DNS Response Size Issues

Status of this Memo

By submitting this Internet-Draft, each author represents that any applicable patent or other IPR claims of which he or she is aware have been or will be disclosed, and any of which he or she becomes aware will be disclosed, in accordance with Section 6 of BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html.

Copyright Notice

Copyright (C) The Internet Society (2006). All Rights Reserved.

Abstract

With a mandated default minimum maximum message size of 512 octets, the DNS protocol presents some special problems for zones wishing to expose a moderate or high number of authority servers (NS RRs). This document explains the operational issues caused by, or related to this response size limit.
1 - Introduction and Overview

1.1. The DNS standard (see [RFC1035 4.2.1]) limits message size to 512 octets. Even though this limitation was due to the required minimum IP reassembly limit for IPv4, it became a hard DNS protocol limit and is not implicitly relaxed by changes in transport, for example to IPv6.

1.2. The EDNS0 protocol extension (see [RFC2671 2.3, 4.5]) permits larger responses by mutual agreement of the requestor and responder. However, deployment of EDNS0 cannot be expected to reach every Internet resolver in the short or medium term. The 512 octet message size limit remains in practical effect at this time.

1.3. Since DNS responses include a copy of the request, the space available for response data is somewhat less than the full 512 octets. Negative responses are quite small, but for positive and delegation responses, every octet must be carefully and sparingly allocated. This document specifically addresses delegation response sizes.

2 - Delegation Details

2.1. A delegation response will include the following elements:

   Header Section: fixed length (12 octets)
   Question Section: original query (name, class, type)
   Answer Section: (empty)
   Authority Section: NS RRset (nameserver names)
   Additional Section: A and AAAA RRsets (nameserver addresses)

2.2. If the total response size would exceed 512 octets, and if the data that would not fit was "required", then the TC bit will be set (indicating truncation). This will usually cause the requestor to retry using TCP, depending on what information was desired and what information was omitted. (For example, truncation in the authority section is of no interest to a stub resolver who only plans to consume the answer section.) If a retry using TCP is needed, the total cost of the transaction is much higher. See [RFC1123 6.1.3.2] for details on the requirement that UDP be attempted before falling back to TCP.

2.3. RRsets are never sent partially unless TC bit set to indicate truncation. When TC bit is set, the final apparent RRset in the final nonempty section must be considered "possibly damaged" (see [RFC1035 6.2], [RFC2181 9]).
2.4. With or without truncation, the glue present in the additional data section should be considered "possibly incomplete", and requestors should be prepared to re-query for any damaged or missing RRsets. Note that truncation of the additional data section might not be signalled via the TC bit since additional data is often optional.

2.5. DNS label compression allows a domain name to be instantiated only once per DNS message, and then referenced with a two-octet "pointer" from other locations in that same DNS message. If all nameserver names in a message are similar (for example, all ending in ".ROOT-SERVERS.NET"), then more space will be available for uncompressable data (such as nameserver addresses).

2.6. The query name can be as long as 255 characters of presentation data, which can be up to 256 octets of network data. In this worst case scenario, the question section will be 260 octets in size, which would leave only 240 octets for the authority and additional sections (after deducting 12 octets for the fixed length header.)

2.7. Average and maximum question section sizes can be predicted by the zone owner, since they will know what names actually exist, and can measure which ones are queried for most often. For cost and performance reasons, the majority of requests should be satisfied without truncation or TCP retry.

2.8. Some queries to non-existing names can be large, but this is not a problem because negative responses need not contain any answer, authority or additional records. (See [RFC2308 2.1] for more information about the format of negative responses.)

2.9. The minimum useful number of name servers is two, for redundancy (see [RFC1034 4.1]). In case of multihomed name servers, it is advantageous to include an address record from each of several name servers before including several address records for any one name server. If address records for more than one transport (for example, A and AAAA) are available, then it is advantageous to include records of both types early on, before the message is full.

2.10. The best case is no truncation at all. This is because many requestors will retry using TCP by reflex, or will automatically re-query for RRsets that are "possibly truncated", without considering whether the omitted data was actually necessary.

2.11. Each added NS RR for a zone will add a minimum of between 16 and 44 octets to every untruncated referral or negative response from the
zone’s authority servers (16 octets for an NS RR, 16 octets for an A RR, and 28 octets for an AAAA RR), in addition to whatever space is taken by the nameserver name (NS NSDNAME as well as A or AAAA owner name).

2.12. While DNS distinguishes between necessary and optional resource records, this distinction is according to protocol elements necessary to signify facts, and takes no official notice of protocol content necessary to ensure correct operation. For example, a nameserver name that is in or below the zone cut being described by a delegation is "necessary content," since there is no way to reach that zone unless the parent zone’s delegation includes "glue records" describing that name server’s addresses.

2.13. It is also necessary to distinguish between "explicit truncation" where a message could not contain enough records to convey its intended meaning, and so the TC bit has been set, and "silent truncation", where the message was not large enough to contain some records which were "not required", and so the TC bit was not set.

2.14. An delegation response should prioritize glue records as follows.

first
   All glue RRsets for one name server whose name is in or below the zone being delegated, or which has multiple address RRsets (currently A and AAAA), or preferably both;

second
   Alternate between adding all glue RRsets for any name servers whose names are in or below the zone being delegated, and all glue RRsets for any name servers who have multiple address RRsets (currently A and AAAA);

thence
   All other glue RRsets, in any order.

The goal of this priority scheme is to offer "necessary" glue first, avoiding silent truncation for this glue if possible.

2.15. If any "necessary content" is silently truncated, then it is advisable that the TC bit be set in order to force a TCP retry, rather than have the zone be unreachable. Note that a parent server’s proper response to a query for in-child glue or below-child glue is a referral rather than an answer, and that this referral MUST be able to contain the in-child or below-child glue, and that in outlying cases, only EDNS or TCP will be large enough to contain that data.
3 - Analysis

3.1. An instrumented protocol trace of a best case delegation response follows. Note that 13 servers are named, and 13 addresses are given. This query was artificially designed to exactly reach the 512 octet limit.

;; flags: qr rd; QUERY: 1, ANS: 0, AUTH: 13, ADDIT: 13
;; QUERY SECTION:
;; [23456789.123456789.123456789.123456789.123456789.com A IN] ; 080

;; AUTHORITY SECTION:
com. 86400 NS E.GTLD-SERVERS.NET. ; 0112
com. 86400 NS F.GTLD-SERVERS.NET. ; 0128
com. 86400 NS G.GTLD-SERVERS.NET. ; 0144
com. 86400 NS H.GTLD-SERVERS.NET. ; 0160
com. 86400 NS I.GTLD-SERVERS.NET. ; 0176
com. 86400 NS J.GTLD-SERVERS.NET. ; 0192
com. 86400 NS K.GTLD-SERVERS.NET. ; 0208
com. 86400 NS L.GTLD-SERVERS.NET. ; 0224
com. 86400 NS M.GTLD-SERVERS.NET. ; 0240
com. 86400 NS A.GTLD-SERVERS.NET. ; 0256
com. 86400 NS B.GTLD-SERVERS.NET. ; 0272
com. 86400 NS C.GTLD-SERVERS.NET. ; 0288
com. 86400 NS D.GTLD-SERVERS.NET. ; 0304

;; ADDITIONAL SECTION:
A.GTLD-SERVERS.NET. 86400 A 192.5.6.30 ; 0320
B.GTLD-SERVERS.NET. 86400 A 192.33.14.30 ; 0336
C.GTLD-SERVERS.NET. 86400 A 192.26.92.30 ; 0352
D.GTLD-SERVERS.NET. 86400 A 192.31.80.30 ; 0368
E.GTLD-SERVERS.NET. 86400 A 192.12.94.30 ; 0384
F.GTLD-SERVERS.NET. 86400 A 192.35.51.30 ; 0400
G.GTLD-SERVERS.NET. 86400 A 192.42.93.30 ; 0416
H.GTLD-SERVERS.NET. 86400 A 192.54.112.30 ; 0432
I.GTLD-SERVERS.NET. 86400 A 192.43.172.30 ; 0448
J.GTLD-SERVERS.NET. 86400 A 192.48.79.30 ; 0464
K.GTLD-SERVERS.NET. 86400 A 192.52.178.30 ; 0480
L.GTLD-SERVERS.NET. 86400 A 192.41.162.30 ; 0496
M.GTLD-SERVERS.NET. 86400 A 192.55.83.30 ; 0512

;; MSG SIZE sent: 80 rcvd: 512
3.2. For longer query names, the number of address records supplied will be lower. Furthermore, it is only by using a common parent name (which is GTLD-SERVERS.NET in this example) that all 13 addresses are able to fit. The following output from a response simulator demonstrates these properties:

```perl
% perl respsize.pl a.dns.br b.dns.br c.dns.br d.dns.br
a.dns.br requires 10 bytes
b.dns.br requires 4 bytes
c.dns.br requires 4 bytes
d.dns.br requires 4 bytes
# of NS: 4
For maximum size query (255 byte):
  only A is considered:        # of A is 4 (green)
  A and AAAA are considered:   # of A+AAAA is 3 (yellow)
  preferred-glue A is assumed: # of A is 4, # of AAAA is 3 (yellow)
For average size query (64 byte):
  only A is considered:        # of A is 4 (green)
  A and AAAA are considered:   # of A+AAAA is 4 (green)
  preferred-glue A is assumed: # of A is 4, # of AAAA is 4 (green)
```

```perl
% perl respsize.pl ns-ext.isc.org ns.psg.com ns.ripe.net ns.eu.int
ns-ext.isc.org requires 16 bytes
ns.psg.com requires 12 bytes
ns.ripe.net requires 13 bytes
ns.eu.int requires 11 bytes
# of NS: 4
For maximum size query (255 byte):
  only A is considered:        # of A is 4 (green)
  A and AAAA are considered:   # of A+AAAA is 3 (yellow)
  preferred-glue A is assumed: # of A is 4, # of AAAA is 2 (yellow)
For average size query (64 byte):
  only A is considered:        # of A is 4 (green)
  A and AAAA are considered:   # of A+AAAA is 4 (green)
  preferred-glue A is assumed: # of A is 4, # of AAAA is 4 (green)
```

(Note: The response simulator program is shown in Section 5.)

Here we use the term "green" if all address records could fit, or "yellow" if two or more could fit, or "orange" if only one could fit, or "red" if no address record could fit. It’s clear that without a common parent for nameserver names, much space would be lost. For these examples we use an average/common name size of 15 octets, befitting our assumption of GTLD-SERVERS.NET as our common parent name.
We're assuming an average query name size of 64 since that is the typical average maximum size seen in trace data at the time of this writing. If Internationalized Domain Name (IDN) or any other technology which results in larger query names be deployed significantly in advance of EDNS, then new measurements and new estimates will have to be made.

4 - Conclusions

4.1. The current practice of giving all nameserver names a common parent (such as GTLD-SERVERS.NET or ROOT-SERVERS.NET) saves space in DNS responses and allows for more nameservers to be enumerated than would otherwise be possible, since the common parent domain name only appears once in a DNS message and is referred to via "compression pointers" thereafter.

4.2. If all nameserver names for a zone share a common parent, then it is operationally advisable to make all servers for the zone so served also be authoritative for the zone of that common parent. For example, the root name servers (?ROOT-SERVERS.NET) can answer authoritatively for the ROOT-SERVERS.NET. This is to ensure that the zone’s servers always have the zone’s nameservers’ glue available when delegating.

4.3. Thirteen (13) seems to be the effective maximum number of nameserver names usable traditional (non-extended) DNS, assuming a common parent domain name, and given that response truncation is undesirable as an average case, and assuming mostly IPv4-only reachability (only A RRs exist, not AAAA RRs).

XXX 4.4. Adding up to five IPv6 nameserver address records (AAAA RRs) to a prototypical delegation that currently contains thirteen (13) IPv4 nameserver addresses (A RRs) for thirteen (13) nameserver names under a common parent, would not have a significant negative operational impact on the domain name system.

5 - Source Code

#!/usr/bin/perl
#
# SYNOPSIS
#  repsize.pl [ -z zone ] fqdn_ns1 fqdn_ns2 ...
#  if all queries are assumed to have a same zone suffix,
#  such as "jp" in JP TLD servers, specify it in -z option
#
use strict;
use Getopt::Std;
my ($sz_msg) = (512);
my ($sz_header, $sz_ptr, $sz_rr_a, $sz_rr_aaaa) = (12, 2, 16, 28);
my ($sz_type, $sz_class, $sz_ttl, $sz_rdlen) = (2, 2, 4, 2);
my (%namedb, $name, $nssect, %opts, $optz);
my $n_ns = 0;

getopt('z', %opts);
if (defined($opts{'z'})) {
    server_name_len($opts{'z'})); # just register it
}

foreach $name (@ARGV) {
    my $len;
    $n_ns++;
    $len = server_name_len($name);
    print "$name requires $len bytes\n"
    $nssect += $sz_ptr + $sz_type + $sz_class + $sz_ttl
        + $sz_rdlen + $len;
}

print "# of NS: $n_ns\n";
arsect(255, $nssect, $n_ns, "maximum");
arsect(64, $nssect, $n_ns, "average");

sub server_name_len {
    my ($name) = @_; 
    my (%labels, $len, $n, $suffix);
    $name =~ tr/A-Z/a-z/;
    @labels = split(/\./, $name);
    $len = length(join('.', @labels)) + 2;
    for ($n = 0; $#labels >= 0; $n++, shift @labels) {
        $suffix = join('.', @labels);
        return length($name) - length($suffix) + $sz_ptr
            if (defined($namedb($suffix)));
    }
    return $len;
}

sub arsect {
    my ($sz_query, $nssect, $n_ns, $cond) = @_; 
    my ($space, $n_a, $n_a_aaaa, $n_p_aaaa, $sansect);
    $sansect = $sz_query + 1 + $sz_type + $sz_class;
    $space = $sz_query - $sz_header - $sansect - $nssect;
    $n_a = atmost(int($space / $sz_rr_a), $n_ns);
}
$n_a_aaaa = \text{atmost}(\text{int}(\frac{\text{space}}{\text{sz_rr_a + sz_rr_aaaa}}), n_ns);
$n_p_aaaa = \text{atmost}(\text{int}(\frac{\text{space} - \text{sz_rr_a} \times n_ns}{\text{sz_rr_aaaa}}), n_ns);
\text{printf} \ "For %s size query (%d byte):
", \$cond, \$sz_query;
\text{printf} \ " only A is considered: \"; \n\text{printf} \ " \# of A is %d (%s)\n", n_a, \&\text{judge}(n_a, n_ns);
\text{printf} \ " A and AAAA are considered: \"; \n\text{printf} \ " \# of A+AAAA is %d (%s)\n", n_a_aaaa, \&\text{judge}(n_a_aaaa, n_ns);
\text{printf} \ " preferred-glue A is assumed: \"; \n\text{printf} \ " \# of A is %d, \# of AAAA is %d (%s)\n", n_a, n_p_aaaa, \&\text{judge}(n_p_aaaa, n_ns);
}

sub judge(
    my (\$n, \$n_ns) = @_;
    return "green" if (\$n >= \$n_ns);
    return "yellow" if (\$n >= 2);
    return "orange" if (\$n == 1);
    return "red";
}

sub atmost(
    my (\$a, \$b) = @_;
    return 0 if (\$a < 0);
    return \$b if (\$a > \$b);
    return \$a;
}

6 - Security Considerations

The recommendations contained in this document have no known security implications.

7 - IANA Considerations

This document does not call for changes or additions to any IANA registry.

8 - Acknowledgement The authors thank Peter Koch and Rob Austein for their valuable comments and suggestions.
9 - References


10 - Authors’ Addresses

Paul Vixie
950 Charter Street
Redwood City, CA 94063
+1 650 423 1301
vixie@isc.org

Akira Kato
University of Tokyo, Information Technology Center
2-11-16 Yayoi Bunkyo
Tokyo 113-8658, JAPAN
+81 3 5841 2750
kato@wide.ad.jp

Full Copyright Statement

Copyright (C) The Internet Society (2006).

This document is subject to the rights, licenses and restrictions contained in BCP 78, and except as set forth therein, the authors retain all their rights.

This document and the information contained herein are provided on an "AS IS" basis and THE CONTRIBUTOR, THE ORGANIZATION HE/SHE REPRESENTS OR IS SPONSORED BY (IF ANY), THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Intellectual Property

The IETF takes no position regarding the validity or scope of any Intellectual Property Rights or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; nor does it represent that it has made any independent effort to identify any such rights. Information on the procedures with respect to rights in RFC documents can be found in BCP 78 and BCP 79.

Copies of IPR disclosures made to the IETF Secretariat and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such
proprietary rights by implementers or users of this specification can be obtained from the IETF on-line IPR repository at http://www.ietf.org/ipr.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights that may cover technology that may be required to implement this standard. Please address the information to the IETF at ietf-ipr@ietf.org.

Acknowledgement

Funding for the RFC Editor function is provided by the IETF Administrative Support Activity (IASA).