Exploring usable Path MTU in the Internet

Ana Custura
University of Aberdeen

Gorry Fairhurst
University of Aberdeen

Iain Learmonth
University of Aberdeen

***Abstract*—To optimise their transmission, Internet endpoints need to know the largest size of packet they can send across a specific Internet path, the Path Maximum Transmission Unit (PMTU). This paper explores the PMTU size experienced across the Internet core, wired and mobile edge networks.**

Our results show that MSS Clamping has been widely deployed in edge networks, and some webservers artificially reduce their advertised MSS, both of which we expect help avoid PMTUD failure for TCP. The maximum packet size used by a TCP connection is also constrained by the acMSS. MSS Clamping was observed in over 20% of edge networks tested. We find a significant proportion of webservers that advertise a low MSS can still be reached with a 1500 byte packet. We also find more than half of IPv6 webservers do not attempt PMTUD and clamp the MSS to 1280 bytes.

Furthermore, we see evidence of black-hole detection mechanisms implemented by over a quarter of IPv6 webservers and almost 15% of IPv4 webservers. We also consider the implications for UDP - which necessarily can not utilise MSS Clamping. The paper provides useful input to the design of a robust PMTUD method that can be appropriate for the growing volume of UDP-based applications, by determining ICMP quotations can be used as to verify sender authenticity.

I. Introduction

Internet standards define the Maximum Transmission Unit (MTU) [1] as the largest size of packet that may be sent across a link, without requiring fragmentation. Endpoints also maintain a Path MTU (PMTU) representing the lowest MTU along a path to a specific endpoint [2]. Since this value is not immediately known to a sender, PMTU Discovery (PMTUD) has been specified for IPv4 [3] [4] and IPv6 [5].

PMTUD provides a network layer mechanism to dynamically determine the PMTU, utilising ICMP messages returned by each router when it encounters a packet larger than its local link MTU. For IPv4, an ICMP Type 3 (Destination Unreachable) message is returned to the source of the packet, with the code "fragmentation is needed and DF set" [3] for packets that set the Don't Fragment (DF) flag. While IPv4 permits router fragmentation when the DF flag is not set, fragmentation can only happen at the source for IPv6. IPv6 sets a minimum MTU for IPv6 links of 1280 bytes and requires nodes to implement PMTUD [5]. IPv6 PMTUD relies on ICMP Packet Too Big (PTB) messages. For paths which cannot support a 1280 byte MTU, fragmentation and reassembly functions must be implemented at levels below IPv6 [6].

This paper presents a survey of IPv4 and IPv6 PMTUD behaviour and also seeks to identify which mechanisms are used to make PMTUD more robust for TCP traffic. When determining PMTUD behaviour, we use existing measurement methodologies and tools [7] [8], providing an up-to-date PMTUD behaviour report for both IPv4 and IPv6.

We include an in-depth exploration of server and client advertised Maximum Segment Size (MSS), taking advantage of existing measurement platforms, MONROE [9] and RIPE Atlas [10], and using or extending existing tools Netalyzr [11] and PATHspider [12].

We also look at ICMP quotation health and analyze a large number of ICMP quotations from a previous large-scale measurement campaign. These results provide important insight to inform the design of new methods that can robustly discover the PMTU for UDP traffic where there are currently no standard methods to improve robustness. Such work is particularly important with the growing volume of UDP-based traffic following the emergence of the QUIC [13] web transport protocol, and its current standardisation in the IETF QUIC working group.

The following sections describe the background to our study, including a survey of related work. We then present our measurement campaigns using active packet probes and the results, which we discuss in the context of recent work in the IETF concerning IPv6 PMTUD. We also present the implications of our results for Packetization Layer PMTUD [14] and summarise our findings in the conclusion.

II. Background to PMTUD

While PMTUD provides a mechanism to discover the minimum MTU along an Internet path, there are occasions when it fails in ways that are difficult to diagnose [15].

Historically, the success of PMTUD has been impacted by the way ICMP messages are handled in the network [16] [17]. Firewalls that discard ICMP messages make this method unreliable, as may routers and denial of service (DOS) prevention tools that rate-limit ICMP messages.

Tunnel endpoints also need to interact with ICMP messages to ensure they are correctly forwarded, as do devices that provide load-balancing such as Equal Cost Multipath Routing (ECMP) [18]. ECMP uses a hash of the connection 4-tuple. In this case, ICMP may not be forwarded on the correct path, as was the case for Cloudflare 2015 outage [19] where ICMP messages were occasionally handled by the wrong server - not the one initiating the PMTUD connection.

Blackholing can also result from inconsistent configuration. A router will not generate ICMP PTB messages when the router believes the local link can forward a packet, but the local interface then fails to send it [17]. Despite these challenges, past research [8] [7] has shown PMTUD success.

A. TCP Maximum Segment Size

TCP connections can utilise a mechanism to avoid the need to use PMTUD. The TCP MSS is defined as the largest TCP segment that can be accepted by the remote endpoint of a TCP connection ([20] [21]). Receivers can advertise their MSS using a TCP Option at connection setup. For example, an IPv4 link with a 1500 byte MTU and no other overhead, typically results in a MSS of 1460 bytes. The received MSS dictates the maximum size that the sender may use for the PMTU to the endpoint. When no MSS is advertised, the default is assumed to be 536 bytes.

Although the MSS Option was originally intended to be sent end-to-end, some middleboxes adjust this TCP MSS Option in an attempt to pre-empt or avoid PMTUD being triggered, behaviour known as MSS Clamping. These devices reduce the advertised MSS to take into account the actual MSS they expect to experience along the path. For instance, a tunnel could reduce the MSS to account for the additional encapsulation headers that it adds. This technique is widely used by home broadband routers, e.g. when supporting PPPoE. [22] suggests there is also evidence that some middleboxes pro-actively reduce the MSS, to account for overhead that may commonly be added elsewhere on the path. Similarly, a server or front-end middlebox may advertise a low MSS to clients behind tunnels, or clear the IPv4 DF flag when sending large packets.

PMTUD can be extended to provide protection from black-holing. to discover the PMTU by the transport protocol (packetization layer) using PLPMTUD [14]. This works by probing the path with increasingly large packets to determine a usable PMTU. When a probe fails, the packet size is lowered, and new probes sent until the packet size converges on the PMTU. This algorithm can be used alongside PMTUD to detect ICMP black-holes, or by itself to determine the PMTU. Although specified in 2007, more than 10 years later PLPMTUD is still disabled by default in the Linux kernel, and has only been added in 2014 to FreeBSD.

III. Related work

A large scale passive-measurement paper [22] from 2011 used packet traces to study IPv4 TCP advertised MSS. It found a trend towards servers advertising values smaller than 1460 and attributes this to servers compensating for tunnel overheads. They found an MSS of 1460 accounted for almost half of the observed advertisements. They also noted that advertised MSS is not a reliable indicator of the MSS that will be used by a sender.

Our work uses active probes to collect the advertised MSS from a large number of clients and webservers. We survey webservers corresponding to the entire Cisco top 1 Million domains, and clients from different types of edge networks - wired, wireless and mobile and differentiate between the MSS advertised in each case. We include data on the MSS advertised by IPv6 servers (absent in [22]). We also analyze how the advertised MSS is used as a mechanism to prevent PMTUD failure.

TABLE I
Experiments and datasets

Purpose	Tool used	Dataset Name	Date
Collect server MSS	PATHspider	A.1 "PATHspider"	Jan 2018
Validate server MSS	Ping	A.2 "Ping"	Feb 2018
Collect wireless/mobile client MSS	Pathtrace	B.1 "MONROE"	Dec 2016
Collect wired edge client MSS	RIPE Atlas Traceroute	B.2 "RIPE"	Jan 2018
Explore server PMTUD	Scamper	C.1 "Scamper"	Jan 2018
Explore edge PMTUD	Netalyzr Traceroute	C.2 "Netalyzr"	Dec 2017
Inspect ICMP quotations	Pathtrace	D "ICMP"	Jan 2017

A large measurement study of PMTUD behaviour using Scamper [7], found that at least 80% of the tested IPv4 and IPv6 webservers took the correct action when receiving a PTB message. The study found that 10% of webservers only sent packets of size 1380 bytes and advertised an MSS of 1380 bytes, and that these were more likely to fail at PMTUD. Their paper also tested 910 IPv6 targets (out of only 1027 IPv6 hosts in the Alexa Topsites list at the time of the study).

PLPMTUD and other methods for black-hole detection employed by Linux, MacOS and Windows are also outlined in [7], but their data shows no evidence of these. We contribute a refresh on their study using the same methodology, for both IPv4 and IPv6. Since their study, IPv6 adoption has increased, and our paper uses Scamper, version 20171204, to test 4,324 unique IPv6 hosts in the Cisco Umbrella top 1 Million list. We also evaluate PMTUD behaviour in combination with measurement of the TCP advertised MSS. The results are interpreted based on an understanding of recent changes in link and IP technologies.

A study [23] of ICMP quotations collected with traceroute-style measurements looked at instances where quotations did not match the sent probe. They found that a very small number (26) of routers inserted an MSS option in-line with the TCP header and asserted that this was added as a workaround for paths with broken PMTUD. They also observed the length of the quoted packet and found that the majority of routers (87.5%) only quote the first 28 bytes of the probe packet. We repeat the methodology in [23] on 20,000 ICMP and ICMPv6 quotations. Our contribution examines whether the ICMP messages triggered by routers in response to packets too large to forward can be used to verify ICMP sender authenticity.

IV. Experiment Design

Our paper employs several tools, platforms and experiments to explore various aspects of PMTUD. Table I presents a summary of the used tools, the purpose and names of each resulting dataset, as well as the month when each measurement was finished.

To collect advertised MSS for servers in the Internet core, we used PATHspider, a tool for A/B path transparency testing. This was used to send an HTTP request to a target and collect the advertised MSS on the return path. We tested the Cisco

Umbrella top 1 million[1] in December 2017 from the Janet[2] academic network (Dataset A.1, "PATHspider").

For the same targets, an ICMP ping test subsequently determined whether or not the target chooses its advertised MSS because of a MTU constraint, or sets a lower MSS (presumably expecting to reduce the likelihood of black-holing). This resulted in Dataset A.2, "Ping".

We explored MSS in the wired edge by measuring the advertised MSS from over 3,000 RIPE Atlas [10] probes across a range of access network technologies (Dataset B.1, "RIPE").

We also analyse the MSS in the mobile edge. This used data already collected using Pathtrace in February 2017 [24] for clients in mobile edge networks. The clients measured the European MONROE [9] platform providing over 120 vantage points spanning 12 mobile providers and several edge networks. Probes were sent using traceroute to a selection of 1000 webservers. We analyze the advertised MSS along the path to the target servers as returned in ICMP quoted packets. The results constitute Dataset B.2, "MONROE".

To survey PMTUD behaviour, we perform tests using Scamper [25], from a server in the Janet academic network. We observe responses to HTTP GET requests towards 60,000 domains using both IPv4 and IPv6 addresses. The Scamper methodology [8] takes the MTU to be tested as a parameter. If the HTTP response of the target is larger than the tested MTU and has the DF set, Scamper sends an ICMP Destination Unreachable or PTB message towards the target. If the target reduces the size of its packets, we infer PMTUD was successful. If no response, we allow four retransmissions before concluding that PMTUD fails. Scamper also records whether the tested server retransmits IPv4 packets clearing the DF flag or if the observed packets did not have the DF flag set. These results are found in Dataset C.1, "Scamper".

To complement the data collected for the wired edge, PMTUD from the mobile edge also explored measurements from 50 vantage points using both traceroute and Netalyzr [11] - Dataset C.2, "Netalyzr".

Finally, in Dataset D, "ICMP", we explored the quotations in ICMP messages returned by routers. These messages are required to return (quote) the packets that triggered each message. This was explored to evaluate whether quotations can be used to verify the authenticity of an ICMP message. We dissected ICMP and ICMPv6 Time Exceeded messages collected from a previous traceroute-style measurement campaign towards 100,000 webservers. Although these messages were returned due to a packet′s TTL/Hop-count reaching zero, we expect the same quotation method to be used for all ICMP messages and therefore expect the same result when PTB messages are generated.

For all tests, Internet names were resolved using *Hellfire*[3]. When a name resolved to more than one address, one IPv4 and one IPv6 addresses were selected to diversify the sampling. The Janet academic network has been tested previously and is known to not modify TCP options or IP headers.

[1]https://umbrella.cisco.com/blog/2016/12/14/cisco-umbrella-1-million/

[2]https://www.jisc.ac.uk/janet

[3]https://github.com/irl/hellfire

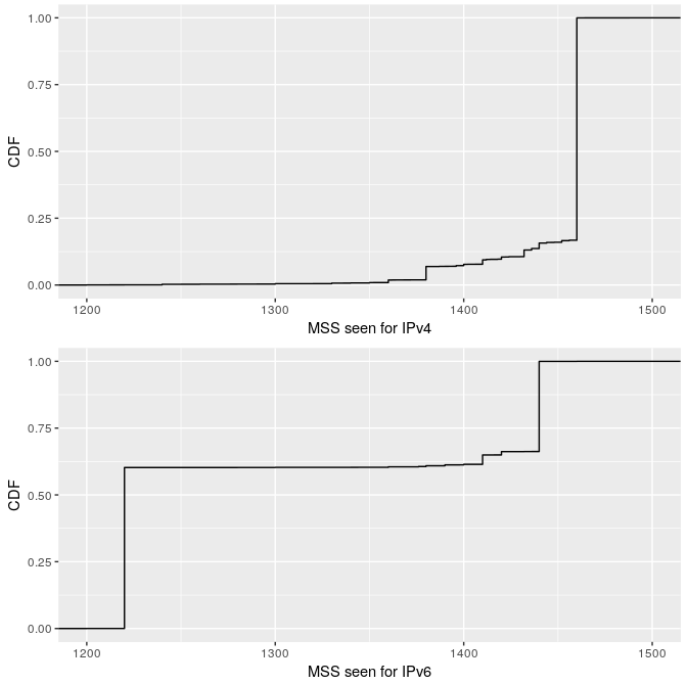

Fig. 1. Avertised MSS (in bytes) on TCP SYN/ACK server response seen at Janet academic network

V. Results

A. Server advertised MSS

This section considers results in Dataset A.1, "PATHspider", and A.2, "Ping". Out of the total of 405,623 analyzed IPv4 servers in Dataset A.1, over 80% (333,684, 82.25%) report an MSS of 1460 bytes, corresponding to an MTU of 1500 bytes. The next most common MSS values are 1380 (5%) and 1432 bytes (2.4%). Out of 107,207 IPv6 targets, around 60% advertised the minimum IPv6 MSS, 1220, and only 33.5% advertised a default MSS of 1440 bytes. The most common value after that is 1410, 3.5% of the total. Advertised MSS values between 1200 and 1500 bytes are further presented in Figure 1.

36 tests reported the maximum possible MTU, with a value of 65k. This advertised MSS value was also observed in [22] and deemed impractical, as at the time there was no technology supporting segments of that size. We traced our results to a cloud provider using Infiniband [26], this link-layer technology can use jumbo frames of up to 65k [27] in a data centre.

For Dataset A.2, we conducted a ping test on 295,000 webservers from Dataset A.1, to determine how many targets are reachable with a ping packet equal to 1500 bytes. We also sent a control probe with the size based on the advertised MSS. The reason our control probe is not a 64 byte ping is that we found many webservers to firewall large ICMP packets.

TABLE II
PING EXPERIMENT RESULTS n=295,488

Behaviour	hosts	%
No response to ping from Server	101259	34.2%
Both probes failed with PTB message	657	0.2%
Both probes succeeded	190219	64.3%
1500 byte probe failed	2378	0.8%
1500 byte probe failed with PTB message	568	0.2%

TABLE III
SUMMARY FOR MSS OPTIONS SEEN ON THE RETURN PATH

$n = 4088$		
hosts	%	Advertised MSS
323	8%	1400 bytes
323	8%	1410 bytes
174	4.2%	1420 bytes
68	1.6%	1452 bytes

Out of the total, two-thirds (65.8%) of targets were reachable with our probes. Out of the reachable servers, 98.4% of servers responded to a ping of 1500 bytes, indicating the path supported this PMTU. For 1.2% which did not, we saw an ICMP PTB message in 24% of cases, as outlined in Table II. Out of the subset which advertised an MSS lower than 1460 (34,920), 93% were reachable with a probe of 1500 bytes.

We found 657 or 0.3% of the total reachable servers failed both probes with PTB messages, which indicates that respecting the server advertised MSS results in packets too big for those paths.

B. MSS seen from mobile and wireless clients

Dataset B.1, "MONROE", consists of traceroute-style measurements collected from the MONROE [9] platform. This platform contains nodes with a mixture of cellular and wireless edge interfaces.

We used the default interface MTU of MONROE nodes (1500 bytes) and did not send any additional TCP options. Our tests observed a total of 888 (21%) hops that returned a quotation with an MSS option, evidence of MSS Clamping in the mobile and edge networks. In comparison, only two replies in the Internet core dataset indicated rewriting of the MSS by a device on the path.

Table III shows a low number of paths clamped the MSS to 1452, the result of including some wireless edge networks in the dataset. A MONROE node may have up to three mobile interfaces and up to one wireless interface, resulting in an MSS of 1452. This is a common MSS advertised by routers behind a PPPoE link. 1452 is also a prominent value advertised in [22], where it is also attributed to DSL subcribers behind a PPPoE link.

The other values encountered suggest the networks have been configured to accommodate a sizeable encapsulation overhead and perform MSS Clamping to avoid fragmentation inside a tunnel. We sent 1500 byte UDP probes with the DF flag set on ten of the mobile paths that were found to insert an MSS option. These packets traversed the network and arrived at our server, even when the probe size was larger than the MSS, as validated in Section V-D. For these 10 paths, we also sought to determine which mobile network inserted the MSS option. We found a variety of networks, displayed in Table IV.

TABLE IV
INSERTED MSS OPTIONS BY MOBILE NETWORK, $n = 10\ paths$

Network	Inserted MSS option
Telenor Norway	1410 bytes
Telia Sweden	1400 bytes
Vodafone Italy	1400 bytes
Wind Italy	1420 bytes

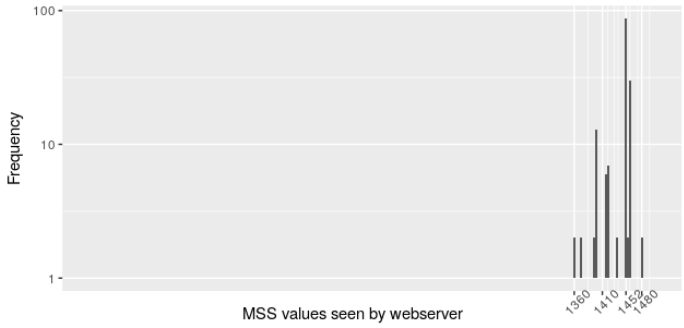

Fig. 2. Observed MSS Values for RIPE Atlas test

C. MSS seen from the wired edge

To evaluate the wired edge, we performed a TCP traceroute from 3000 RIPE Atlas probes towards a server controlled by the authors in the Janet academic network (Dataset B.2, "RIPE"). When performing TCP traceroute tests, RIPE Atlas probes send a TCP SYN packet with no options. Our webserver advertises an MSS of 1460 when replying. RIPE Atlas allows us to observe the MSS received from our server over the return path. At the same time, we capture the MSS that was advertised by the probes. This allows us to infer whether MSS manipulations in the path are symmetric.

4.8% of the sent probes arrive at our server carrying an MSS option. This option must have been inserted by a middlebox on the path. The observed MSS values are presented in Figure 2.

Devices on three of the paths had set the MSS to over 1460, which, if used, would exceed the MTU of standard Ethernet. Most paths clamped the MSS to 1452 bytes (Ethernet over PPPoE encapsulation) or to 1460 bytes. We isolated the paths for which this happens, and looked at the corresponding MSS seen by the probe. We found the change to be symmetric in 88% (140) of cases, implying middleboxes perform the same change bidirectionally.

Significantly more (23% or 764) of the MSS values received by the probes differ from the value that was sent. This suggests that middleboxes are more likely to interfere with an MSS value only if it exists. The most frequent change is to set the MSS to 1452 (44%), followed by 1412 (9%) and changes range between 536 and 9176 bytes (!).

D. MTU in the Internet

We analyzed PMTUD behaviour for a random 60,000 domains chosen from the Cisco Umbrella top 100,000, in dataset C.1, "Scamper". We first resolved domain names to IP addresses, and selected the first IPv4 and IPv6 addresses as targets for a Scamper PMTUD test using the domain name as the URL to test with HTTPS. Where the same IP address appeared more than once in the list, the test result was only counted one time.

The test MTU values were 1420 and 576 bytes and in both cases we advertised an MSS of 1460. The test was then repeated, but filtering the local ICMP PTB messages, using the 'blackhole' option supplied with the tool. Filtering prevents classical PMTUD from detecting the lower MTU, allowing us to determine whether some other PMTU method was being used (such as PLPMTUD, or black-hole detection).

We exclude from our results cases where TCP did not connect, there was no data received from the server, or the received packets were smaller than the tested MTU (around 20% of the total tested). Some webservers may reset the connection due to SNI mismatch, and not all domains in the Top 1M list may point to webservers.

We observed that around 12% of targets did not set the DF flag in packets that they sent. This shows that these endpoint are not configured to perform PMTUD. For IPv4, almost 67% of the remaining set succeeded in performing PMTUD. 16% failed to complete the connection when the test MTU was 1420. The rate of success further reduces when the test MTU was 576, resulting in almost 20% failure.

For an MTU of 1420, 2.7% IPv4 servers did not reduce their sending PMTU, but instead cleared the DF flag on subsequent packets. A slightly higher percentage (4.1%) of servers do the same for an MTU of 576. When the bottleneck did not generate ICMP messages, the PMTU success drops to 8.2%, with 67.4% failing to reduce their packet size to a level below the MTU tested and not completing the test. We saw no servers that cleared the DF flag as a response to a possible ICMP black-hole. IPv4 behaviour is detailed in Table V.

For IPv6, we set an MTU of 1280. For more than half the results, replies of exactly 1280 bytes size were received, which were not sufficiently large for a PTB message to be sent. These servers were configured to use the minimum IPv6 MTU. For the remaining servers, the test with this MTU succeeded for 95% of tests. When the bottleneck did not generate ICMP messages, 32% of servers still reduced their packet size and successfully completed the web transfer with the test MTU. IPv6 behaviour is detailed in Table VI. Table VII presents a side-by-side comparison with Luckie et al.[7] for the same tested MTU.

Futher to the Internet core experiment, we used traceroute PMTUD and Netalyzr in the mobile edge to probe Internet paths to remote webservers using 16 mobile operators from over 40 vantage points within the MONROE platform (Dataset C.2). Both experiments consistently reported a PMTU of 1500 bytes, the default Ethernet MTU.

TABLE V
IPV4 PMTUD BEHAVIOUR

	1420 MTU	576 MTU	576 MTU Black-hole
PMTUD Too Small	7.45%	3.7%	0.95%
PMTUD Success	68.2%	63.9%	8.2%
PMTUD Failure	16.4%	19.5%	67.4%
No DF set	12.5%	12.3%	15.2%
Clear DF	2.7%	4.1%	NIL

TABLE VI
IPV6 PMTUD BEHAVIOUR

	1280 MTU	1280 MTU Black-hole
PMTUD Too Small	59.6%	53.1%
PMTUD Success	95.5%	32%
PMTUD Failure	4.5%	67.9%

E. ICMP quotation dissection

A recent update to the IPv6 base specifications [5] noted the need for methods to validate ICMP PTB messages, to avoid vulnerabilities to off-path denial of service attack. To generate ICMP packets with quotations, we used probe packets with an engineered TTL, which triggered an ICMP response from routers along the path. Our goal was to determine if the quoted packet in a received ICMP could be used to validate that the message was sent in response to a packet from the sender.

For network tests across the Internet core we recorded a total of 125,212 ICMP and ICMPv6 replies (Dataset D, "ICMP"). For the purpose of this analysis we consider replies from unique IP addresses, 14257 in total. For the mobile edge, a total of 4,772 ICMP type 11 replies originating from unique IP addresses were analyzed.

We assess a reply as healthy if:

1) the quoted payload is at least 28 bytes in length
2) the quoted payload matches the original probe on the source address, destination address, source port, destination port and transport protocol

Most quotations were observed to have the minimum permitted size by RFC792, (64.5%) for IPv4 and (89.8%) for IPv6. For both IPv4 and IPv6, the second most popular choice (19.4% and 9.8%) is to quote the original packet including ICMP MPLS extensions [28] (while ICMP TTL exceeded in transit messages may be extended, ICMP PTB must not [29]).

TABLE VII
PMTUD BEHAVIOUR COMPARISON WITH LUCKIE ET AL. 2010

	IPv6 1280 MTU		IPv4 576 MTU	
	2010 [7]	2018	2010 [7]	2018
No TCP connection	N/A	6.9%	1.6%	14%
Early TCP reset	0.1%	0.2%	0.3%	0.6%
No data packets	0.4%	1.1%	0.2%	1.1%
Data packets too small	17.2%	59.6%	3.9%	3.7%
DF not set	N/A	N/A	2.2%	9.1%
Clear DF after PTB	N/A	N/A	4.5%	3.3%
PMTUD Success	80%	30.4%	78%	51%
PMTUD Failure	2.2%	1.4%	9.4%	15.5%

Side-by-side results for the same tested MTU. 2010 data from [7], with IPv6 results from Amsterdam and IPv4 results from Hamilton, NZ.

TABLE VIII
SUMMARY FOR ICMP QUOTED PAYLOAD LENGTHS, CORE VS. MOBILE

Core $n = 10169$		Mobile $n = 4398$		Size in bytes and description
hosts	%	hosts	%	
7075	69.5%	2887	65.4%	28 (minimum size for IPv4)
846	8.3%	103	2.3%	40 (original IPv4 packet size)
NA	NA	72	1.6%	44 (original IPv4 packet w/4 bytes options)
0	0%	816	18.5%	68 (original IPv4 packet w/20 bytes options)
1944	19.1%	390	8.9%	140 (quotation w/MPLS extensions)
301	3%	144	3.3%	144 (quotation w/MPLS extensions)

TABLE IX
SUMMARY FOR ICMP QUOTED PAYLOAD LENGTHS FOR IPV6

IPv6 $n = 4088$		Size in bytes and description
hosts	%	
3673	89.8%	60 (original IPv6 packet size)
12	0.3%	140 (quotation w/MPLS extensions)
388	9.5%	144 (quotation w/MPLS extensions)
15	0.3%	148 (quotation w/MPLS extensions)

We found no evidence of quotations smaller than the minimum permitted size, which means all information required to match a probe is returned. Data are presented in Tables VIII and IX.

In the Internet core dataset, we found a total of 12 replies for which the quoted transport payload did not match the probe that was sent. These replies correctly quote source and destination address, but include corrupted transport payloads - which could not be used to validate port information in the quoted packet. We considered all other replies from these IP addresses and found that, with one exception, they consistently return broken ICMP quotations. Furthermore, this only appears on the paths from three of the 8 vantage points tested. However, due to the nature of the payload this does not appear to be due to a decapsulation issue. No broken payloads were found in mobile edge tests.

VI. DISCUSSION

This section provides a discussion of the reported results from three perspectives.

A. What PMTU can be expected?

We found that 99% of MSS values seen client-side, for both IPv4 and IPv6 range between 1200 and 1460 bytes inclusive. For server MSS advertisements, the same figure is 98.95%. Including values from 1000 bytes do not appear to offer additional benefit, the percentages remain the same (99.1%, 98.96%). The most commonly advertised IPv4 MSS is still 1460 bytes, experienced to a larger extent than found in [22], although there is still a trend towards advertising smaller values (as noted later). We found 1% of IPv4 and IPv6 webservers deploy MTUs larger than 1500 bytes, with notable values corresponding to Ethernet Jumbo frames (8k) and other link technologies, such as Infiniband (65k).

Our results also show a wide range of deployed link MTUs. In many cases the size of PMTU can be much higher than the default MTU imposed by Ethernet, and there remains the potential to increase the PMTU as this deployment increases. However, at the same time, sending endpoints need a way to confirm a workable PMTU for the paths they use.

Our results in Section V-D show a large proportion of IPv6 webservers, over 50%, that do not attempt to send TCP segments larger than 1280 bytes. The first peak of Figure 1 also shows up to 60% of webservers advertise an MSS value of 1220 bytes. This implies more than half of the tested IPv6 hosts are configured to advertise a MSS corresponding to the minimum IPv6 MTU. While this alleviates a sending endpoint from the need to utilise PMTUD, it also limits the efficiency of transmission.

B. What potential is there for PMTUD to raise the PMTU?

Classical PMTUD is a standard IETF-defined method for determining an effective PMTU for IPv4 and IPv6. It can probe paths regularly to detect PMTU changes and although it usually results in reducing the MTU for a given path (including black-hole detection), the same mechanism can also raise it.

95% tested IPv6 servers succeed in performing PMTUD to reduce the PMTU, when they receive the appropriate ICMP PTB messages. Our IPv6 results are consistent with the results found by [7]. Both results are presented in Table VI.

PMTUD was also found to be successful in adjusting the PMTU for up to 68% of IPv4 servers tested (see Section V-D). However, the method fails for up to 20% of tests. Comparing our results in Section V-D to [7], our tests experienced almost twice as much PMTUD failure failure for IPv4. 9.1% of IPv4 endpoints were configured not to even attempt PMTUD (i.e., did not set the DF flag on packets they sent). This is almost five times more endpoints that reported by [7]. Results are presented for comparison in Table VII.

[5] calls for validating the source of PTB messages using the quoted packet information, as a way of reducing the risk from off-path denial of service attacks. Our results indicate that validation is feasible, but we note that this requires an endpoint stack implementing PMTUD to access the port information to validate the quoted packet.

For IPv6, a TCP endpoint can avoid PMTUD by advertising an IPv6 MSS of 1220 bytes by configuring the minimum IPv6 MTU of 1280 on the host interface. We have seen evidence of both in Section V-D and Figure 1. This advertises a lower MSS than is actually supported by the part of the path they control. This practice has also been reported by others [19]. Our results in Section V-A suggest although 93% of servers advertise an MSS lower than 1500 bytes, they were were still reachable with a packet of size 1500 bytes.

We found a popular choice (Figure 1) was to advertise an MSS of 1380 bytes. This corresponds to the size using IPv4 ESP in Tunnel Mode [30]. Our results are consistent with reports of middleboxes setting the default advertised MSS to 1380 (e.g. Citrix CloudBridge WAN traffic accelerators [31], Cisco ASA appliances [32]).

There is also evidence of rewriting the TCP MSS option by middleboxes. MSS Clamping was observed for both mobile

and edge networks and we presented results showing MSS Clamping in access networks. In some cases, this appeared to be performed to avoid PMTUD failures, presumably because of locally known links with a smaller MTU. In the case of mobile networks, some middleboxes were also found to add an MSS option reducing the MSS, even though the network path actually supported an MTU of 1500 bytes.

In summary, the use of these mechanisms suggest manufacturers and network administrators do not trust PMTUD. While MSS Clamping reduces the risk of segment sizes exceeding the PMTU, it prevents endpoints that can support a larger MSS from taking advantage of paths that are able to forward these larger packets. MSS Clamping is also not without issues: we found a number of misconfigured middleboxes which set the MSS to a larger number than the network would support (see Section V-C), which could lead to black-holing of segments that then exceed the PMTU.

C. What are the prospects in using PLPMTUD to raise the PMTU?

For PMTUD to succeed without the support of receiving ICMP messages requires sending endpoints to deploy some form of black-hole detection. 68% IPv6 and 76% IPv4 tested webservers did not reduce their PMTU when a limiting link MTU was connected via a router that was unable or unwilling to return PTB messages to the sender.

On Windows platforms, PMTUD black-hole detection can be enabled as a feature, but only for TCP traffic. It works by reducing segment size to 536 bytes and setting the DF flag on outgoing packets when no acknowledgement is received after retransmission [33] [34]. Similar results could be achieved for TCP by enabling PLPMTUD on Linux servers.

PLPMTUD provides another way to implement black-hole detection for TCP. This is able to detect and respond to black-holes. It also performs robust path probing to allow senders to discover a larger PMTU, when permitted by the receiver-advertised MSS, thus raising the MTU for a path. The PLPMTUD implementation in Linux has had several bugs [35]. Following patches in 2015, and from Linux Kernel 4.1, PLPMTUD starts the search from a default PMTU value of 1024, previously 512 bytes. Increasing the base PMTU value to 1200 could improve search times when probing for an upper limit. The availability of data showing the achievable PMTU is expected to be an important method in evolving suitable probing algorithms for PLPMTUD.

We would encourage continued research to develop support for PLPMTUD, since the robustness it provides is desirable in the increasingly heterogenous Internet. However we also note that the use of pre-emptive MSS Clamping (i.e. by boxes that do not themselves have a MTU limit) is likely to result in obstacle to detecting and using a larger PMTU for many network paths.

UDP traffic represents an increasing volume of Internet traffic. The volume and importance of this traffic is only set to increase as QUIC is further deployed for web transport. However, there is no standardised method to improve the robustness of PMTUD for UDP traffic. Google data analysing UDP traversal [36] suggests that up to 64% of access networks fail to carry 1500 byte UDP packets with the DF flag set. This means that some form of PMTUD is therefore desirable to select an appropriate packet size. However, our results show that it would be unwise to rely on classical PMTUD. Blackhole detection mechanisms are primarily implemented for TCP. Methods involving the setting of MSS options and the use of MSS Clamping do not work for transports other than TCP. Our results therefore suggest there could be a failure rate of more than 19% for an MTU under 1500 bytes. In response to these challenges, Datagram PLPMTUD is being designed [37].

VII. Conclusion and next steps

This paper provides new measurement data on the use of PMTUD. It describes the PMTU values seen for current Internet paths and the implications of receiver-advertised MSS on the PMTU reached for TCP traffic. Our results are based on active probing of network infrastructure from the wired edge, mobile edge and from a core vantage point.

We found that more servers are not using PMTUD than previously reported, and a considerable proportion already employs ICMP black-hole detection mechanisms. We also observed a range of methods being used by network administrators and stacks that seek to mitigate or avoid PMTUD failure by adjusting the receiver-supplied MSS value. We observed an increase in the numbers of IPv6 nodes that were configured to advertise a MSS based on the minimum IPv6 MTU, rather than their link MTU. This may also be an attempt to avoid the implications of PMTUD failure. There was also evidence of MSS Clamping in both the wired and mobile edge, particularly by routers that we presume connect PPPoE links and in middleboxes in the mobile network. Although successful in avoiding the need for PMTUD, such MSS-based solutions are only applicable to TCP.

There is also an interest in providing robust PMTUD for the growing share of UDP traffic. We therefore explored the implications on non-TCP traffic and the paper provides input to the design of future robust techniques for datagram PMTUD.

VIII. Acknowledgements

This work is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement No. 644399 (MONROE) through the Open Call. Additionally this work was partially supported by the European Union's Horizon 2020 research and innovation programme under grant agreement No. 688421 (MAMI). The opinions expressed and arguments employed reflect only the authors' view. The European Commission is not responsible for any use that may be made of that information.

References

[1] R. Braden, "Requirements for Internet Hosts - Communication Layers," RFC 1122, Internet Engineering Task Force, Oct. 1989. [Online]. Available: http://www.ietf.org/rfc/rfc1122.txt

[2] C. A. Kent and J. C. Mogul, "Fragmentation considered harmful," *SIGCOMM Comput. Commun. Rev.*, vol. 25, no. 1, pp. 75–87, Jan. 1995.

[3] J. Mogul and S. Deering, "Path MTU discovery," RFC 1191, Internet Engineering Task Force, Nov. 1990. [Online]. Available: http://www.ietf.org/rfc/rfc1191.txt

[4] J. McCann, S. Deering, and J. Mogul, "Path MTU Discovery for IP version 6," RFC 1981, Internet Engineering Task Force, Aug. 1996. [Online]. Available: http://www.ietf.org/rfc/rfc1981.txt

[5] J. McCann, S. E. Deering, J. Mogul, and R. M. Hinden, "Path MTU Discovery for IP version 6," RFC 8201, Jul. 2017. [Online]. Available: https://rfc-editor.org/rfc/rfc8201.txt

[6] D. S. E. Deering and R. M. Hinden, "Internet Protocol, Version 6 (IPv6) Specification," RFC 8200, Jul. 2017. [Online]. Available: https://rfc-editor.org/rfc/rfc8200.txt

[7] M. Luckie and B. Stasiewicz, "Measuring Path MTU Discovery Behaviour," in *Proceedings of the 10th ACM SIGCOMM Conference on Internet Measurement (IMC '10)*. New York, NY, USA: ACM, 2010, pp. 102–108.

[8] A. Medina, M. Allman, and S. Floyd, "Measuring interactions between transport protocols and middleboxes," in *Proceedings of the 4th ACM SIGCOMM Conference on Internet Measurement (IMC '04)*. New York, NY, USA: ACM, 2004, pp. 336–341.

[9] MONROE, "Measuring Mobile Broadband Networks in Europe," https://www.monroe-project.eu/.

[10] RIPE NCC Staff, "RIPE Atlas: A Global Internet Measurement Network," *Internet Protocol Journal*, vol. 18, no. 3, Sep 2015.

[11] C. Kreibich, N. Weaver, B. Nechaev, and V. Paxson, "Netalyzr: Illuminating The Edge Network," in *Internet Measurement Conference (IMC)*, 2010.

[12] I. R. Learmonth, B. Trammell, M. Kühlewind, and G. Fairhurst, "PATHspider: A tool for active measurement of path transparency," in *Proceedings of the 2016 Applied Networking Research Workshop*, July 2016, pp. 62–64.

[13] R. Hamilton, J. Iyengar, I. Swett, and A. Wilk, "QUIC: A UDP-Based Secure and Reliable Transport for HTTP/2," Internet Engineering Task Force, Internet-Draft draft-hamilton-early-deployment-quic-00, Jul. 2016, work in Progress.

[14] R. Gellens, D. Singer, and P. Frojdh, "The Codecs Parameter for "Bucket" Media Types," RFC 4281, Internet Engineering Task Force, Nov. 2005. [Online]. Available: http://www.ietf.org/rfc/rfc4281.txt

[15] K. Lahey, "TCP Problems with Path MTU Discovery," RFC 2923, Internet Engineering Task Force, Sep. 2000. [Online]. Available: http://www.ietf.org/rfc/rfc2923.txt

[16] R. van den Berg and P. Dibowitz, "Over-Zealous Security Administrators Are Breaking the Internet," in *Proceedings of the 16th USENIX Conference on System Administration (LISA '02)*. Berkeley, CA, USA: USENIX Association, 2002, pp. 213–218.

[17] M. Luckie, K. Cho, and B. Owens, "Inferring and debugging path MTU discovery failures," in *Proceedings of the 5th ACM SIGCOMM Conference on Internet Measurement (IMC '05)*. Berkeley, CA, USA: USENIX Association, 2005, p. 17.

[18] C. Hopps, "Analysis of an Equal-Cost Multi-Path Algorithm," RFC 2992, Internet Engineering Task Force, Nov. 2000. [Online]. Available: http://www.ietf.org/rfc/rfc2992.txt

[19] M. Majkowski, "Path MTU discovery in practice," *Cloudflare Blog*, May 2015, (https://blog.cloudflare.com/path-mtu-discovery-in-practice/).

[20] J. Postel, "The TCP Maximum Segment Size and Related Topics," RFC 879, Internet Engineering Task Force, Nov. 1983. [Online]. Available: http://www.ietf.org/rfc/rfc879.txt

[21] D. Borman, "TCP Options and Maximum Segment Size (MSS)," RFC 6691, Internet Engineering Task Force, Jul. 2012. [Online]. Available: http://www.ietf.org/rfc/rfc6691.txt

[22] S. Alcock and R. Nelson, "An Analysis of TCP Maximum Segment Sizes," https://wand.net.nz/sites/default/files/mss_ict11.pdf, accessed 2018-01-24.

[23] D. Malone and M. Luckie, "Analysis of ICMP quotations," in *Passive and Active Network Measurement*, S. Uhlig, K. Papagiannaki, and O. Bonaventure, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 228–232.

[24] A. Custura, G. Fairhurst, and A. Venne, "Exploring DSCP modification pathologies in mobile edge networks," in *Proceedings of the 2017 IEEE/IFIP Workshop on Mobile Network Measurement*, 6 2017.

[25] M. Luckie, "Scamper: A Scalable and Extensible Packet Prober for Active Measurement of the Internet," in *Proceedings of the 10th ACM SIGCOMM Conference on Internet Measurement (IMC '10)*. New York, NY, USA: ACM, 2010, pp. 239–245.

[26] V. Kashyap, "IP over InfiniBand: Connected Mode," RFC 4755, Internet Engineering Task Force, Dec. 2006. [Online]. Available: http://www.ietf.org/rfc/rfc4755.txt

[27] Burleson Consulting, "Jumbo frames and RAC," http://www.dba-oracle.com/t_rac_tuning_jumbo_frames.htm.

[28] R. Bonica, D. Gan, D. Tappan, and C. Pignataro, "ICMP Extensions for Multiprotocol Label Switching," RFC 4950, Internet Engineering Task Force, Aug. 2007. [Online]. Available: http://www.ietf.org/rfc/rfc4950.txt

[29] ——, "Extended ICMP to Support Multi-Part Messages," RFC 4884, Internet Engineering Task Force, Apr. 2007. [Online]. Available: http://www.ietf.org/rfc/rfc4884.txt

[30] S. Kent, "IP Encapsulating Security Payload (ESP)," RFC 4303, Internet Engineering Task Force, Dec. 2005. [Online]. Available: http://www.ietf.org/rfc/rfc4303.txt

[31] Citrix, "How to Find Maximum Size of IP Data Payload that can Traverse WAN Environment Without Fragmentation," https://support.citrix.com/article/CTX115434.

[32] Cisco, "ASA Throughput and Connection Speed Troubleshooting and Analyzing Packet Captures," https://www.cisco.com/c/en/us/support/docs/security/asa-5500-x-series-next-generation-firewalls/113393-asa-troubleshoot-throughput-00.html.

[33] J. L. Carrell, E. Tittel, and J. Pyles, *Guide to TCP/IP: IPv6 and IPv4*. Boston, MA: Cengage Learning, 2016.

[34] Microsoft TechNet, "Enable PMTU Discovery," https://technet.microsoft.com/en-us/library/cc957539.aspx.

[35] Stack Overflow, "TCP_MAXSEG inaccurate? (was: Linux path MTU probing not working on accepted socket if requested using setsockopt())," https://stackoverflow.com/questions/36740276/tcp-maxseg-inaccurate-was-linux-path-mtu-probing-not-working-on-accepted-s.

[36] A. Langley, A. Riddoch, A. Wilk, A. Vicente, C. Krasic, D. Zhang, F. Yang, F. Kouranov, I. Swett, J. Iyengar, J. Bailey, J. Dorfman, J. Roskind, J. Kulik, P. Westin, R. Tenneti, R. Shade, R. Hamilton, V. Vasiliev, W.-T. Chang, and Z. Shi, "The QUIC Transport Protocol: Design and Internet-Scale Deployment," in *Proceedings of the Conference of the ACM Special Interest Group on Data Communication (SIGCOMM '17)*. New York, NY, USA: ACM, 2017, pp. 183–196.

[37] G. Fairhurst, T. Jones, M. Txen, and I. Ruengeler, "Packetization Layer Path MTU Discovery for Datagram Transports," Internet Engineering Task Force, Internet-Draft draft-ietf-tsvwg-datagram-plpmtud-01, Mar. 2018, work in Progress. [Online]. Available: https://datatracker.ietf.org/doc/html/draft-ietf-tsvwg-datagram-plpmtud-01