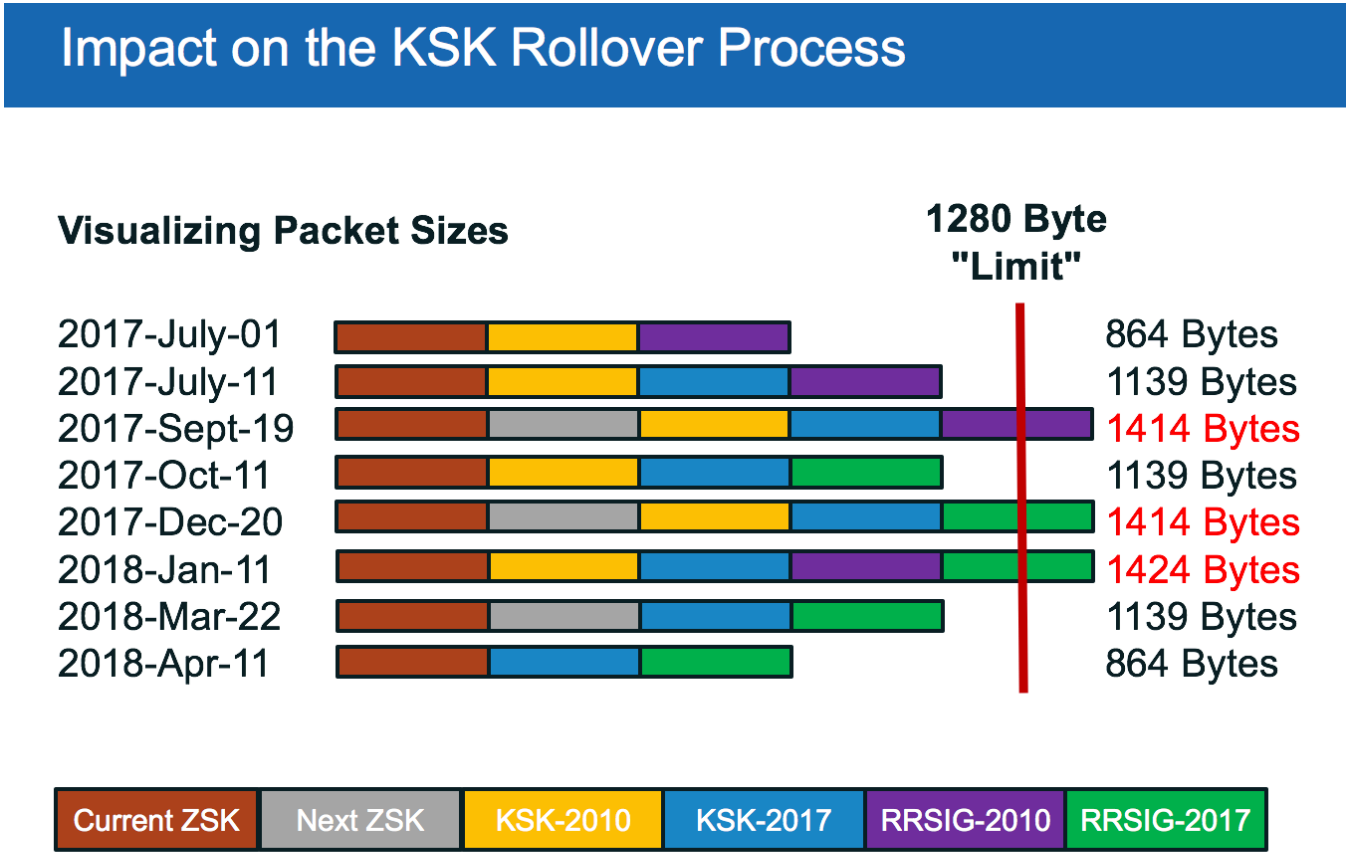
Scoring the DNS Root Server System

Geoff Huston

APNIC

The KSK Roll

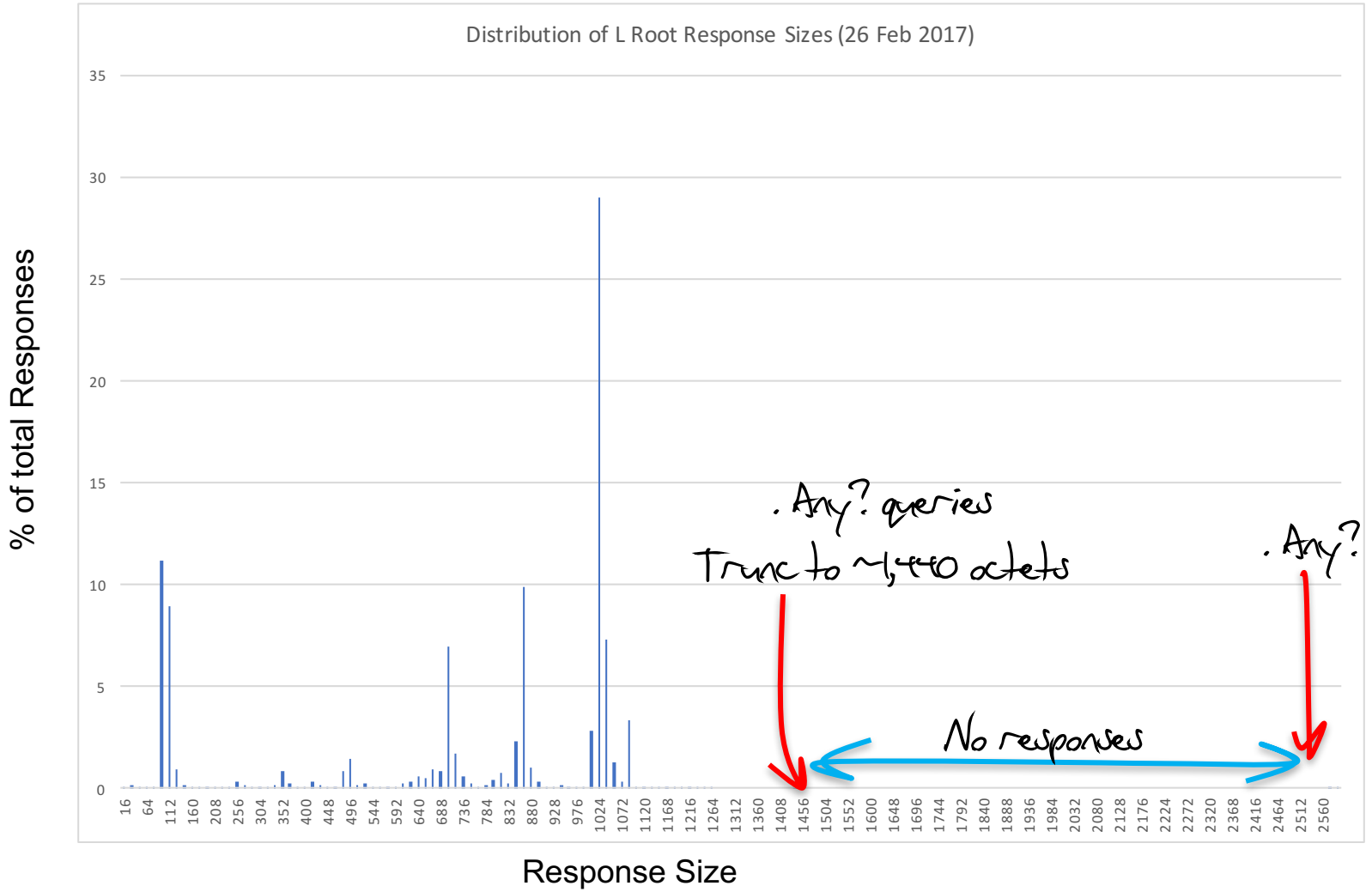
The current KSK roll at the root of the DNS includes three periods when the signed response to a query for the DNSKEY RR will exceed 1280 octets



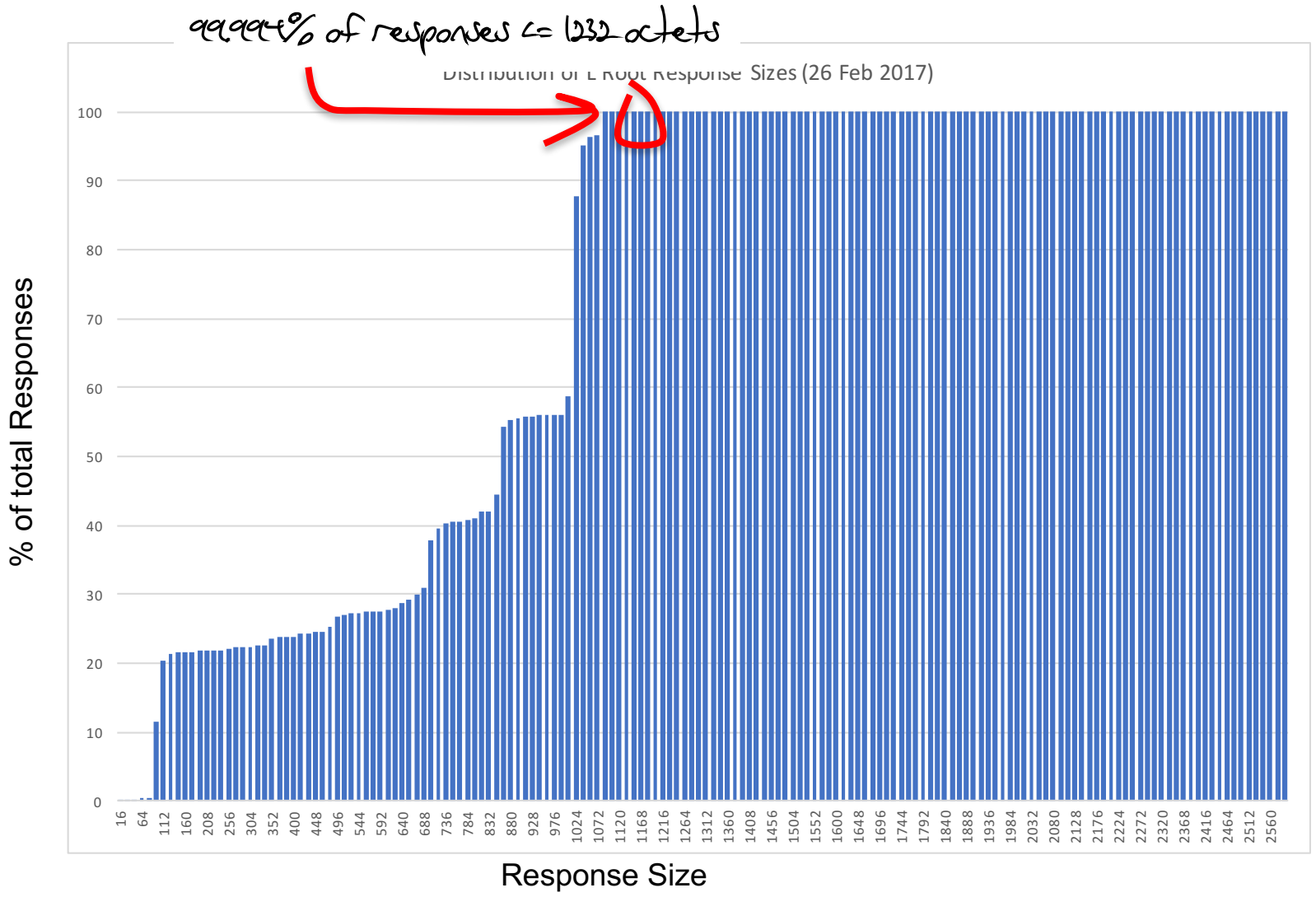
ICANN KSK Roll presentation – 17 March 2017

Root Zone Response Sizes

Root Servers do not normally pass back large responses



Root Zone Response Sizes



Root Zone "large" responses

- “Large” responses from root servers are currently very uncommon (and they are probably not critical to a resolver’s operation)
- The KSK roll will exercise this aspect of DNS behaviour in order to revoke the old KSK
- Two questions:
  - What is an ’ideal’ way for a server to pass back a large response?
  - How do root servers pass back large DNS responses today?

"Ideal" Large Packet Response Behaviours

IPv4 and UDP

- The defacto MTU for the IPv4 Internet is 1,500 octets
- Larger responses in UDP typically require IP fragmentation
- Firewalls don't like fragments
  - Some discard all trailing frags
  - Some reassemble the packet and make an admission decision based on the re-assembled packet and then refragment the payload
- UDP should not fragment the payload until the payload size reaches 1,472 octets
- At this point for the DNS it's a design trade-off between potential loss of trailing fragments or deliberately pushing the client to TCP by truncating the DNS response

IPv4 and TCP

- Another approach is to truncate the response at a maximum DNS payload of 1472 octets to avoid IP fragmentation, so that the resolver will re-query over TCP
- BUT not every DNS resolver supports TCP
  - Some 17% of resolvers, serving some 6% of end users, do not perform a TCP query following a truncated UDP response *

- TCP MSS should be initially offered as the interface size, less the header overheads of IP (20 octets) and TCP (20 octets)

  1,460 is a good MSS value to offer when using IPv4

* “A Question of Protocol” http://www.potaroo.net/ispcol/2013-09/dnstcp.html

Robust IPv4 Behaviour

- Deliver a IPv4 UDP payload up to 1,472 octets without fragmentation at source
- Do not truncate IPv4 UDP packets with a payload up to 1,472 octets *
- Offer a IPv4 TCP MSS of 1,460 octets

* Assuming that the offered EDNS(0) UDP buffer size permits this

IPv6 and UDP

There is no clear MTU for all IPv6

- 6to4 uses 1,480 octets
- Teredo uses 1,472 octets
- “native IPv6” uses 1,500 octets
- The minimum assured unfragmented IPv6 packet size is 1,280 octets, so many systems interpret that as a “safe” MTU for IPv6

IPv6 and Path MSS

- IPv6 does not permit routers to fragment packets
- It is possible for a sender to generate a packet that is too big for the path
- When a router cannot forward a packet because it is too large for the next hop, it generate an ICMPv6 Packet Too Big diagnostic message that is sent back to the source
  - The source can only “correct” the problem if it receives this ICMPv6 packet

The IPv6 Problem Space

- Some Firewall configurations drop trailing fragments
- Some Firewall configurations drop all ICMPv6 packets
- In an anycast situation some path elements may see a different anycast instance – i.e. an ICMPv6 PTB sent to the "source" may not get to the right source
- Fragments require Extension Headers and many switching elements in the network drop IPv6 packets will Extension Headers
- Many resolvers cannot query using TCP – either the resolvers are not configured to do so, or local firewall filter rules block TCP port 53 packets

IPv6 and ICMPv6 PTB

What should the server do when it receives an ICMPv6 PTB message?

- In TCP:
  - the sender can (should?) adjust its MSS to avoid sending fragments within ths session
  - It should retransmit the TCP segment
  - It should cache the new value and use this as the new MSS in any following TCP sessions with this resolver
- In UDP:
  - It's unclear what it should do in the case of UDP
  - A conservative approach would be to push the new MSS into a forwarding table, and hold it in this cache for some system-defined time
    - this system response needs to be carefully managed as uncontrolled expansion of the forwarding table with host entries represents a host vulnerability
  - An alternative approach would be to ignore the PTB and rely on the client to adjust its queries via the EDNS(0) Buffer Size to elicit a truncated response from the server – but this would only be applicable to the DNS

IPv6 and UDP

- Despite the 1,280 assumption, it's probably "safer" these days to use a 1500 MTU size for UDP
  - The problems here with UDP fragmentation are the loss of IPv6 packets that have the Fragmentation Extension Header
    - We observed an increase from 1% to 30% IPv6 packet drop rate when passing a 1,400 octet response when we dropped the server's MTU from 1,500 to 1,280*
  - Also there is the firewall filter issue and the loss of trailing fragments
  - ICMPv6 PTB messages are often filtered or lost
  - The use of a per-host MTU table is a potential vulnerability
  - What does a front end IPv6 UDP traffic balancer do with an incoming ICMPv6 PTB message and a farm of back-end UDP engines?

- The tradeoff is again between fragmented UDP loss rate and the issues with pushing the client to use TCP

* http://www.potaroo.net/ispcol/2016-10/dnsipv6.html

IPv6 and TCP

- Again, not every resolver supports TCP
- There are path MTU issues with TCP in some instances, and the issues of ICMPv6 PTB message filtering and TCP "black holes"
- A highly conservative IPv6 TCP MSS would be 1,220 octets
  - This will minimize the risk of Path MTU 'blackholing' and fragmentation loss
  - As DNS is not a very high volume application, the marginally lower carriage efficiency of a low MSS is offset by the higher probability of TCP robustness

About Truncation

If a client offers a EDNS(0) UDP Buffer size, should the server honor it?

For example:

If the client offers a 4,096 UDP buffer size, and the response is 2,500 octets in size, is it more robust for the server to send UDP fragments of the 2,500 octet response, or just truncate immediately?

The larger UDP response is in some sense more “correct”, *assuming the honest intentions of all query agents*

Sending large DNS responses over UDP is a risk for DDOS reflection attacks

Robust IPv6 Behaviour

- Deliver a IPv6 UDP payload up to 1,452 octets without fragmentation and without truncation *
- Do not truncate IPv6 UDP packets with a payload up to 1,452 octets *
- Respond to IPv6 ICMP PTB messages with a smaller packet size consistent with the MTU in the PTB message in UDP and TCP
- Offer a IPv6 TCP MSS of 1,220 octets

* Assuming that the offered EDNS(0) UDP buffer size permits this

Testing Root Servers

Construct a conventional query that generates a large response

- We use a query for a DS record of a large non-existent name, using EDNS(0) with DNSSEC OK and a UDP EDNS(0) buffer size of 4096 octets
- The DNS response is 1,268 octets in size
  - 1296 octet IPv4 packet
  - 1316 octet IPv6 packet

Query Tests

- Query using UDP in IPv4, and respond to a truncated response by re-querying with TCP
- Query using TCP in IPv4 using a local MSS of 1,460 octets

- Query using UDP in IPv6, and respond to a truncated response by re-querying with TCP
- Query using TCP in IPv6 using a local MSS of 1,440 octets

Let's add another test

dig . ANY @root

The response is normally 2,587 octets

Except:

- If you ask E, H or K over UDP (2,015 octet response)
- Or if you ask some instances of F (2,615 octet response)
- Or instances of F served by Cloudflare, where the response is NOTIMPL *
- Or L, where if you use V6 the response is 2,605 octets, and it's 2,105 octets using UDP on V4 and 2,587 octets using TCP on V4

* As seen in April 2017

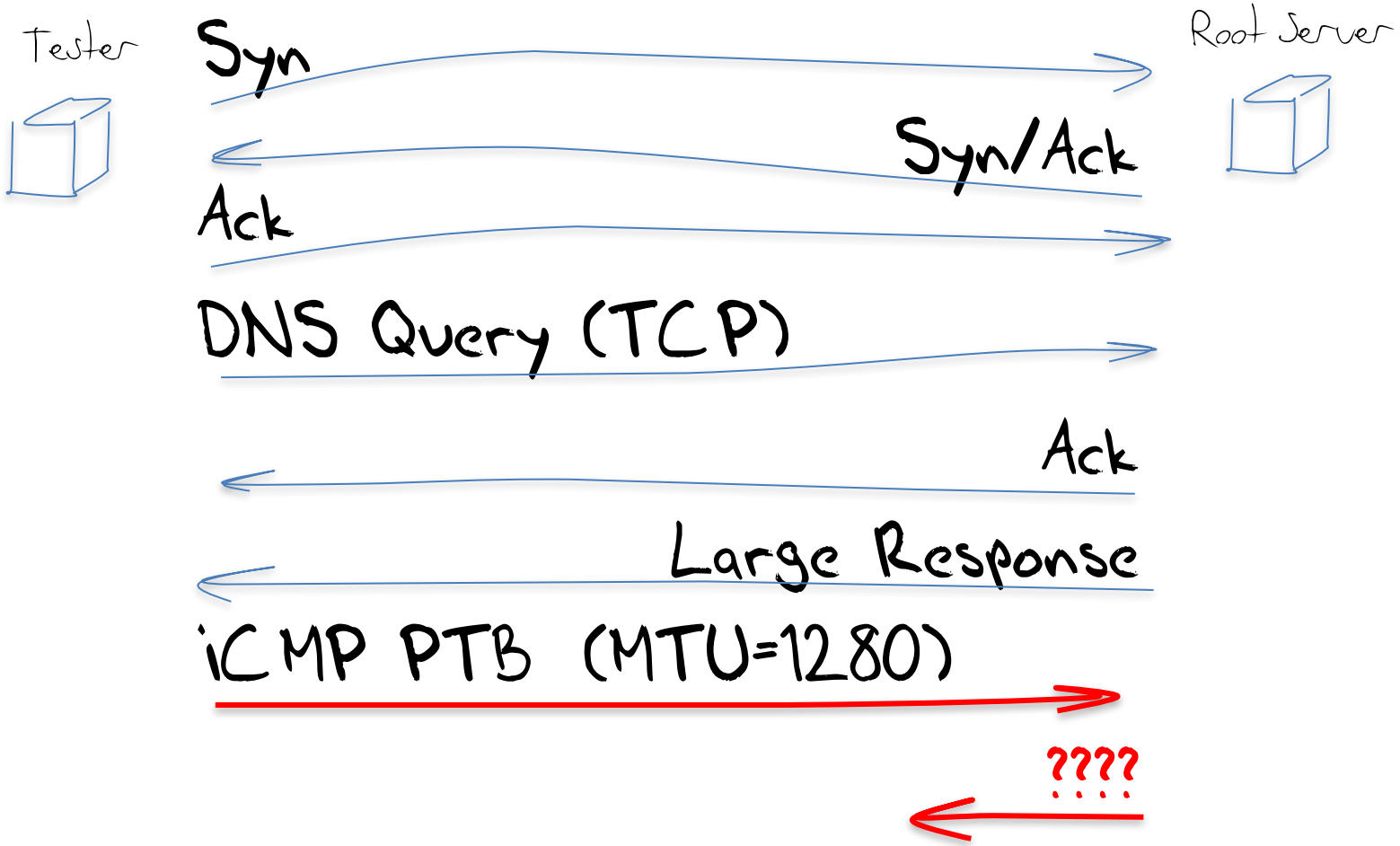
IPv6 UDP ICMPv6 PTB test

- Set up a steady sequence of queries to each root server at 10 second intervals
- Synthesize ICMPv6 PTB messages in response to received UDP responses greater than 1,280 octets in size
- Turn these PTB messages on for 2 minutes every hour
- Run the job for 5 hours

IPv6 TCP ICMPv6 PTB test

- Set up a raw socket for outgoing packets, and use the pcap library to capture incoming packets
- Delve deep into operating system stupidities to turn off gratuitous RSTs and TCP session firewalls
- Turn off totally unhelpful interface TCP segment management
- Trap the large TCP response and fake an ICMP response

IPv6 TCP ICMPv6 PTB test



Anycast

- These tests were conducted from Brisbane, Canberra (A&R network), Dallas, Frankfurt and Singapore
- It may be that there are differences across the entire anycast constellation for each root letter, but no material differences were observed in the tested subset
- There are major discrepancies between V4 and V6 in some cases, but that's a different story about the issues of anycast and Dual Stack!

The Scoresheets

 Operates in a robust manner with large responses

 Poor choice in operational behaviour

 Not relevant for this behaviour profile

A and J

These servers always truncate IPv6 UDP responses such that the UDP response is always under 1280 octets in size. This way they avoid fragmentation and UDP PMTU issues, but may encounter TCP issues instead. They use a large IPv6 TCP MSS. A responds to TCP ICMPv6 PTB messages, while J does not

- Delivers a IPv4 UDP payload up to 1,472 octets without fragmentation
- Does not truncate IPv4 UDP packets with a payload up to 1,472 octets
- Offers a IPv4 TCP MSS of 1,460 octets

- Delivers a IPv6 UDP payload up to 1,452 octets without fragmentation
- Truncates IPv6 UDP packets at 1,280 octets
- A responds to IPv6 ICMP PTB messages, J does not
- Offers a IPv6 TCP MSS of 1,440 octets

B and G

These servers always truncate IPv4 and IPv6 UDP responses such that the UDP response is always under 1280 octets in size. This way they avoid fragmentation and UDP PMTU issues, but may encounter TCP issues instead. They use a large IPv6 TCP MSS and do not respond to ICMPv6 PTB messages in TCP

- Delivers a IPv4 UDP payload up to 1,472 octets without fragmentation
- Truncates IPv4 UDP packets at 1,280 octets
- Offers a IPv4 TCP MSS of 1,460 octets

- Delivers a IPv6 UDP payload up to 1,452 octets without fragmentation
- Truncates IPv6 UDP packets at 1,280 octets
- Responds to IPv6 ICMP PTB messages
- Offers a IPv6 TCP MSS of 1,440 octets (1,220 would be more robust)

C, D, E, I and L

These servers appear to operate with a 1,500 octet MTU for both protocols in both UDP and TCP. They react to ICMPv6 PTB by clamping a 1,280 octet MTU against a host route for 10 minutes. They use a large IPv6 TCP MSS.

- Delivers a IPv4 UDP payload up to 1,472 octets without fragmentation
- Does not truncate IPv4 UDP packets with a payload up to 1,472 octets
- Offers a IPv4 TCP MSS of 1,460 octets

- Delivers a IPv6 UDP packet > 1,280 in size without fragmentation
- Does not truncate IPv6 UDP packets with a payload up to 1,452 octets
- Respond to UDP ICMPv6 PTB. D and I respond to TCP ICMPv6 PTB
- Offers a IPv6 TCP MSS of 1,440 octets (1,220 would be more robust)

F

These servers appear to operate with a 1,500 octet MTU for IPv4 in both UDP and TCP. They appear to use a local 1,280 octet MTU setting for UDP in IPv6, fragmenting the outbound packet to fit within this MTU. There are subtle differences between ‘original’ F root instances and those operated by Cloudflare

- Delivers a IPv4 UDP payload up to 1,472 octets without fragmentation
- Does not truncate IPv4 UDP packets with a payload up to 1,472 octets
- Offers a IPv4 TCP MSS of 1,460 octets

- Fragments IPv6 UDP packets at 1,280 octets
- Does not truncate IPv6 UDP packets with a size > 1,280
- Responds to IPv6 ICMP PTB messages
- Cloudflare instances offer TCP MSS of 1,220 octets, yet pass 1,271 octet segments (?)

F instances offer a TCP MSS of 1,440 octets, but fragment at 1,232 octet TCP segments

M

These servers appear to operate with a 1,500 octet MTU for IPv4 in both UDP and TCP. They appear to use a local 1,280 octet MTU setting for UDP in IPv6, fragmenting the outbound packet to fit within this MTU. They use a large IPv6 TCP MSS.

- Delivers a IPv4 UDP payload up to 1,472 octets without fragmentation
- Does not truncate IPv4 UDP packets with a payload up to 1,472 octets
- Offers a IPv4 TCP MSS of 1,460 octets

- Fragments IPv6 UDP packets at 1,280 octets
- Does not truncate IPv6 UDP packets with a size > 1,280
- Responds to IPv6 ICMP PTB messages
- Offers a IPv6 TCP MSS of 1,440 octets, yet fragments its TCP response at 1,220 octets

H and K

These servers appear to operate with a 1,500 octet MTU for both protocols in both UDP and TCP. They do not react to ICMPv6 PTB messages. H uses a large IPv6 TCP MSS

Delivers a IPv4 UDP payload up to 1,472 octets without fragmentation

Does not truncate IPv4 UDP packets with a payload up to 1,472 octets

Offers a IPv4 TCP MSS of 1,460 octets

Delivers a IPv6 UDP payload up to 1,452 octets without fragmentation

Does not truncate IPv6 UDP packets with a payload up to 1,452 octets

H responds to TCP ICMPv6 PTB, but not UDP. K does not respond in UDP

H offers a IPv6 TCP MSS of 1,440 octets. K offers 1,220

Results

Root	IPv4			IPv6			ICMPv6 PTB	
	Truncate	Fragment	TCP MSS	Truncate	Fragment	TCP MSS	UDP	TCP
A	1,500		1,460	1,280		1,440		Y
B	1,280		1,460	1,280		1,440		N
C		1,500	1,460		1,500/1,280 *	1,440	Y	N
D	1,500		1,460	1,500		1,440	Y	Y
E		1,500	1,460		1,500/1,280 *	1,440	Y	N
F		1,500	1,460		1,280	1,440		**
G	1,280		1,460	1,280		1,440		N
H		1,500	1,460		1,500/1,280 *	1,440	N	Y
I		1,500	1,460		1,280	1,220	Y	
J	?		1,460	1,280		1,440		N
K		1,500	1,460		1,500/1,280 *	1,220	N	
L		1,500	1,460		1,500	1,440	Y	N
M		1,500	1,460		1,280	1,440		**

* 1,500/1,280 - these servers will send up to 1,500 octet responses, but will fragment at the 1,280 octet point

** These servers fragmented the TCP segments at 1,280 octets

Results

Root	IPv4			IPv6			ICMPv6 PTB	
	Truncate	Fragment	TCP MSS	Truncate	Fragment	TCP MSS	UDP	TCP
A	1,500		1,460	1,280		1,440		Y
B	1,280		1,460	1,280		1,440		N
C		1,500	1,460		1,500/1,280 *	1,440	Y	N
D	1,500		1,460	1,500		1,440	Y	Y
E		1,500	1,460		1,500/1,280 *	1,440	Y	N
F		1,500	1,460		1,280	1,440		**
G	1,280		1,460	1,280		1,440		N
H		1,500	1,460		1,500/1,280 *	1,440	N	Y
I		1,500	1,460		1,280	1,220	Y	
J	1,200		1,460	1,280		1,440		N
K		1,500	1,460		1,500/1,280 *	1,220	N	
L		1,500	1,460		1,500	1,440	Y	N
M		1,500	1,460		1,280	1,440		**

These three servers cannot provide a large response in IPv6 over a constrained path

* 1,500/1,280 - these servers will send up to 1,500 octet responses, but will fragment at the 1,280 octet point

** These servers fragmented the TCP segments at 1,280 octets

The Star Table

	IPv4			IPv6			ICMPv6 PTB	
Root	Truncate	Fragment	TCP MSS	Truncate	Fragment	TCP MSS	UDP	TCP
A	★	★	★	★	☆	★	☆	★
B	★	☆	★	★	☆	★	☆	★
C	★	★	★	★	★	★	★	★
D	★	★	★	★	★	★	★	★
E	★	★	★	★	★	★	★	★
F	★	★	★	☆	★	★	☆	☆
G	★	☆	★	★	☆	★	☆	★
H	★	★	★	★	★	★	☆	★
I	★	★	★	☆	★	★	★	★
J	★	★	★	★	☆	★	☆	★
K	★	★	★	★	★	★	★	☆
L	★	★	★	★	★	★	★	★
M	★	★	★	☆	★	★	☆	☆

Engineering Trade-Offs

- There is no single “correct” way to handle large DNS responses
- Each root server has made its own design decision about truncation vs fragmentation in IPv6 for large responses
- As long as there is a dual stack DNS, and as long as this variation in behaviour still exists, then a persistent dual stack resolver will get the answer they are seeking from one of the DNS root servers (eventually!)
  - Assuming, of course, that they are able to pose queries using both UDP and TCP, and in IPv4 and IPv6

Back to the KSK roll

The 1,414 / 1,425 octet response might require a resolver to try harder, using both IPv6 and IPv4 and a number of root server letters to get the root zone's DNSKEY RR

- If resolver cannot query over TCP/IPv4 then **B** and **G** cannot provide a complete response in UDP/IPv4
- If resolver cannot query over TCP/IPv6 then **A**, **B**, **G** and **J** cannot provide a complete response in UDP/IPv6
- If the resolver sits behind an MTU-constrained IPv6 Path, then **B**, **C**, **E**, **G**, **J** and **L** will not respond to an ICMPv6 PTB message in TCP/IPv6
- If the resolver sits behind a firewall that strips IPv6 Fragmented packets, then **F** and **M** will be unable to deliver a response in TCP/IPv6

Conclusion

The various root server behaviours are to some extent anomalous, but probably not fatal to DNSSEC-validating resolvers

As long as a DNSSEC-validating resolver is persistent, and is able to try a combination of protocols and Root Servers, then it will probably be able to retrieve the larger Root Zone DNSKEY RR

Questions?