stewart, R., and P. Lei, "Padding Chunk and Parameter for the Stream Control Transmission Protocol (SCTP)", RFC 4820, DOI 10.17487/RFC4820, March 2007, <https://www.rfc-editor.org/info/rfc4820>.

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

[RFC6951] Tuexen, M. and R. Stewart, "UDP Encapsulation of Stream Control Transmission Protocol (SCTP) Packets for End-Host to End-Host Communication", RFC 6951, DOI 10.17487/RFC6951, May 2013, <https://www.rfc-editor.org/info/rfc6951>.

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

[RFC8201] McCann, J., Deering, S., Mogul, J., and R. Hinden, Ed., "Path MTU Discovery for IP version 6", STD 87, RFC 8201, DOI 10.17487/RFC8201, July 2017, <https://www.rfc-editor.org/info/rfc8201>.

[RFC8261] Tuexen, M., Stewart, R., Jesup, R., and S. Loreto, "Datagram Transport Layer Security (DTLS) Encapsulation of SCTP Packets", RFC 8261, DOI 10.17487/RFC8261, November 2017, <https://www.rfc-editor.org/info/rfc8261>.

### 9.2. Informative References

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

[RFC2923] Lahey, K., "TCP Problems with Path MTU Discovery", RFC 2923, DOI 10.17487/RFC2923, September 2000, <https://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, <https://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, <https://www.rfc-editor.org/info/rfc4821>.

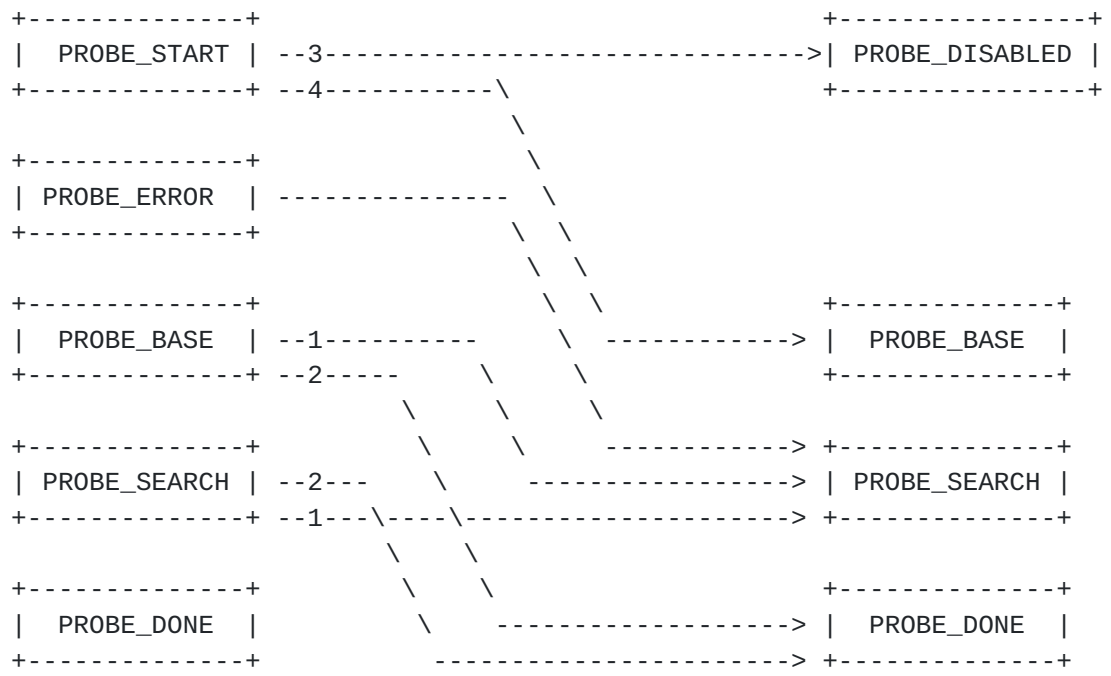
[RFC4890] Davies, E. and J. Mohacsi, "Recommendations for Filtering ICMPv6 Messages in Firewalls", RFC 4890, DOI 10.17487/RFC4890, May 2007, <https://www.rfc-editor.org/info/rfc4890>.

## Appendix A. Event-driven state changes

This appendix contains an informative description of key events:

Path Setup: When a new path is initiated, the state is set to PROBE_START. As soon as the path is confirmed, the state changes to PROBE_BASE and probing for this path is started. The first probe packet is sent with the size of the BASE_PMTU.

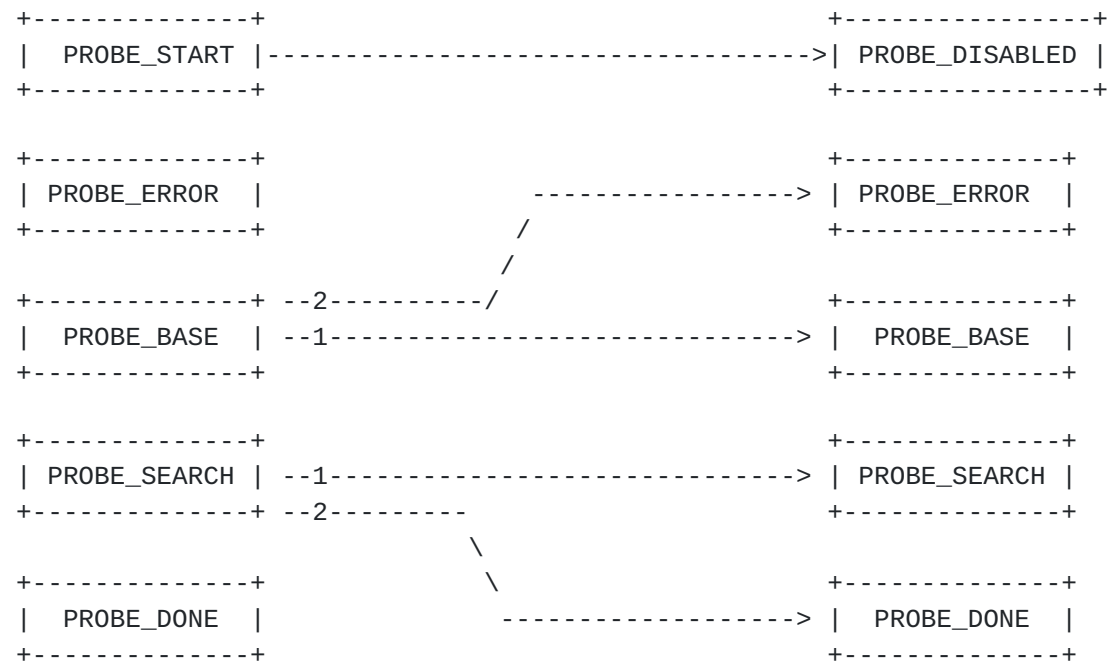
Arrival of an Acknowledgment: Depending on the probing state, the reaction differs according to Figure 7, which is a simplification of Figure 4 focusing on this event.




Condition 1: The maximum PMTU size has not yet been reached. Condition 2: The maximum PMTU size has been reached. Conition 3: Probe Timer expires and PROBE_COUNT = MAX_PROBEs. Condition 4: PROBE_ACK received.

Figure 7: State changes at the arrival of an acknowledgment

Probing timeout: The PROBE_COUNT is initialised to zero each time the value of PROBED_SIZE is changed and when a acknowledgment confirming delivery of a probe packet arries. The PROBE_TIMER is started each time a probe packet is sent. It is stopped when an acknowledgment arrives that confirms delivery of a probe packet of PROBED_SIZE. If the probe packet is not acknowledged before the PROBE_TIMER expires, the PROBE_COUNT is incremented. When the PROBE_COUNT equals the value MAX_PROBES, the state is changed, otherwise a new probe packet of the same size (PROBED_SIZE) is resent. The state transitions are illustrated in Figure 8. This shows a simplification of Figure 4 with a focus only on this event.




Condition 1: The maximum number of probe packets has not been reached. Condition 2: The maximum number of probe packets has been reached.

Figure 8: State changes at the expiration of the probe timer

PMTU raise timer timeout: The path through the network can change over time. It impossible to discover whether a path change has increased the actual PMTU by exchanging packets less than or equal to the PLPMTU. This requires PLPMTUD to periodically send a probe packet to detect whether a larger PMTU is possible. This probe packet is generated by the PMTU_RAISE_TIMER. When the timer expires, probing is restarted with the BASE_PMTU and the state is changed to PROBE_BASE.

Arrival of a PTB message: The active probing of the path can be supported by the arrival of a PTB message sent by a router or middleboxes indicating the router's local link MTU. Two cases can be distinguished:

1. The indicated link MTU in the PTB message is between the already probed and PLPMTU and the probe that triggered the PTB message.

2. The indicated link MTU in the PTB message is smaller than the PLPMTU.

In first case, the PROBE_BASE state transitions to the PROBE_ERROR state. In the PROBE_SEARCH state, a new probe packet is sent with the sized reported by the PTB message. Its result is handled according to the former events.

The second case could be a result of a network re-configuration. If the reported link MTU in the PTB message is greater than the BASE_MTU, the probing starts again with a value of PROBE_BASE. Otherwise, the method enters the state PROBE_ERROR.

Note: Not all routers include the link MTU size when they send a PTB message. If the PTB message does not indicate the link MTU, the probe is handled in the same way as condition 2 of Figure 8.

## Appendix B. Revision Notes

Note to RFC-Editor: please remove this entire section prior to publication.

Individual draft -00:

- Comments and corrections are welcome directly to the authors or via the IETF TSVWG working group mailing list.
- This update is proposed for WG comments.

Individual draft -01:

- Contains the first representation of the algorithm, showing the states and timers
- This update is proposed for WG comments.

Individual draft -02:

- Contains updated representation of the algorithm, and textual corrections.
- The text describing when to set the effective PMTU has not yet been validated by the authors
- To determine security to off-path-attacks: We need to decide whether a received PTB message SHOULD/MUST be validated? The text on how to handle a PTB message indicating a link MTU larger than the probe has yet not been validated by the authors

- No text currently describes how to handle inconsistent results from arbitrary re-routing along different parallel paths

- This update is proposed for WG comments.

Working Group draft -00:

- This draft follows a successful adoption call for TSVWG

- There is still work to complete, please comment on this draft.

Working Group draft -01:

- This draft includes improved introduction.

- The draft is updated to require ICMP validation prior to accepting PTB messages - this to be confirmed by WG

- Section added to discuss Selection of Probe Size - methods to be evlauated and recommendations to be considered

- Section added to align with work proposed in the QUIC WG.

Working Group draft -02:

- The draft was updated based on feedback from the WG, and a detailed review by Magnus Westerlund.

- The document updates RFC 4821.

- Requirements list updated.

- Added more explicit discussion of a simpler black-hole detection mode.

- This draft includes reorganisation of the section on IETF protocols.

- Added more discussion of implementation within an application.

- Added text on flapping paths.

- Replaced 'effective MTU' with new term PLPMTU.

Working Group draft -03:

- Updated figures

- Added more discussion on blackhole detection
- Added figure describing just blackhole detection
- Added figure relating MPS sizes
- Updated full state machine artwork for clarity
- Changed all text to refer to /packet probes/ /validation/ (rather than /verification/).

## Authors' Addresses

Godred Fairhurst
University of Aberdeen
School of Engineering
Fraser Noble Building
Aberdeen AB24 3U
UK

Email: gorry@erg.abdn.ac.uk

Tom Jones
University of Aberdeen
School of Engineering
Fraser Noble Building
Aberdeen AB24 3U
UK

Email: tom@erg.abdn.ac.uk

Michael Tuexen
Muenster University of Applied Sciences
Stegerwaldstrasse 39
Stein fart 48565
DE

Email: tuexen@fh-muenster.de

Irene Ruengeler
Muenster University of Applied Sciences
Stegerwaldstrasse 39
Stein fart  48565
DE

Email: i.ruengeler@fh-muenster.de