# Large (UDP) Packets in IPv6

Geoff Huston
APNIC

# What's the problem?

# What's the problem?

**Fragmentation and Extension Header Support in the IPv6 Internet**

**Fernando Gont**

IPV6 TOOLKIT

IEPG 88
November 3, 2013. Vancouver, BC, Canada

iEPG presentation – November 2013

## Fragmentation

- Failure rate: 47.68 %

Groups Documents Meetings Other gih@apnic.net Document search

Meetings
Agenda
Materials
Past proceedings
Upcoming
Request a session
Session requests

Other
IPR disclosures
Liaison statements
IESG agenda
Tutorials
Report a bug

| | | |
|---|---|---|
| | Send notices to | draft-ietf-v6ops-ipv6-ehs-in-real-world.all@tools.ietf.org |
| IANA | IANA review state | IANA OK - No Actions Needed |
| | IANA action state | No IC |
| RFC Editor | RFC Editor state | EDIT |

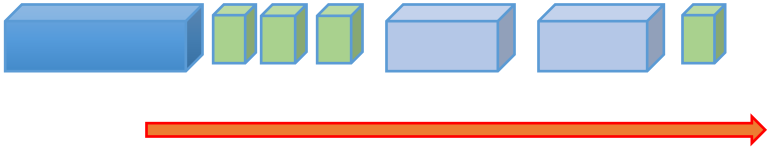
Email authors | IPR | References | Referenced by | Nits | Search lists | Track

```
IPv6 Operations Working Group (v6ops)                            F. Gont
Internet-Draft                                     SI6 Networks / UTN-FRH
Intended status: Informational                                J. Linkova
Expires: June 12, 2016                                            Google
                                                                T. Chown
                                               University of Southampton
                                                                  W. Liu
                                                     Huawei Technologies
                                                       December 10, 2015

 Observations on the Dropping of Packets with IPv6 Extension Headers in
                             the Real World
              draft-ietf-v6ops-ipv6-ehs-in-real-world-02

Abstract

   This document presents real-world data regarding the extent to which
   packets with IPv6 extension headers are dropped in the Internet (as
   originally measured in August 2014 and later in June 2015, with
   similar results), and where in the network such dropping occurs.  The
   aforementioned results serve as a problem statement that is expected
   to trigger operational advice on the filtering of IPv6 packets
   carrying IPv6 Extension Headers, so that the situation improves over
   time.  This document also explains how the aforementioned results
   were obtained, such that the corresponding measurements can be
   reproduced by other members of the community and repeated over time
   to observe changes in the handling of packets with IPv6 extension
   headers.
```

So what?

# Packet Networks like variable packet sizes

- The range of packet sizes supported in a network represents a set of engineering trade-offs:
  - Bit error rate of the underlying media
  - Desired carriage efficiency
  - Transmission speed vs packet switching speed



# IPv4 Packet Design

**FORWARD** fragmentation

- If a router cannot forward a packet on its next hop due to a packet size mismatch then it is permitted to fragment the packet, preserving the original IP header in each of the fragments

# IPv4 and the "Don't Fragment" bit

If Fragmentation is not permitted by the source, then the router discards the packet. The router may send an ICMP to the packet source with an Unreacahble code (Type 3, Code 4)

Later IPv4 implementations added a MTU size to this ICMP message

BUT: ICMP messages are extensively filtered in the Internet so applications should not count on receiving these messages!

# Trouble at the Packet Mill

- Lost frags require a resend of the entire packet – this is far less efficient than repairing a lost packet
- Fragments represent a security vulnerability as they are easily spoofed
- Fragments represent a problem to firewalls – without the transport headers it is unclear whether frags should be admitted or denied
- Packet reassembly consumes resources at the destination

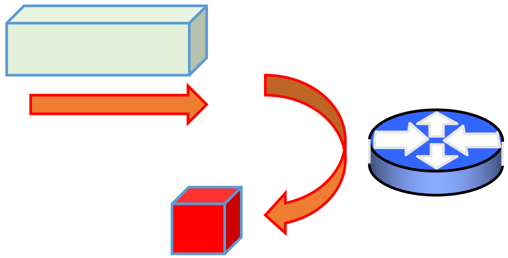
# The thinking at the time…

Fragmentation was a Bad Idea!

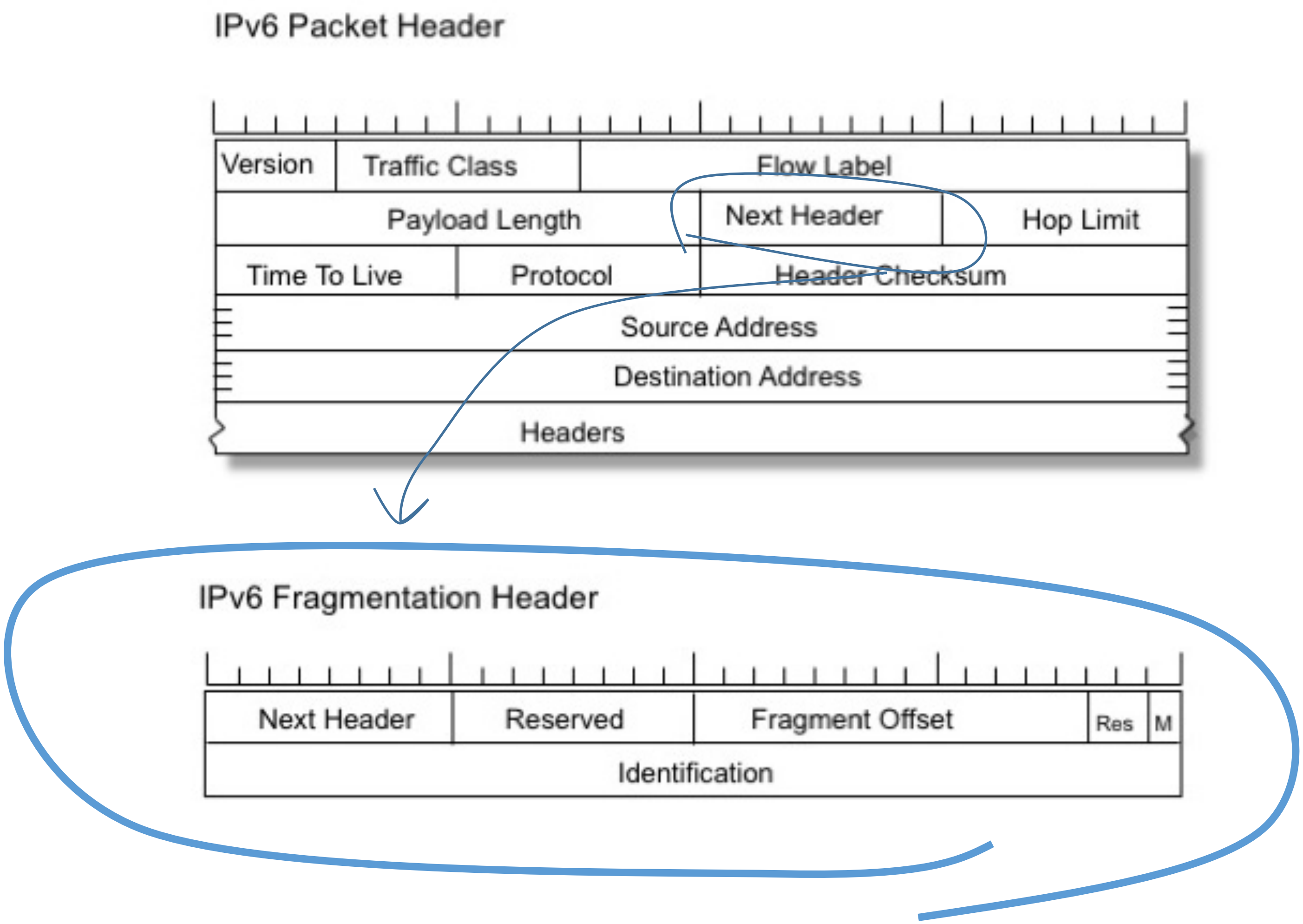
Kent, C. and J. Mogul, "Fragmentation Considered Harmful", Proc. SIGCOMM '87 Workshop on Frontiers in Computer Communications Technology, August 1987

# IPv6 Packet Design

- Attempt to repair the problem by effectively jamming the DON’T FRAGMENT bit to always ON
- IPv6 uses BACKWARD signalling
  - When a packet is too big for the next hop a router should send an ICMP6 TYPE 2 (Packet Too Big) message to the source address and include the MTU of the next hop.



# IPv6 Source Fragmentation



# What changed? What's the same?

- Both protocols may fragment a packet at the source
- Both protocols support a Packet Too Big signal from the interior of the network to the source
- Only IPv4 routers may generate fragments on-the-fly
- IPv6 relies on support for Extension Headers to support its implementation of IP packet fragmentation
  - But that has its own set of implications (See slide 3)!

# What does "Packet Too Big" mean anyway?

errrrr

# It's a Layering Problem

- Fragmentation was seen as an IP level problem
  - It was meant to be agnostic with respect to the upper level (transport) protocol
- But we don't treat it like that
  - And we expect different transport protocols to react to fragmentation notification in different ways

# What does "Packet Too Big" mean anyway?

For **<u>TCP</u>** it means that the active session referred to in the ICMP payload* should drop its session MSS to match the MTU **

In an ideal network you should never see IPv6 fragments in TCP!

* assuming that the payload contains the original end-to-end IP header

** assuming that the ICMP is genuine

# What does "Packet Too Big" mean anyway?

For **UDP** its not clear:

- The offending packet has gone away!
- Some IP implementations appear to ignore it
- Others add a host entry to the local IP Forwarding table that records the MTU
- Others perform a rudimentary form of MTU reduction in a local MTU cache

# Problems

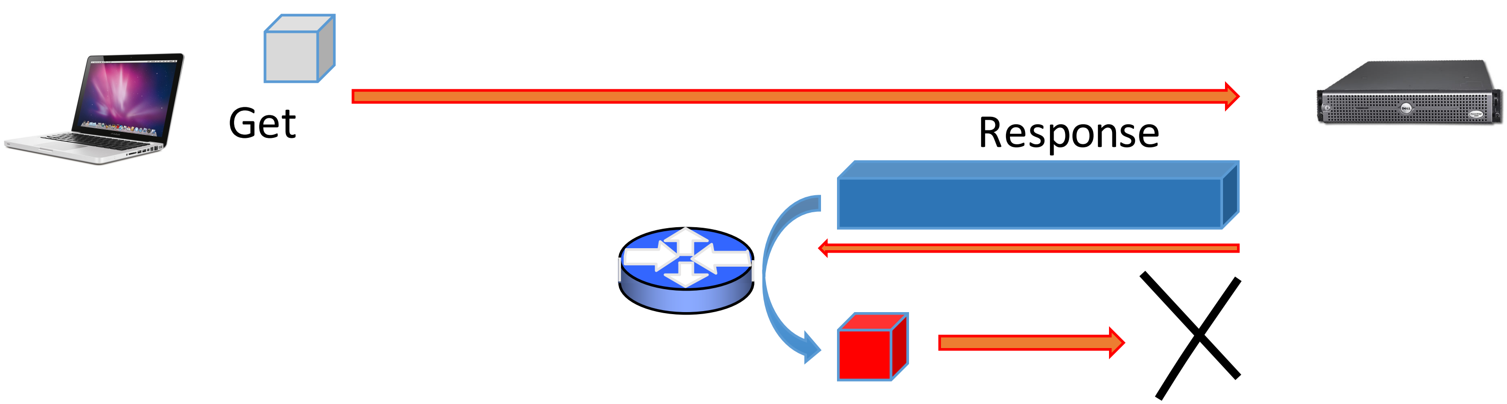
ICMP is readily spoofed

- ICMP messages can consume host resources
- An attacker may spoof a high volume stream of ICMP PTB messages with random IPv6 source addresses
- An attacker may spoof ICMP PTB messages with very low MTU values

# Problems

ICMP is widely filtered

- leading to black holes in TCP sessions
  - GET is a small HTTP packet
  - The response can be arbitrarily large, and if there is a path MTU mismatch the response can wedge



# Problems

Leading to ambiguity in UDP

- Is this lack of a response due to network congestion, routing & addressing issues, or MTU mismatch?
- Should the receiver just give up, resend the trigger query, or revert to TCP? (assuming that it can)

# What did IPv6 do differently?

IPv6 defined a minimum unfragmented packet size of 1,280 bytes:

IPv6 Specification: RFC2460

```
5. Packet Size Issues

   IPv6 requires that every link in the internet have an MTU of 1280
   octets or greater.  On any link that cannot convey a 1280-octet
   packet in one piece, link-specific fragmentation and reassembly must
   be provided at a layer below IPv6.
```

# What did IPv6 do differently?

IPv6 defined a min… of 1,28…

i've been told that 1280 is 1024 + 256!

Which seems like a pretty specious line of reasoning to me!

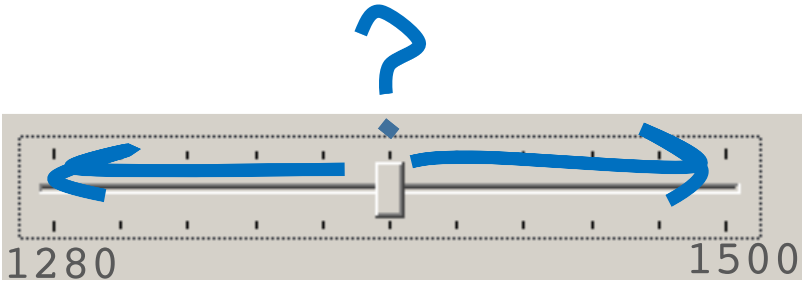
it represented a compromise about non-reassembling tunneling and has resulted in a fractured iPv6 network.

…reassembly must

1500 would've been a more robust outcome in my view.

# Bewteen 1280 and 1500

What should an IPv6 host use as a local MTU value?




- If you set it at 1280 then you invite fragmentation if you need to send larger packets, which will risk EH loss on fragmented packets
- If you set it at 1500 then you may encounter risks with MTU mismatch and PTB notification loss when talking with a host with a smaller MTU and encounter MTU Black Holes

# Lets look

- So if the issue is the combination of IPv6, UDP and larger packets then perhaps we can experiment with this
  - It’s called “the DNS” !
- So we set up an experiment…
  - Response 1 : 131 octets
  - Response 2: 1400 octets
  - Response 3: 1700 octets

And set up a name server reachable only on IPv6 and only on UDP

# What we expect to see

| Size | Fetched | Failed | Reason |
|---|---|---|---|
| Small (150 octets) | 99% | 1% | Noise |
| 1160 octets | 99% | 1% | Noise |
| 1400 octets | ? | ? | PTB |
| 1700 octets | <52% | >48% | EH Loss, Frag loss, PTB |

# What we saw

| | Tested | Always Fetched | Both | Always Missed |
|---|---|---|---|---|
| 150 | 11,719 | 8,792 (75.02%) | 377 (3.22%) | 2,550 (21.76%) |
| 1,160 | 2,004 | 1,353 (67.51%) | 5 (0.25%) | 646 (32.24%) |
| 1280 | | | | |
| 1,400 | 9,789 | 7,374 (75.33%) | 385 (3.93%) | 2,030 (20.74%) |
| 1,425 | 1,977 | 1,313 (66.41%) | 7 (0.35%) | 657 (33.23%) |
| 1,453 | 1,987 | 1,298 (65.32%) | 9 (0.45%) | 680 (34.22%) |
| 1500 | | | | |
| 1,700 | 11,170 | 5,859 (52.45%) | 172 (1.54%) | 5,139 (46.01%) |

# What we saw

| | Tested | Always Fetched | Both | Always Missed |
|---|---|---|---|---|
| 150 | 11,719 | 8,792 (75.02%) | 377 (3.22%) | 2,550 (21.76%) ?? |
| 1,160 | 2,004 | 1,353 (67.51%) | 5 (0.25%) | 646 (32.24%) |
| 1280 | | | | |
| 1,400 | 9,789 | 7,374 (75.33%) | 385 (3.93%) | 2,030 (20.74%) |
| 1,425 | 1,977 | 1,313 (66.41%) | 7 (0.35%) | 657 (33.23%) |
| 1,453 | 1,987 | 1,298 (65.32%) | 9 (0.45%) | 680 (34.22%) |
| 1500 | | | | |
| 1,700 | 11,170 | 5,859 (52.45%) | 172 (1.54%) | 5,139 (46.01%) |

There is quite some noise in this data – the small-size response shows a 21% loss rate, which is likely to be due to a combination of:

- DNS multi-slave query engine farms
- IPv6 Link Layer address manipulation
- ICMPv6 Address unreachable
- DNS timeouts

# Unreachables

- Dual Stack configurations hide a multitude of sins
- And one of these is the use of unreachable IPv6 addresses for DNS resolvers
  - 11,077 distinct unreachable IPv6 addresses !
  - Out of 22,000 distinct IPv6 /128 addresses
    - Which is not quite as bad as it looks – a number of resolvers are “aggressive” in their use of /64 interface identifiers

# Filtering the results

- Join individual resolver /128 addresses into common /64's
- Only look at resolver /64's that fetch either of the two low-size controls
  - Which means that the IPv6 address is reachable
  - And the resolver will successfully resolve a glueless delegation

# What we saw:

| Size | Tested | Always Fetched | Both | Always Missed |
|---|---|---|---|---|
| 150 | 5,433 | 5,290 (97%) | 143 (3%) | 0 |
| 1,160 | 654 | 651 (99%) | 3 (1%) | 0 |

1280

| | | | | |
|---|---|---|---|---|
| 1400 | 4,658 | 4,495 (96%) | 133 (3%) | 30 (1%) |
| 1425 | 636 | 619 (97%) | 5 (1%) | 12 (2%) |
| 1453 | 638 | 609 (95%) | 6 (1%) | 23 (4%) |

1500

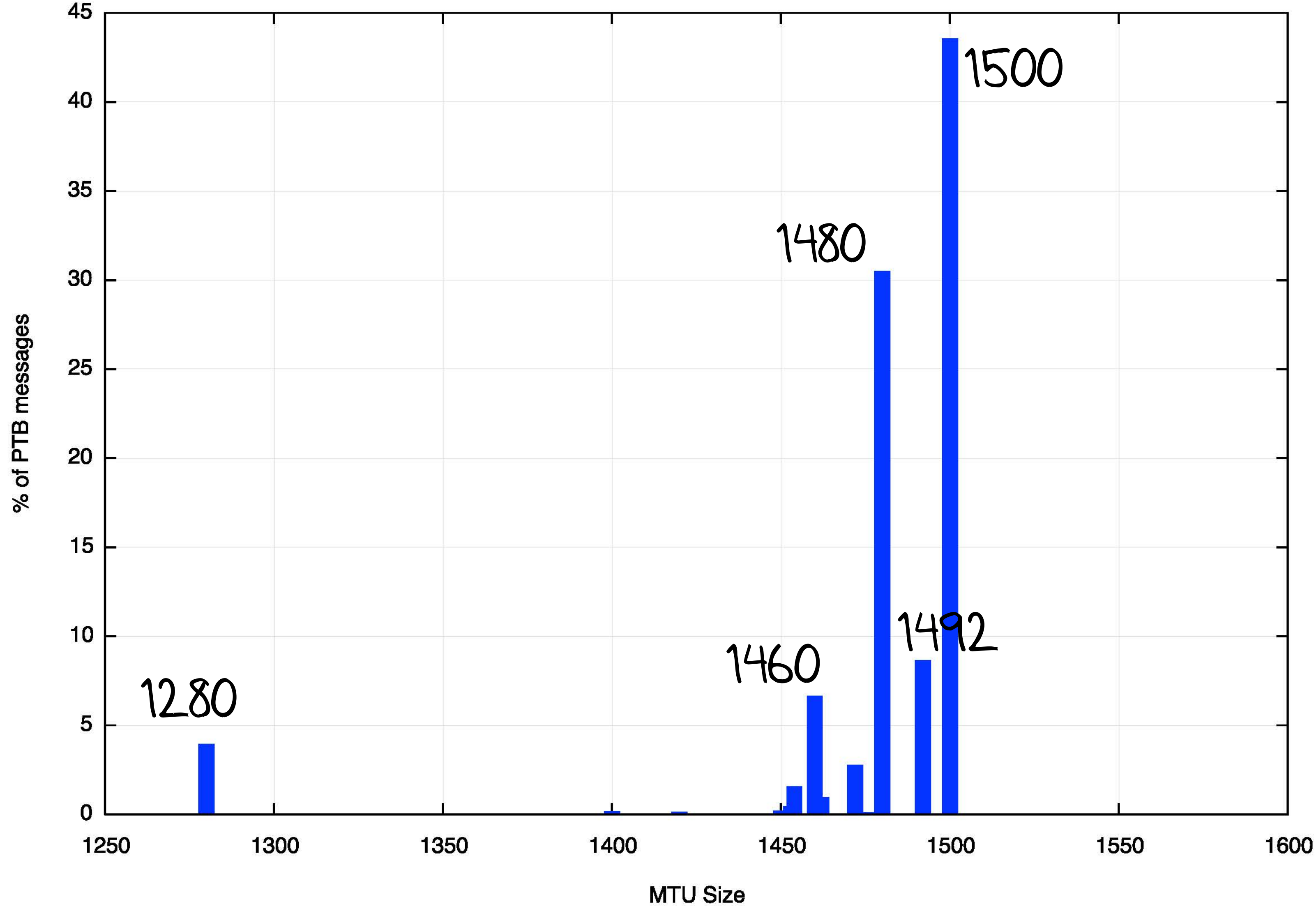
| | | | | |
|---|---|---|---|---|
| 1700 | 4,686 | 3,464 (74%) | 79 (1%) | 1,143 (25%) |

# What we saw:

| Size | Tested | Always Fetched | Both | Always Missed |
|---|---|---|---|---|
| 150 | 5,433 | 5,290 (97%) | 143 (3%) | 0 |
| 1,160 | 654 | 651 (99%) | 3 (1%) | 0 |
| 1280 → | | | | |
| 1400 | 4,658 | 4,495 (96%) | 133 (3%) | 30 (1%) |
| 1425 | 636 | 619 (97%) | 5 (1%) | 12 (2%) |
| 1453 | 638 | 609 (95%) | 6 (1%) | 23 (4%) |
| 1500 → | | | | |
| 1700 | 4,686 | 3,464 (74%) | 79 (1%) | 1,143 (25%) |

Between 1280 and 1500 the failure rate rises as the packet size rises.

# PTB MTU size distribution



# What we saw:

| Size | Tested | Always Fetched | Both | Always Missed |
|---|---|---|---|---|
| 150 | 5,433 | 5,290 (97%) | 143 (3%) | 0 |
| 1,160 | 654 | 651 (99%) | 3 (1%) | 0 |

1280 →

| Size | Tested | Always Fetched | Both | Always Missed |
|---|---|---|---|---|
| 1400 | 4,658 | 4,495 (96%) | 133 (3%) | 30 (1%) |
| 1425 | 636 | 619 (97%) | 5 (1%) | 12 (2%) |
| 1453 | 638 | 609 (95%) | 6 (1%) | 23 (4%) |

1500 →

| Size | Tested | Always Fetched | Both | Always Missed |
|---|---|---|---|---|
| 1700 | 4,686 | 3,464 (74%) | 79 (1%) | 1,143 (25%) |

There is a visible signal here for packets > 1500 octets.

it is not a 48% drop rate, but it is certainly more than 20% over and above the other packet sizes. There is a definite problem here with large iPv6 packets.

# EH drop? Or something more mundane?

**1,143** IPv6 /64s consistently cannot fetch a 1,700 octet UDP response

- **331** /64's generated ICMP Fragmentation reassembly ICMP messages
  - Firewall front end discarding trailing fragments
- **61** /64's generated Packet Too Big messages
- **751** failing /64's generated no ICMP messages

  i.e. EH packet drop was a maximum of 16% in this experiment

# What we saw with a 1280 MTU:

| Size | Tested | Always Fetched | Both | Always Missed |
|---|---|---|---|---|
| 150 | 4,777 | 4,600 (96%) | 177 (4%) | 0 |
| 1280 | | | | |
| 1400 | 4,662 | 3,695 (79%) | 80 (2%) | 887 (19%) |
| 1500 | | | | |
| 1700 | 4,635 | 3,429 (74%) | 95 (2%) | 1,111 (24%) |

Dropping the local MTU pushes a further 18% fragmentation drop into the 1,400 Byte packet

# What are we seeing?

Whether its EH drop of frag filtering, there is something deeply concerning in these numbers:

- A protocol that suffers a ~20% packet drop rate on fragmented packets presents a problem!
- Hosts should use the largest locally supported MTU for UDP (and use a 1,220 MSS for TCP)
- Applications should assume that large IPv6 fragmented packets may silently die in transit. They should be prepared to perform a rapid cutover to TCP in the event of suspected packet loss in UDP

- Should we revive **draft-bonica-6man-frag-deprecate**?

# Thanks!