The ISP Column
A monthly column on things Internet

December 2020
Geoff Huston,
Joao Damas

DNS 2XL

The first part of this report on the handling of large DNS responses looked at the behaviour of the DNS, and the interaction between recursive resolvers and authoritative name servers in particular and examined what happens when the DNS response is around the Internet's de facto MTU size of 1,500 octets.

For responses larger than 1,500 octets we saw failure in some 2.5% of all cases. What we observed was two forms of DNS failure. The first was that resolvers were signalling to the server via query attributes that it was acceptable for the server to send large responses over fragmented UDP, but then were unable to reassemble the fragmented response, either due to local host or local network constraints. This scenario occurred in around three quarters of all such failure cases in our measurement tests. The second failure form was where the resolver had received a truncated DNS response and there was a subsequent TCP failure. This included the failure to open a TCP session or a TCP path MTU mismatch where the TCP session hung when attempting to pass back the DNS response. This occurred in slightly less than one quarter of all failure cases in our measurement tests. The measurement setup and the results from this work are to be found in part 1 of this report, "DNS XL" (https://www.potaroo.net/ispcol/2020-11/xldns.html).

However, we're not finished with these measurements. The results that are presented in the first part of this report are based on respecting the packet size constraints expressed in DNS queries. These constraints are that no UDP DNS response should exceed 512 octets unless there is an EDNS(0) extension with a UDP buffer size provided in the query, and the value of this buffer size field is greater than 512. When there is a UDP buffer size in the query, then the DNS response should be no larger than this size. In such cases where this is not possible, then the server will respond with a truncated DNS response over UDP. In this measurement the truncated response packet has an empty answer section, so the resolver making the query cannot use this truncated response to assemble an answer, and it should trigger the resolver to repeat the query over a TCP session with the server.

In this, the second part of the report, we ask the question: What if we break with these conventions?

In particular, we are interested in understanding the likely changes to DNS resolution behaviour of fragmented UDP responses, the behaviour of TCP responses and the behaviour of the DNS as a whole if all recursive resolvers were use the DNS Flag Day 2020 (https://dnsflagday.net/2020/) setting of 1,232 octets as a buffer size in their queries. Here, we will look at the behaviour of the DNS when we process incoming queries as if they all had an EDNS(0) extension and there was a buffer size in this extension that was set to a particular value. Yes, this server-based rewriting of queries is cheating, and it's not what resolvers may be expecting, but it allows us to gain some further insights into the capabilities of the resolver to authoritative part of the DNS.

We are going to perform five variants of changing DNS queries. We will firstly test UDP buffer sizes where all incoming queries are altered to have buffer sizes of 512, 1,232, 1,440 and 4,096 octets. We will then modify the MSS of incoming TCP SYN packets and set this value to 1,220 octets.

1. All UDP

When we force the buffer size to 4,096 octets for all incoming queries then at no stage will a recursive resolver receive a response with the truncation bit set. This means that the server will respond to all queries over UDP with a UDP response, and it will fragment all larger UDP responses. The fragmentation onset will reflect the server's local MTU setting of 1,500 octets. The results are shown in Table 1.

> Actually, that's not quite "all." In 1% of cases, we observe a query over TCP, even though a truncated response has not been previously sent. It appears that some of the time a resolver that is not receiving fragmented UDP responses will probe the server with TCP in some kind of liveness test.

DNS Response Size	Tests	UDP Pass Rate	UDP Fail Rate	IPv4 Failure Rate	IPv6 Failure Rate	Control Failure Rate
1150	1,140,192	99.6%	0.4%	0.6%	0.1%	
1190	1,138,792	99.6%	0.4%	0.6%	0.1%	
1230	1,273,730	99.6%	0.4%	0.6%	0.1%	0.5%
1270	1,272,765	98.1%	1.9%	2.4%	1.2%	0.5%
1310	1,275,436	98.2%	1.8%	2.4%	1.2%	0.5%
1350	1,272,634	98.2%	1.8%	2.4%	1.2%	0.5%
1390	1,273,332	98.1%	1.9%	2.4%	1.2%	0.5%
1430	1,274,189	97.8%	2.2%	2.6%	1.6%	0.5%
1470	1,274,581	96.9%	3.1%	3.7%	17.6%	1.0%
1510	1,273,496	85.0%	15.0%	14.2%	17.6%	2.4%
1550	1,274,776	85.0%	15.0%	14.4%	17.7%	2.6%
1590	1,276,441	85.1%	14.9%	14.4%	17.6%	2.6%
1630	1,275,233	85.1%	14.9%	14.5%	17.6%	2.6%

Table 1 – Failure Rate on UDP Test

The columns in Table 1 reflect a dual stack failure rate, an IPv4-only experiment, and an IPv6-only experiment, and the control, which is the experiment that does not alter the received buffer size in any way.

There are some unexpected outcomes in this data. The first is that we observed a 2% failure rate for unfragmented UDP responses with DNS payload sizes of 1,270 octets and greater. Oddly enough the failure rate for DNS payloads between 1,270 octets and 1,430 octets in IPv4-only (2.4%) is double that of IPv6-only (1.2%). These DNS responses are packaged by the server as unfragmented UDP packets.

As the smaller control unfragmented DNS response was successfully processed by the resolver, this presumably implies that there is some network infrastructure close to some resolvers that is discarding UDP packets where the payload size is between 1,270 and 1,430 octets, or the resolvers themselves are not accepting incoming DNS packets of size greater than 1232 octets in some circumstances.

> This particular result is likely to be due to the nature of the experimental setup and resolver behaviour, rather than being due to network behaviour.
>
> In this experiment we are deliberately abusing the DNS specification and the experiment's server is ignoring the resolver clients' offered UDP buffer size values.

Most resolver implementations appear not to raise an exception if the DNS response in the UDP packet is larger than the UDP buffer size specified in the query, but some resolver implementations appear perform a correlation between query and response. These implementations appear to be discarding a UDP response if the DNS payload is larger than the UDP buffer size in the original query. Similarly, there are instances where the response is being discarded if no buffer size was originally given by the resolver client and the response is larger than 512 octets.

When we look at the comparison between the resolver client's buffer size and the size of the UDP response, then for each individual test there are three possible types of response: all responses for the test are smaller than the query-specified buffer sizes, all responses for the test are larger than the query-specified buffer sizes, or it's a mixed scenario. We then divide up each case into success and fail. The results are as follows:

DNS Size	Success			Fail		
	Smaller	Larger	Mixed	Smaller	Larger	Mixed
1150	58.20%	37.70%	4.10%	0.04%	99.95%	0.02%
1190	58.62%	37.44%	3.95%	0.04%	99.94%	0.02%
1230	58.74%	37.19%	4.06%	0.04%	99.94%	0.02%
1270	59.55%	36.80%	3.65%	0.04%	99.94%	0.02%
1310	59.69%	36.37%	3.93%	0.04%	99.94%	0.02%
1350	59.44%	36.94%	3.62%	0.04%	99.94%	0.02%
1390	59.67%	36.40%	3.93%	0.04%	99.94%	0.02%
1430	59.67%	36.52%	3.81%	0.04%	99.94%	0.02%
1470	57.42%	38.23%	4.35%	0.04%	99.94%	0.02%

It's clear that in these unfragmented UDP cases the majority of failures occur when the DNS response is larger than the query-specified buffer size.

The conclusion drawn from this data is that the observed loss rates for unfragmented UDP responses when we use a test that deliberately disregards the offered UDP buffer size are generally attributable to these resolver clients rejecting the server's responses in those cases where the response is larger than the size specified in the original query. There is no evidence of systematic network failure when using these packet sizes, either in IPv4 or in IPv6.

When we quote figures about IPv6 we are talking about the pass and failure rates as they relate to the subset of users who are located behind IPv6-capable DNS resolvers. This is currently measured to be around some 55% of users (https://www.potaroo.net/ispcol/2020-07/dns6.html).

For UDP packets that are fragmented by the server before they are sent, namely with payloads greater than 1,472 octets (and 1,452 in IPv6) the failure rate rises considerably for both protocols. IPv6 fragmentation is evidently not handled as well as IPv4, but both protocols show an extremely high loss rate. There are likely to be two factors going on in this scenario. Firstly, there is the 'oversized' response being discarded by the resolver, which would account for a 2.4% failure rate based on the data from the smaller unfragmented packets. The additional failure component appears to be related to a fragmentation drop behaviour, which appear to account for the remaining 12% failure rate. In IPv6 the fragmentation-related drop rate appears to account for 15.2% of failure cases while in IPv4 the 'oversize' drop rate is higher and the residual fragmentation drop rate is 12%.

Why isn't the IPv6 fragmentation drop rate of 15.2% even higher? Other studies have reported IPv6 fragmentated packet drop rates between 20% to upward of 45%.

The reason probably lies in the particular circumstances of this experiment. Here we are looking at the path between recursive resolvers and a small set of authoritative servers. The servers are located in a data centre hosted environment that admits fragmented IPv6 packets and the recursive resolvers would presumably be located in operationally managed facilities that would likely to be also managed to achieve operational robustness. In order words, here we are looking more at the 'core' of the network rather than the connections to the edges.

The higher IPv6 fragmented packet drop rates have generally been observed in studies using end-to-end measurements which would presumably include edge networks. This implies that this observed 15.2% IPv6 UDP fragmentation drop rate reflects aspects of the recursive-to-authoritative network path but is not a good starting point to make more universal claims about IPv6 fragmentation performance in the end-to-end Internet. It's also the case that the IPv4 fragmentation drop rate is 12% in this scenario. This is a critical observation, in that other studies of end-to-end fragmentation drop rates in IPv4 do not report such high levels of packet drop.

This implies that the observed IPv6 fragmentation drop rate is more likely to be due to specific security-based filter rules relating to UDP packet fragmentation rather than network behaviours dropping IPv6 packets with extension headers in this particular measurement scenario.

What is also somewhat unexpected is that the average query count is so high for failure cases when the response is fragmented (Table 2). The lack of a truncated response leads some resolution systems to re-query at a high rate over the 60 second measurement window.

DNS Size	Pass		Fail	
	UDP	Control	UDP	Control
1150	5.2		12.1	
1190	5.2		12.0	
1230	5.3	4.8	11.6	12.3
1270	5.3	4.9	11.0	12.0
1310	5.3	4.9	11.2	12.0
1350	5.3	4.9	11.2	12.5
1390	5.3	4.9	11.1	12.7
1430	5.3	4.9	12.3	12.6
1470	5.4	5.2	11.8	27.5
1510	7.8	5.8	91.0	46.1
1550	7.9	5.8	90.9	43.7
1590	7.8	5.8	90.9	43.6
1630	7.9	5.9	91.4	44.2

Table 2 – Average Query Count for UDP-only Test

A similar pattern is visible when looking at the average time taken to perform this resolution task (Table 3). While the average number of queries to successfully resolve a name rises by 2 queries for fragmented UDP packets, the average time taken to successfully complete the resolution process rises by a further 80ms on average when the UDP response is fragmented.

DNS Size	Pass		Fail	
	UDP	Control	UDP	Control
1150	201		7,420	
1190	200		7,395	
1230	201	66	6,903	5,801
1270	204	68	2,870	5,829
1310	203	68	2,934	5,970
1350	203	71	2,971	6,107
1390	203	72	2,925	6,163
1430	202	73	3,579	6,190
1470	234	247	6,501	12,385
1510	287	1,221	24,010	20,842
1550	289	1,230	23,863	19,708
1590	289	1,232	23,828	19,605
1630	293	1,250	23,710	19,799

Table 3 – Average Query Time (ms) for UDP-only Test

These results do not place fragmented UDP in a good light for the DNS, irrespective of the IP protocol version. There is a base rate of some 14% of experiments that fail when the only resolution mechanism is fragmented UDP, and this rises by a further 2.5% when IPv6-only is used. The elapsed time to resolve also stretches out, and 8 seconds on average for resolution of a name when fragmented UDP is the only resolution mechanism is simply too long a time to be useful.

The implication of these results suggests that the original recommendation in RFC 6891to use a default buffer size parameter value of 4,096 octets was overly optimistic about the performance characteristics of fragmented UDP when negotiating firewalls and filters in front of DNS resolvers. Avoiding UDP fragmentation in the DNS appears to be a prudent measure, not because of network drop per se, but because of the common operational conventions in filtering fragmented DNS over UDP packets.

Let's test this theory some more.

What if we alter our measurement environment to truncate every response larger than 512 octets and only serve larger DNS responses over TCP?

2. All TCP

When we force the buffer size to 512 for all received queries then the experiment server will use a truncated response for all queries received over UDP. The truncated response contains no answer section, so the resolver will need to perform the query over TCP to resolve the name. The results are shown in Table 4.

DNS Response Size	Tests	TCP Pass Rate	TCP Fail Rate	IPv4 Failure Rate	IPv6 Failure Rate	Control Failure Rate
1150	1,104,539	98.5%	1.6%	1.9%	1.6%	
1190	1,105,126	98.5%	1.6%	1.9%	1.6%	
1230	1,105,601	98.5%	1.6%	1.9%	1.6%	0.5%
1270	1,104,571	98.5%	1.6%	1.9%	1.6%	0.5%
1310	1,104,521	98.5%	1.6%	1.9%	1.6%	0.5%
1350	1,104,068	98.5%	1.6%	2.0%	1.6%	0.5%
1390	1,105,080	98.5%	1.6%	1.9%	1.6%	0.5%
1430	1,104,527	98.5%	1.6%	1.9%	1.6%	0.5%
1470	1,103,423	98.3%	1.8%	2.1%	1.8%	1.0%
1510	1,104,960	98.3%	1.8%	2.1%	1.8%	2.4%
1550	1,105,566	98.3%	1.8%	2.1%	1.8%	2.6%
1590	1,103,609	98.3%	1.8%	2.1%	1.8%	2.6%
1630	1,106,284	98.3%	1.8%	2.1%	1.8%	2.6%

Table 4 – Failure Rate on TCP Test

It appears that some 1.6% of users sit behind a resolver that cannot perform DNS over TCP. If we look at the users behind IPv4-capable resolvers, then the proportion rises slightly to 1.9%. When we look at the subset of users behind IPv6-capable resolvers the number drops slightly to 1.6%. It is likely that more recent resolver deployments support both IPv6 and TCP, while there is a set of legacy resolver systems that do not support IPv6 and a higher proportion of these resolvers do not support TCP.

The failure rate rises slightly, by 0.2%, when the TCP response requires two TCP segments. This also means that the first TCP segment is sent using a segment size equal to the receiver's offered MSS value. If there are any path MTU issues on the TCP path, then the first full-size packet may encounter a TCP black hole situation where the ICMP message is not passed back to the TCP sender (the DNS server), and the TCP connection hangs.

DNS Response Size	Failure Count	NO TCP Failure	TCP ACK Failure	TCK OK Failure
1150	16,090	60%	2.5%	37%
1190	16,235	60%	2.7%	38%
1230	16,287	59%	2.4%	39%
1270	16,258	59%	2.4%	38%
1310	16,272	60%	2.6%	38%
1350	16,249	59%	2.3%	39%
1390	16,099	59%	2.7%	38%
1430	16,373	60%	2.2%	38%
1470	18,092	53%	11.8%	35%
1510	18,055	53%	12.7%	34%
1550	18,220	53%	12.5%	35%
1590	18,469	52%	12.4%	35%
1630	18,283	52%	12.1%	35%

Table 5 – ACK Failure Rate on TCP Test

This appears to be the reason behind the increased failure rate in TCP ACK failure when the DNS payload exceeds the MSS and the response is delivered using a full-sized packet (where the offered MSS equals the outbound MTU minus the packet header overheads. As we noted in the first part of this report (Figure 11 of DNS XL, Part 1), some 80% of TCP sessions over IPv4 and 57% of TCP sessions over IPv6 use an MSS setting in the TCP session that assumes a 1,500-octet path MTU.

However, the more dominant factors when failure occurs are cases where there is no TCP at all and cases where there is what appears to be a successfully completed TCP transaction.

More than half the time failure occurs when the resolver cannot open the TCP and pass the query to the server. Most likely this is an enthusiastic filter setting close to the resolver that does not allow the DNS to use TCP port 53.

The other failure mode is not so readily explained. In a little over one third of cases the TCP session passes the response to the remote client and the client end of the TCP session acknowledges the data. This would normally lead us to conclude that the resolver now has the data. But the resolver does not then complete the overall DNS resolution process. It is unclear why this occurs. A possible explanation is that the DNS application is discarding TCP responses that exceed its UDP payload size, although why a resolver would apply a UDP maximum payload setting to responses received over TCP is not readily explained.

The average query count for pass experiments is 1 – 2 queries greater than the control, and 1 query greater than the UDP-only count for smaller packets and much the same as UDP-only for larger DNS responses. The query count for failed experiments is 10 times higher than UDP-only for smaller packets, and similar for the larger DNS responses (Table 6).

DNS Size	Pass		Fail	
	TCP	Control	TCP	Control
1150	7.1		104.1	
1190	6.9		110.1	
1230	6.9	4.8	87.8	12.3
1270	6.9	4.9	88.7	12.0
1310	6.9	4.9	100.6	12.0
1350	6.8	4.9	91.1	12.5
1390	7.1	4.9	86.3	12.7
1430	6.9	4.9	68.8	12.6
1470	7.0	5.2	78.7	27.5
1510	6.9	5.8	75.1	46.1
1550	6.9	5.8	94.8	43.7
1590	6.9	5.8	98.3	43.6
1630	6.9	5.9	73.6	44.2

Table 6 – Average Query Count for TCP-only Test

TCP takes some additional time to start in the DNS. There is 1 round trip time to deliver the UDP truncated response and a further round trip time to complete the TCP handshake, so we can expect the delay with TCP to be longer than simple UDP. Compared to the results in Table 3 (where only UDP was used), the results for this TCP-only experiment show's an increased the elapsed time by a little under double the time (Table 7). However, larger responses are delivered reliably. Unlike fragmented UDP, the TCP failure rate is consistently low.

DNS Size	Pass		Fail	
	TCP	Control	TCP	Control
1150	341		33,938	
1190	299		30,372	
1230	331	66	29,431	5,801
1270	369	68	28,893	5,829
1310	340	68	28,887	5,970
1350	382	71	29,169	6,107
1390	339	72	28,734	6,163
1430	366	73	31,033	6,190
1470	298	247	28,351	12,385
1510	331	1,221	30,201	20,842
1550	410	1,230	28,335	19,708
1590	315	1,232	28,387	19,605
1630	321	1,250	29,088	19,799

Table 7 – Average Query Time (ms) for TCP-only Test

It appears that unfragmented UDP is both fast and reliable, while for larger responses where UDP fragmentation is unavoidable TCP is more reliable, albeit somewhat slower. What happens when we force this behaviour by setting the buffer size in all queries to a value where UDP fragmentation is avoided?

3. Buffer Size of 1,232 octets

The next scenario to be explored here is that being used in DNS Flag Day 2020. Here we set our server to behave as if all incoming queries use a buffer size of 1,232 octets. The intent here is to use UDP when we can be reasonably confident that the UDP packet will not encounter UDP fragmentation scenarios, and then shift to TCP for larger responses. The shift to TCP is of course controlled by the server providing a truncated response in UDP. In our case we are once again pushing this beyond conventional behaviour, in that we are not loading an answer section into the truncated response. The only way that the resolver will receive the response is by using TCP once the DNS response size exceeds 1,232 octets. The results of this measurement experiment are shown in Table 8.

DNS Response Size	Tests	1232 Pass Rate	1232 Fail Rate	1232 IPv4 Failure Rate	1232 IPv6 Failure Rate	Control Failure Rate
1150	1,113,090	99.5%	0.5%	0.5%	0.6%	
1190	1,113,104	99.5%	0.5%	0.5%	0.6%	
1230	1,111,703	99.4%	0.6%	0.6%	0.7%	0.5%
1270	1,114,563	98.4%	1.6%	1.5%	1.8%	0.5%
1310	1,113,632	98.4%	1.6%	1.5%	1.8%	0.5%
1350	1,113,669	98.4%	1.6%	1.5%	1.8%	0.5%
1390	1,115,152	98.4%	1.6%	1.5%	1.8%	0.5%
1430	1,114,069	98.4%	1.6%	1.5%	1.8%	0.5%
1470	1,111,607	98.2%	1.8%	1.7%	2.0%	1.0%
1510	1,112,349	98.2%	1.8%	1.7%	1.9%	2.4%
1550	1,112,795	98.2%	1.8%	1.7%	2.0%	2.6%
1590	1,112,351	98.2%	1.8%	1.7%	1.9%	2.6%
1630	1,112,523	98.2%	1.8%	1.7%	2.0%	2.6%

Table 8 – Failure Rate on Buffer Size of 1,232 Test

Predictably, we see the UDP-only failure rate (0.5%) for DNS responses of less than 1,232 octets and the TCP-only failure rate (1.6%) for larger packets. This is comparable to the control experiment for smaller responses, slightly worse than the control for responses up to 1,430 octets and slightly better for larger responses.

The average query count in this case is 2 queries more than the control case for smaller DNS responses and 1 query more for larger responses.

DNS Size	Pass		Fail	
	1232	Control	1232	Control
1150	7.1		104.1	
1190	6.9		110.1	
1230	6.9	4.8	87.8	12.3
1270	6.9	4.9	88.7	12.0
1310	6.9	4.9	100.6	12.0
1350	6.8	4.9	91.1	12.5
1390	7.1	4.9	86.3	12.7
1430	6.9	4.9	68.8	12.6
1470	7.0	5.2	78.7	27.5
1510	6.9	5.8	75.1	46.1
1550	6.9	5.8	94.8	43.7
1590	6.9	5.8	98.3	43.6
1630	6.9	5.9	73.6	44.2

Table 9 – Average Query Count for Buffer Size 1,232 Test

The elapsed time to complete resolution rises once the DNS payload exceeds 1,232 octets, and there is on average a further 100ms to complete the resolution process for these larger packets. This is due to the overheads of the truncated DNS response and the TCP handshake time for these response sizes.

DNS Size	Pass		Fail	
	1232	Control	1232	Control
1150	185		7,118	
1190	185		7,375	
1230	184	66	7,049	5,801
1270	290	68	18,805	5,829
1310	289	68	18,725	5,970
1350	290	71	18,986	6,107
1390	290	72	18,809	6,163
1430	290	73	18,594	6,190
1470	293	247	17,958	12,385
1510	292	1,221	18,193	20,842
1550	290	1,230	17,933	19,708
1590	292	1,232	18,162	19,605
1630	295	1,250	18,060	19,799

Table 10 – Average Query Time (ms) for Buffer Size 1,232 Test

With an overall loss rate of 1.8% for DNS payloads larger than 1,232 octets the obvious question is whether we can improve on this scenario. What if we lift the buffer size to just below the onset of UDP packet fragmentation, namely at 1,440 octets?

4. Buffer Size of 1,440 octets

Let's now look at the scenario of lifting of the threshold point to switch to TCP to just below a packet size of 1,500 octets. We will force all queries to use a buffer size setting of 1,440 octets.

We know from the all UDP experiment (Table 1) that there is an elevated response loss rate when the DNS payload size in UDP exceeds the resolver-client specified buffer size in the query, and this is visible in Table 11. This appears to account for a minimum of some 2% of the 2.6% observed failure rate for these smaller-sized packets.

The UDP loss rate for this size range exceeds the TCP loss rate that we observed in Table 8 where the lower buffer size setting of 1,232 octets was used.

DNS Response Size	Tests	1440 Pass Rate	1440 Fail Rate	1440 IPv4 Failure Rate	1440 IPv6 Failure Rate	Control Failure Rate
1150	1,113,090	99.5%	0.5%	0.6%	0.6%	
1190	1,113,104	99.5%	0.5%	0.6%	0.6%	
1230	1,111,703	99.4%	0.6%	0.7%	0.6%	0.5%
1270	1,114,563	97.5%	2.5%	2.3%	2.9%	0.5%
1310	1,113,632	97.5%	2.5%	2.3%	2.9%	0.5%
1350	1,113,669	97.4%	2.6%	2.3%	3.0%	0.5%
1390	1,115,152	97.5%	2.5%	2.3%	2.9%	0.5%
1430	1,114,069	97.2%	2.8%	2.6%	3.2%	0.5%
1470	1,111,607	98.3%	1.7%	1.8%	1.7%	1.0%
1510	1,112,349	98.3%	1.7%	1.8%	1.8%	2.4%
1550	1,112,795	98.3%	1.7%	1.9%	1.7%	2.6%
1590	1,112,351	98.3%	1.7%	1.8%	1.8%	2.6%
1630	1,112,523	98.3%	1.7%	1.8%	1.7%	2.6%

Table 11 – Failure Rate on Buffer Size 1,440 Test

The UDP average query count is uniformly low up until the TCP point, and the truncation and switch to TCP lifts the average query count for successful resolution efforts by slightly over 2 queries. The unsuccessful query count is more than quadrupled when there is a shift to TCP (Table 12).

DNS Size	Pass		Fail	
	1440	Control	1440	Control
1150	4.3		29.3	
1190	4.3		26.9	
1230	4.3	4.8	28.4	12.3
1270	4.3	4.9	28.9	12.0
1310	4.3	4.9	30.8	12.0
1350	4.3	4.9	30.8	12.5
1390	4.3	4.9	29.1	12.7
1430	4.3	4.9	29.5	12.6
1470	6.6	5.2	154.6	27.5
1510	6.6	5.8	142.7	46.1
1550	6.6	5.8	133.2	43.7
1590	6.8	5.8	187.2	43.6
1630	6.7	5.9	180.4	44.2

Table 12 – Average Query Count for Buffer Size 1,440 Test

The UDP-based retrieval is also considerably faster than TCP, completing the resolution in an average of 130ms, compared to 260ms, which is consistent with the overheads of the TCP connection. (Table 13).

DNS Size	Pass		Fail	
	1440	Control	1440	Control
1150	156		26,500	
1190	133		23,522	
1230	134	66	24,315	5,801
1270	131	68	24,761	5,829
1310	138	68	25,378	5,970
1350	157	71	25,027	6,107
1390	128	72	24,930	6,163
1430	142	73	24,375	6,190
1470	274	247	25,677	12,385
1510	228	1,221	26,329	20,842
1550	265	1,230	25,950	19,708
1590	267	1,232	25,610	19,605
1630	247	1,250	26,266	19,799

Table 13 – Average Query Time (ms) for Buffer Size 1,440 Test

This data suggests that the lower buffer size of 1,232 is more robust for resolvers, but it will add delays in resolution time and impose a greater query load on the server, both in terms of the TCP control overhead and the additional query volume for responses whose size falls into the range of 1,232 to 1,440 octets. It is possible, even likely, that the loss rate would fall were resolvers to use a default buffer size of 1,440 octets rather than 1,232 octets. The issue here appears to be application-level settings disregarding received packets and not an intrinsic behavioural property of the network path between the servers and recursive resolvers

5. Buffer Size of 1,440 octets, TCP MSS of 1,200

There is another variant to examine here, and that is to try and reduce the incidence of TCP path MTU issues. One way to achieve this is to drop the MTU setting on the server, so that it will not push out 1,500 octet IP packets. Another way is to modify the incoming MSS of TCP connection packets and rewrite the MSS to a lower value. In this experiment we've used the approach of rewriting the MSS on incoming TCP SYN packets, changing the MSS value to a value of 1,200 octets. This should reduce the TCP failure rate where the server sends the DNS data and does not receive an ACK for the data.

DNS Response Size	Tests	1440 Pass Rate	1440 Fail Rate	1440 IPv4 Failure Rate	1440 IPv6 Failure Rate	Control Failure Rate
1150	1,202,770	99.5%	0.5%	0.5%	0.1%	
1190	1,207,607	99.5%	0.5%	0.5%	0.1%	
1230	1,205,935	99.4%	0.6%	0.5%	0.1%	0.5%
1270	1,206,166	97.5%	2.5%	1.1%	0.2%	0.5%
1310	1,204,420	97.5%	2.5%	1.1%	0.2%	0.5%
1350	1,205,097	97.4%	2.6%	1.1%	0.2%	0.5%
1390	1,204,737	97.5%	2.5%	1.1%	0.2%	0.5%
1430	1,204,415	97.2%	2.8%	1.5%	0.9%	0.5%
1470	1,205,472	98.3%	1.7%	1.7%	1.6%	1.0%
1510	1,208,416	98.3%	1.7%	1.8%	1.6%	2.4%
1550	1,207,806	98.3%	1.7%	1.7%	1.6%	2.6%
1590	1,205,885	98.3%	1.7%	1.7%	1.6%	2.6%
1630	1,206,097	98.3%	1.7%	1.7%	1.6%	2.6%

Table 14 – Failure Rate on Buffer Size 1,440, MSS 1,200 Test

There is a very small change in the failure rate for DNS responses larger than 1,500 octets, and the change is around 0.1%. (Table 14) The change improves the IPv6 performance, dropping the failure rate for larger packets from 1.7% to 1.6%.

The query count profile is largely unaltered, as one would expect, although the level of query thrashing for large responses that fail in TCP is higher. One the issue of TCP "black hole" failure is removed then the other failure cases relating to TCP become the dominant factor, and the number of TCP queries that are made in 60 seconds increases once the stalled TCP sessions are eliminated (Figure 15).

DNS Size	Pass		Fail	
	1440	Control	1440	Control
1150	4.3		27.0	
1190	4.3		27.2	
1230	4.3	4.8	26.0	12.3
1270	4.3	4.9	27.8	12.0
1310	4.3	4.9	29.1	12.0
1350	4.3	4.9	28.9	12.5
1390	4.3	4.9	26.8	12.7
1430	4.3	4.9	26.8	12.6
1470	7.2	5.2	171.6	27.5
1510	7.0	5.8	205.5	46.1
1550	7.2	5.8	172.2	43.7
1590	7.2	5.8	187.5	43.6
1630	7.0	5.9	228.9	44.2

Table 15 – Average Query Count for Buffer Size 1,440, MSS 1,200 Test

DNS Size	Pass		Fail	
	1440	Control	1440	Control
1150	178		24,725	
1190	189		23,700	
1230	169	66	23,283	5,801
1270	201	68	23,550	5,829
1310	195	68	24,592	5,970
1350	177	71	23,693	6,107
1390	167	72	22,922	6,163
1430	172	73	21,919	6,190
1470	408	247	29,350	12,385
1510	494	1,221	29,500	20,842
1550	586	1,230	29,800	19,708
1590	506	1,232	29,676	19,605
1630	663	1,250	30,189	19,799

Table 16 – Average Query Time (ms) for Buffer Size 1,440, MSS 1200 Test

The profile of time to resolve is also similar, although the elapsed time for larger responses is somewhat larger (Figure 16).

6. Max Buffer size of 1,232 octets, TCP MSS of 1,200

So far, we have assumed a model where the resolver client is in control of the onset of UDP fragmentation by using the buffer size parameter in the EDNS(0) extension attached to a DNS query. Some DNS implementations also allow the server to also influence the onset of UDP fragmentation in DNS responses over UDP. In the Bind resolver the configuration option is the `max-udp-size` value:

> **max-udp-size**
>
> Sets the maximum EDNS UDP message size named will send in bytes. Valid values are 512 to 4096 (values outside this range will be silently adjusted). The default value is 4096. The usual reason for setting max-udp-size to a non-default value is to get UDP answers to pass through broken firewalls that block fragmented packets and/or block UDP packets that are greater than 512 bytes. This is independent of the advertised receive buffer (**edns-udp-size**).

The intent of this setting is to allow the server to set its own maximum UDP response size. If the query provides a lower value for the buffer size then the server will use it, but if the query has a higher buffer size value, then this local setting will be used. What happens when we combine this approach with the server-size imposed TCP MSS value of 1,200? The results of this experiment are shown in Table 17.

DNS Response Size	Tests	max 1232 Pass Rate	max 1232 Fail Rate	max 1232 IPv4 Failure Rate	max 1232 IPv6 Failure Rate	Control Failure Rate
1150	1,198,284	99.2%	0.8%	1.0%	0.7%	
1190	1,196,442	99.2%	0.8%	1.1%	0.7%	
1230	1,196,874	99.2%	0.8%	1.1%	0.7%	0.5%
1270	1,196,063	98.3%	1.7%	2.4%	1.3%	0.5%
1310	1,198,020	98.4%	1.6%	2.5%	1.3%	0.5%
1350	1,197,269	98.3%	1.7%	2.5%	1.4%	0.5%
1390	1,196,841	98.4%	1.6%	2.4%	1.3%	0.5%
1430	1,198,235	98.3%	1.7%	2.4%	1.3%	0.5%
1470	1,196,930	98.3%	1.7%	2.5%	1.3%	1.0%
1510	1,196,544	98.3%	1.7%	2.5%	1.3%	2.4%
1550	1,197,824	98.3%	1.7%	2.4%	1.3%	2.6%
1590	1,197,728	98.3%	1.7%	2.5%	1.4%	2.6%
1630	1,196,426	98.3%	1.7%	2.5%	1.3%	2.6%

Table 17 – Failure Rate on Buffer Size max 1,232 with 1,200 TCP MSS Test

The change here is that we are avoiding the case where the client drops the response because it is larger than the clients' originally specified maximum UDP response sizes. Because no UDP response is larger than 1,232 octets of payload then all intermediate sized responses (1,270 octets) and large responses (larger than 1430 octets) switch to TCP, and the larger TCP failure rate (of some 1.7%) kicks in. As observed already, the TCP failure rate for IPv4 resolvers is almost double the IPv6 failure rate.

The profile of number of queries (Table 18) and time to resolve (Table 19) the name is largely similar to the previous case,

DNS Size	Pass		Fail	
	<= 1232	Control	<= 1232	Control
1150	6.0		251.2	
1190	6.0		261.6	
1230	6.1	4.8	268.8	12.3
1270	7.6	4.9	160.7	12.0
1310	7.7	4.9	185.5	12.0
1350	7.6	4.9	170.1	12.5
1390	7.6	4.9	162.9	12.7
1430	7.7	4.9	154.7	12.6
1470	7.8	5.2	152.4	27.5
1510	7.5	5.8	140.8	46.1
1550	7.6	5.8	179.9	43.7
1590	7.6	5.8	143.1	43.6
1630	7.6	5.9	172.3	44.2

Table 18 – Average Query Count for Buffer Size max 1,232, MSS 1,200 Test

DNS Size	Pass		Fail	
	<= 1232	Control	<= 1232	Control
1150	412		12,556	
1190	526		11,257	
1230	426	66	11,803	5,801
1270	609	68	28,076	5,829
1310	545	68	29,785	5,970
1350	510	71	29,608	6,107
1390	489	72	29,621	6,163
1430	521	73	28,852	6,190
1470	561	247	29,070	12,385
1510	687	1,221	29,540	20,842
1550	589	1,230	27,622	19,708
1590	590	1,232	29,055	19,605
1630	491	1,250	28,403	19,799

Table 19 – Average Query Time (ms) for Buffer Size max 1,232, MSS 1200 Test

7. Max Buffer size of 1,440 octets, TCP MSS of 1,200

This case is similar to case 6, but with the UDP-to-TCP threshold lifted to 1,440 octets.

DNS Response Size	Tests	max 1440 Pass Rate	max 1440 Fail Rate	max 1440 IPv4 Failure Rate	max 1440 IPv6 Failure Rate	Control Failure Rate
1150	1,199,907	99.2%	0.8%	0.9%	0.6%	
1190	1,201,395	99.2%	0.8%	1.0%	0.7%	
1230	1,201,890	99.2%	0.8%	0.9%	0.7%	0.5%
1270	1,200,722	99.2%	0.8%	1.0%	0.6%	0.5%
1310	1,201,045	99.2%	0.8%	1.0%	0.7%	0.5%
1350	1,200,161	99.2%	0.8%	0.9%	0.7%	0.5%
1390	1,200,845	99.2%	0.8%	0.9%	0.7%	0.5%
1430	1,201,287	99.2%	0.8%	1.0%	0.6%	0.5%
1470	1,200,239	98.3%	1.7%	2.1%	1.4%	1.0%
1510	1,202,629	98.3%	1.7%	2.1%	1.4%	2.4%
1550	1,200,767	98.3%	1.7%	2.1%	1.4%	2.6%
1590	1,202,288	98.3%	1.7%	2.1%	1.4%	2.6%
1630	1,203,165	98.3%	1.7%	2.1%	1.4%	2.6%

Table 20 – Failure Rate on Buffer Size max 1,440 with 1,200 TCP MSS Test

The outcomes for IPv4 and IPv6 non-fragmented packets in Table 20 are slightly better than the results in Table 14, particularly as it relates to DNS response sizes in the range 1,270 to 1,470 octets. It appears that some 2% of users sit behind recursive resolvers that will check the UDP DNS response size against the buffer size in the original query and reject the query if the response is larger than the query-specified size.

DNS Size	Pass		Fail	
	<= 1440	Control	<= 1440	Control
1150	6.2		303.6	
1190	6.1		183.5	
1230	6.1	4.8	273.8	12.3
1270	6.1	4.9	229.8	12.0
1310	6.0	4.9	253.0	12.0
1350	6.0	4.9	245.2	12.5
1390	6.3	4.9	215.2	12.7
1430	6.0	4.9	215.5	12.6
1470	7.5	5.2	170.8	27.5
1510	7.7	5.8	187.5	46.1
1550	7.4	5.8	129.2	43.7
1590	7.7	5.8	151.2	43.6
1630	7.7	5.9	148.7	44.2

Table 21 – Average Query Count for Buffer Size max 1,440, MSS 1,200 Test

DNS Size	Pass		Fail	
	<= 1440	Control	<= 1440	Control
1150	340		11,138	
1190	344		10,573	
1230	331	66	10,506	5,801
1270	317	68	10,696	5,829
1310	384	68	11,113	5,970
1350	359	71	11,701	6,107
1390	318	72	10,319	6,163
1430	314	73	11,497	6,190
1470	405	247	27,144	12,385
1510	497	1,221	27,214	20,842
1550	411	1,230	26,893	19,708
1590	376	1,232	27,901	19,605
1630	388	1,250	27,268	19,799

Table 22 – Average Query Time (ms) for Buffer Size max 1,440, MSS 1200 Test

The number of queries (Table 21) and query time (Table 22) show a marked performance improvement for intermediate-sized responses as would be expected .

Conclusions

Let's collect the results of these individual experiments into single table that look at the failure rates fro the various packet size management scenarios (Table 23).

There are a set of design trade-offs in the choices for transport for the DNS protocol.

For short responses UDP is an efficient and reliable transport vehicle. However, when the size of the UDP response is larger than the network path MTU and UDP fragmentation is required, then fragmentation packet losses create serious problems for the protocol, and it becomes unreliable.

For that reason, TCP will be more far more reliable than fragmented UDP for larger responses on average. However, TCP is slower and far less efficient than UDP and its basic reliability rate is worse than unfragmented UDP. If carriage efficiency and reliability is a consideration for the DNS, then unfragmented UDP is clearly superior to TCP, while TCP is clearly superior to fragmented UDP.

DNS Response Size	Failure Rates						
	Control	512 (TCP)	4096 (UDP)	1232	<= 1232	1440	<= 1440
1150		1.6%	0.4%	0.5%	0.5%	0.5%	0.5%
1190		1.6%	0.4%	0.5%	0.5%	0.5%	0.5%
1230	0.5%	1.6%	0.4%	0.6%	0.8%	0.6%	0.6%
1270	0.5%	1.6%	1.9%	1.6%	1.7%	2.5%	0.6%
1310	0.5%	1.6%	1.8%	1.6%	1.7%	2.5%	0.6%
1350	0.5%	1.6%	1.8%	1.6%	1.8%	2.6%	0.6%
1390	0.5%	1.6%	1.9%	1.6%	1.7%	2.5%	0.6%
1430	0.5%	1.6%	2.2%	1.6%	1.7%	2.8%	0.6%
1470	1.0%	1.8%	3.1%	1.8%	1.7%	1.7%	1.7%
1510	2.4%	1.8%	15.0%	1.8%	1.7%	1.7%	1.7%
1550	2.6%	1.8%	15.0%	1.8%	1.7%	1.7%	1.7%
1590	2.6%	1.8%	14.9%	1.8%	1.7%	1.7%	1.7%
1630	2.6%	1.8%	14.9%	1.8%	1.7%	1.7%	1.7%

Table 23 – Summary of Failure Rates

What this means is that UDP should be used for as long as it will not encounter fragmentation, and then the DNS should shift to TCP.

How can this be achieved? It is unreasonable to expect that a lightweight UDP-based packet exchange should perform a path MTU discovery operation for each and every transaction. This implies that both the client and the server should use conservative settings for transport parameters that avoid path MTU issues.

What should a DNS client do?

The DNS Flag Day 2020 settings are a good start, but I think that they don't quite catch the entirety of the space. Not only should a client use a EDNS(0) payload size setting equal to or less than 1452 in IPv6 (accounting for a 40 octet IPv6 header and an 8 octet UDP header), and 1472 in IPv4 (accounting for a 20 octet IPv4 header and an 8 octet UDP header). For TCP, a client should also use a TCP MSS setting less than 1440 octets in IPv6 (accounting for a 40 octet IPv6 header and an 20 octet TCP header) and 1460 octets in IPv4 (accounting for a 20 octet IPv4 header and an 20 octet TCP header).

What should a DNS server do?

The server should also avoid fragmentation, and it can do this by setting a maximum payload size value no larger than 1,452 in IPv6 and 1,472 in IPv4. It should also impose a ceiling on the size of outgoing TCP packets of 1,440 packets in IPv6 and 1,460 in IPv4.

Specific circumstances vary, and there is a difference between measurements at the edge of the Internet and within the infrastructure of the network. Our extensive measurements of the behaviour of the inner infrastructure of the Internet between recursive resolvers and authoritative servers indicate that the network behaviour is relatively uniform with IP packet sizes up to 1,500 octets. If we restrict ourselves to settings that relate only to the transactions between recursive resolvers and authoritative servers then the DNS Flag Day 2020 setting of 1,232 octets are too low. The result is that the transaction will invoke TCP too early. A more efficient outcome can be achieved by pushing the UDP packet size to 1,500 octets including the IP header.

At the same time, it is prudent to pull the TCP segment size down. The incremental performance cost of using a 1,200 octet MSS value is extremely small when looking at DNS transactions.

This leads to some recommendations for transport parameter values for DNS clients and servers, shown in Table 24. The intent of these settings is to use UDP all the way to 1,500 octets of IP packet size, then use TCP with a more conservative MSS setting that increases the reliability of TCP sessions.

	IPv4	IPv6
Client EDNS(0) Buffer Size	1,472	1,452
Client TCP MSS	1,200	1,200
Server Max Buffer Size	1,472	1,452
Server max TCP MSS	1,200	1,200

Table 24 – Summary of DNS transport settings, recursive to authoritative

It must be noted that these settings apply only to "inside" of the Internet in the path between recursive resolvers and authoritative servers. The edge of the Internet is shows greater levels of variability and it is probably prudent to use a lower UDP upper bound, although this is as aspect of the DNS where our measurement technique cannot gain a direct insight, so we've refrained from making any particular recommendations for the edge stub-to-recursive resolver scenario. It's likely that the TCP MSS setting of 1,200 octets would still make sense, but less clear if the higher buffer size parameter is equally applicable at the edge.

Disclaimer

The above views do not necessarily represent the views or positions of the Asia Pacific Network Information Centre.

Author

Geoff Huston is the Chief Scientist at APNIC, the Regional Internet Registry serving the Asia Pacific region.

www.potaroo.net