

# Measuring ATR

Joao Damas, Geoff Huston

@APNIC Labs

March 2018

# September 2017:

## V6, the DNS and Fragmented UDP Responses

We used the Ad platform to enroll endpoints to attempt to resolve a DNS name that included a IPv6 fragmented UDP response when attempting to resolve the name server's name

Total number of tests: 10,851,323

Failure Rate in receiving a large response: 4,064,356

IPv6 Fragmentation Failure Rate: **38%**



# The Internet has a problem ...

- Instead of evolving to be more flexible and more capable, it appears that the Internet's transport is becoming more ossified and more inflexible in certain aspects
- One of the major issues here is the handling of large IP packets and IP level packet fragmentation
- We are seeing a number of end-to-end paths on the network that no longer support the carriage of fragmented IP datagrams
- We are concerned that this number might be getting larger, not smaller

# The Internet has a problem ...

## What are we doing about it?

- Fixing this in deployed network infrastructure is extremely hard, so there are efforts underway to modify host and application behaviour to avoid the problem
- One avoidance approach has been to clamp the MSS of sessions to a conservative value that has a very high probability of non-fragmented transmission
  - For example, QUIC uses a fixed 1350 octet MTU
  - IPv6 sets a minimum unfragmented MTU of 1280 octets within the IPv6 specification
- As long as applications and protocol stacks keep all packets under some conservative MTU setting, then the possibility of encountering IP fragmentation and network/host/firewall/middleware fragmentation packet loss is substantially reduced

# The Internet has a problem ...

## But what about the DNS?

- One application that is making increasing use of large UDP packets is the DNS
- This is generally associated with DNSSEC-signed responses (such as “dig +dnssec DNSKEY org”) but large DNS responses can be generated in other ways as well (such as “dig . ANY”)
- In the DNS we appear to be relying on the successful transmission of fragmented UDP packets, but at the same time we see an increasing problem with the coherence in network and host handling of fragmented IP packets, particularly in IPv6

# Changing the DNS

- One potential avoidance response is to shift all the DNS to TCP
  - But this is a last resort response – TCP has much higher overheads in terms of transaction times and server scaling
  - Our current intuition is that the Internet would prefer to continue to operate the DNS to run over UDP as much as possible

# Changing the DNS

- Or shift just all large DNS transactions to TCP
  - This is a reversion to the original DNS specification (pre-EDNS(0) UDP buffer size signalling), where only small (smaller than 512 octets) responses are passed across UDP.
  - Larger DNS responses (larger than 512 octets) are truncated and the truncation is intended to trigger the client to re-query using TCP
  - Again this seems like a heavy-handed solution that is not always required - IP fragmentation works more often than not

# Changing the DNS

- Or we could try a hybrid approach of using both fragmented and truncated responses at the same time

# What is “ATR”?

- It stands for “Additional Truncated Response”  
Internet draft: draft-song-atr-large-resp-00  
September 2017  
Linjian (Davey) Song, Beijing Internet Institute
- It’s a hybrid response to noted problems in IPv4 and IPv6 over handling of large UDP packets and IP fragmentation
- ATR adds an additional response packet to ‘trail’ a fragmented UDP response
- The additional response is just the original query with the Truncated bit set, and the sender delays this additional response packet by 10ms

[Docs] [txt|pdf|xml|html] [Tracker] [Email] [Nits]

Versions: 00

Internet Engineering Task Force L. Song  
Internet-Draft Beijing Internet Institute  
Intended status: Informational September 10, 2017  
Expires: March 14, 2018

**ATR: Additional Truncated Response for Large DNS Response**  
**draft-song-atr-large-resp-00**

## Abstract

As the increasing use of DNSSEC and IPv6, there are more public evidence and concerns on IPv6 fragmentation issues due to larger DNS payloads over IPv6. This memo introduces a simple improvement on authoritative server by replying additional truncated response just after the normal large response.

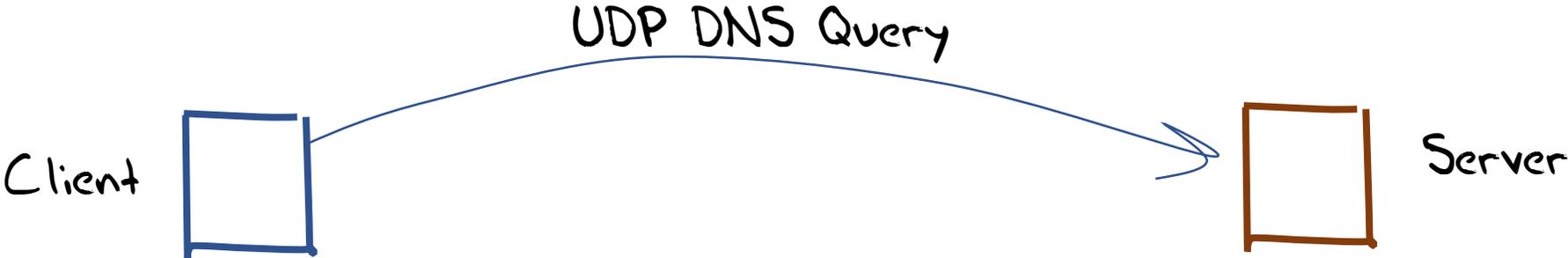
REMOVE BEFORE PUBLICATION: The source of the document with test script is currently placed at GitHub [[ATR-Github](#)]. Comments and pull request are welcome.

## Status of This Memo

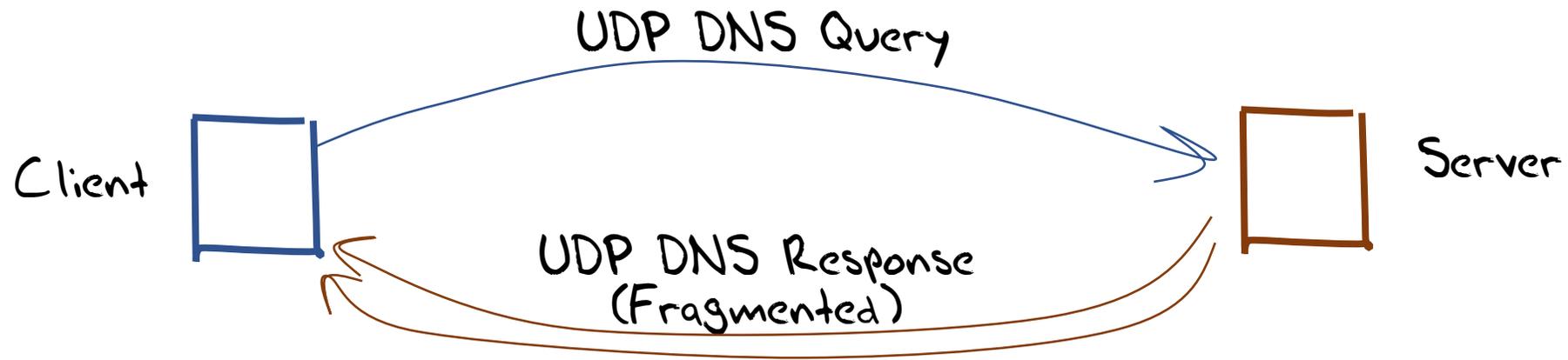
This Internet-Draft is submitted in full conformance with the provisions of [BCP 78](#) and [BCP 79](#).

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/>.

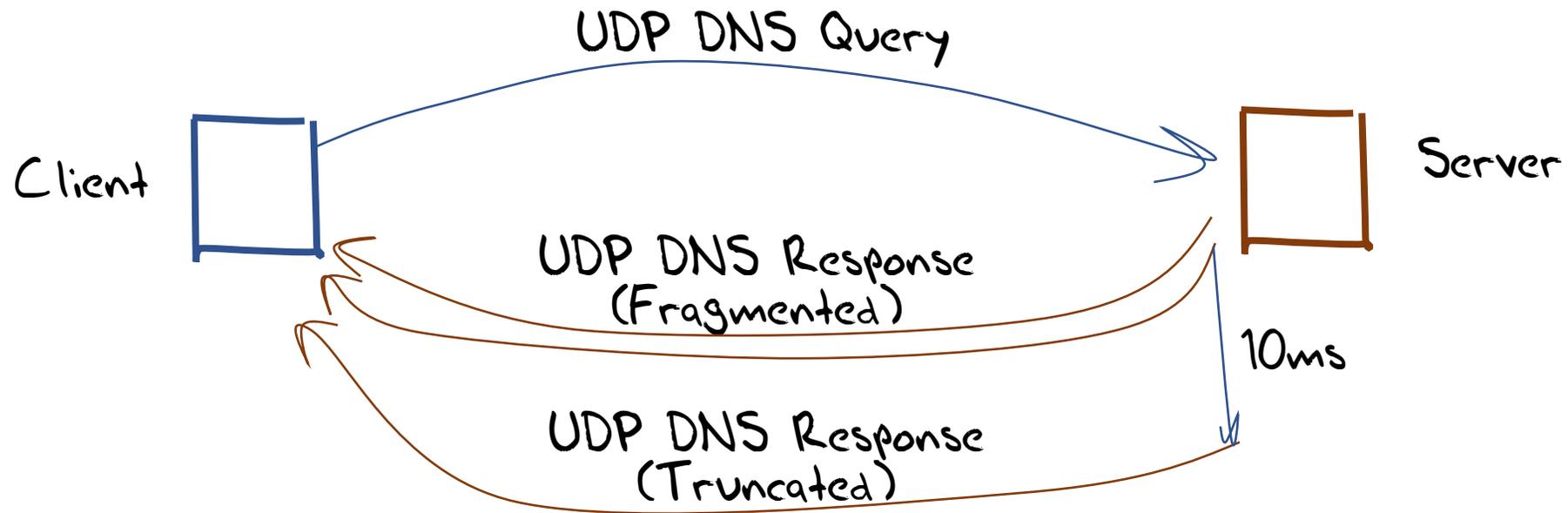
# ATR Operation



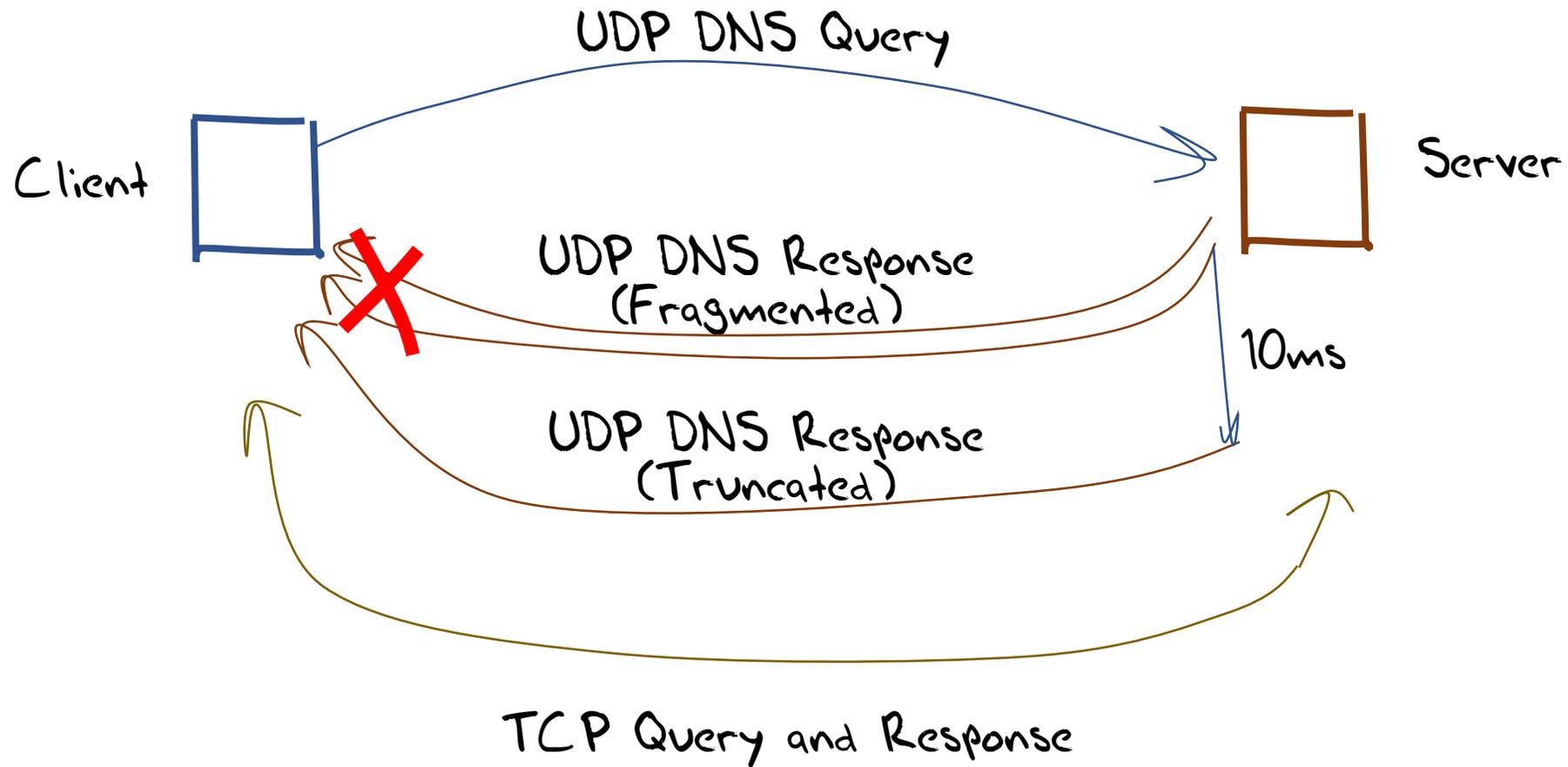
# ATR Operation



# ATR Operation



# ATR Operation



# The Intention of ATR

- When a UDP DNS response is fragmented by the server, then the server will also send a delayed truncated UDP DNS response
  - The delay is proposed to be 10ms
- If the DNS client receives and reassembles the fragmented UDP response the ensuing truncated response will be ignored
- If the fragmented response is lost due to fragmentation loss, then the client will receive the short truncated response
- The truncation setting is intended to trigger the client to re-query using TCP

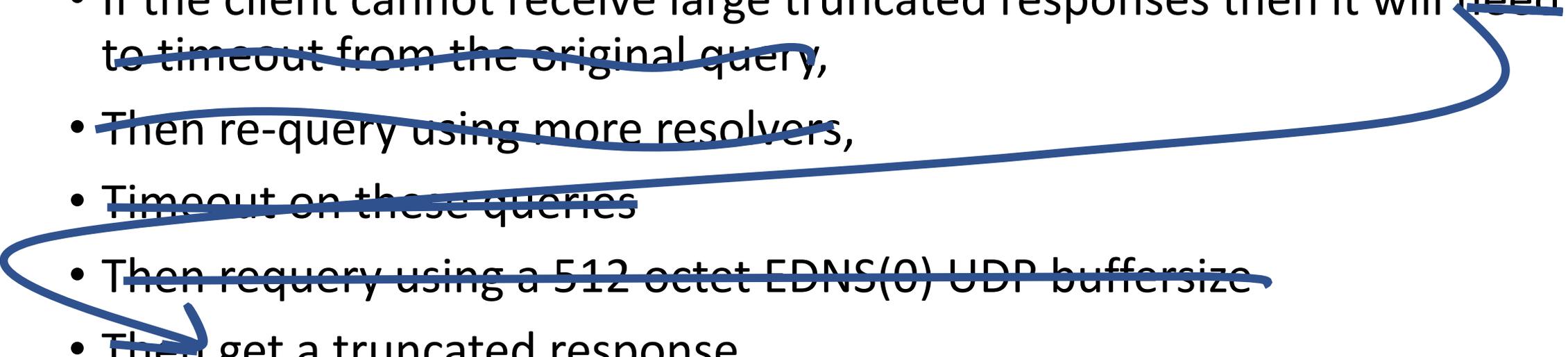
# The Intention of ATR

Today:

- If the client cannot receive large truncated responses then it will need to timeout from the original query,
- Then re-query using more resolvers,
- Timeout on these queries
- Then re-query using a 512 octet EDNS(0) UDP buffersize
- Then get a truncated response
- Then re-query using TCP

# The Intention of ATR

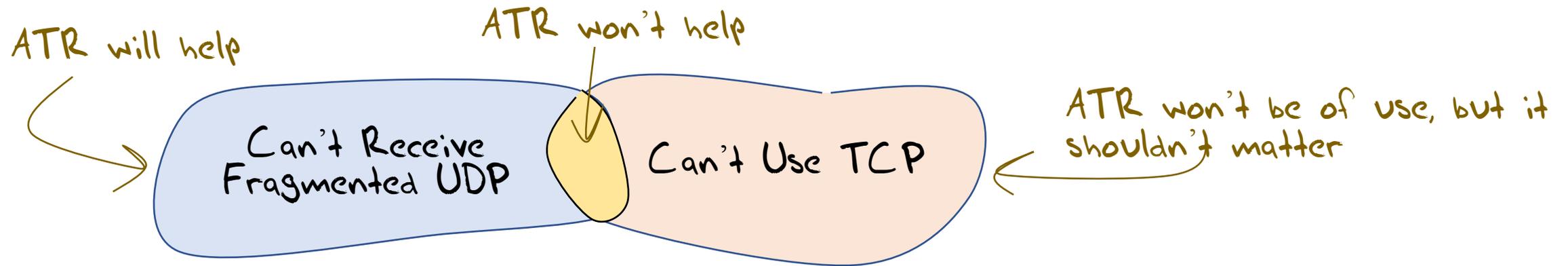
## ATR

- If the client cannot receive large truncated responses then it will need to timeout from the original query,
  - Then re-query using more resolvers,
  - Timeout on these queries
  - Then requery using a 512 octet EDNS(0) UDP buffersize
  - Then get a truncated response
  - Then requery using TCP
- 
- A hand-drawn blue line with arrows highlights the flow of the process. It starts at the end of the first bullet point, loops back to the start of the second, then continues to the start of the third, then to the start of the fourth, and finally to the start of the fifth. The line also underlines the words 'will need' in the first bullet point.

# What could possibly go wrong?

- Network level packet re-ordering may cause the shorter truncated response packet to overtake the fragmented response, causing an inflated TCP load, and the potential for TCP loss to be triggered
- Not every client DNS system supports using TCP to emit queries

# ATR and Resolver Behaviour



How big are each of these pools?  
What proportion of users are impacted?

# Is ATR worth the effort?

- Its unlikely that the resolution delay and resolution failure rate will be any larger using ATR than without it, and more likely that the both average resolution times and the overall DNS resolution failure rate will fall
- But will the gains offset the additional costs in generating this additional response?

# Experiment Details

- Use 6 tests:
  - 2 tests use ATR responses – one is DNS over IPv4, the other is DNS over IPv6
  - 2 tests use only truncated responses – IPv4 and IPv6
  - 2 tests use large fragmented UDP responses - IPv4 and IPv6
- Use the fetch of the web object as the measure of DNS resolution success/failure
- Experiment counts:
  - 48M experiments undertook the DNS over IPv4 tests
  - 23M experiments undertook the DNS over IPv6 tests (47%)

# A. Large Fragmented UDP response

DNS over UDP over IPv4 – 21% loss rate

DNS over UDP over IPv6 – 38% loss rate

The IPv4 loss rate is significant at 21% of all samples

- up to 4% could be attributed to experimental uncertainties on linkage of the DNS to web object retrieval
- which leaves **17%** as a minimum loss rate in IPv4 for large responses

IPv6 shows a very significant level of loss

- which is comparable to that observed in September 2017
- the minimum loss rate is **34%**, accounting for likely web object conversion loss rates

## B. Truncated UDP Response and TCP

DNS over IPv4 – 5% TCP loss rate

DNS over IPv6 – 9% loss rate

We are looking here at the proportion of experiments that react to the truncated response with a TCP session. The “loss” referred to here is a loss relating to the lack of a followup TCP session, and does NOT include any web retrieval loss factors

The IPv6 TCP failure rate slightly less than double the IPv4 failure rate

# C. ATR

- IPv4 DNS:
  - 72% successfully retrieved the object without using TCP
    - 38% did not
    - This is higher than the large packet drop rate probably due to packet reordering factors)
  - 16% used TCP
  - 12% did not use TCP and did not retrieve the web object
- IPv6 DNS:
  - 54% successfully retrieved the object without using TCP
    - 46% did not
    - This is again higher than the large packet drop rate probably due to packet reordering factors)
  - 28% used TCP
  - 18% did not use TCP and did not retrieve the web object

# Large DNS responses

- There is a major issue with large fragmented UDP responses in the DNS, and IPv6 is certainly far more of a problem in this respect than IPv4 - ATR would normally be expected to address this
  - The issue would appear to be zealous firewalls in IPv4 and a combination of firewalls and IPv6 extension header packet drop in IPv6
- However, there is also a significant TCP drop rate for the DNS
  - Here we can probably ascribe this to zealous firewalls, but there may also be UDP-only DNS clients out there as well
- Which means that large responses have a drop rate in the DNS, both in UDP and TCP, that cannot be ignored

# ATR and Large DNS Responses

## IPv4 Failure Rate

WITHOUT ATR: 22%

WITH ATR: 12%

## IPv6 Failure Rate

WITHOUT ATR: 38%

WITH ATR: 18%

This is a measurement of the user level impact, not a count of resolvers

# ATR Assessment

- Is this level of benefit worth the additional server and traffic load when sending large responses?
- Is this load smaller than resolver hunting in response to packet drop?
- Is the faster fallback to TCP for large responses a significant benefit?
- Is 10ms ATR delay too short? Would a longer gap reduce response reordering?
- Do we have any better ideas about how to cope with large responses in the DNS?