

DNSSD
Internet-Draft
Intended status: Standards Track
Expires: 31 August 2026

R. Bellis
ISC
27 February 2026

DNS Multiple QTYPEs
draft-ietf-dnssd-multi-qtypes-14

Abstract

This document specifies a method for a DNS client to request additional DNS record types to be delivered alongside the primary record type specified in the question section of a DNS QUERY (OpCode=0).

About This Document

This note is to be removed before publishing as an RFC.

The latest revision of this draft can be found at <https://dnssd-wg.github.io/draft-ietf-dnssd-multi-qtypes/draft-ietf-dnssd-multi-qtypes.html>. Status information for this document may be found at <https://datatracker.ietf.org/doc/draft-ietf-dnssd-multi-qtypes/>.

Discussion of this document takes place on the DNSSD Working Group mailing list (<mailto:dnssd@ietf.org>), which is archived at <https://mailarchive.ietf.org/arch/browse/dnssd/>. Subscribe at <https://www.ietf.org/mailman/listinfo/dnssd/>.

Source for this draft and an issue tracker can be found at <https://github.com/dnssd-wg/draft-ietf-dnssd-multi-qtypes>.

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/>.

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."

This Internet-Draft will expire on 31 August 2026.

Copyright Notice

Copyright (c) 2026 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Revised BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Revised BSD License.

Table of Contents

1. Introduction	2
2. Terminology	3
3. Specification	3
3.1. Multiple QTYPE EDNS Options Format	3
3.2. Client Request Generation	4
3.3. Server Request Parsing	5
3.4. Server Response Generation	5
3.5. Client Response Processing	7
4. Security Considerations	8
5. IANA Considerations	8
Acknowledgements	8
References	9
Normative References	9
Informative References	9
Appendix A. Examples	10
A.1. Stub query for A with MQType-Query for AAAA + HTTPS	10
A.2. Stub query for DS with MQType-Query for DNSKEY	11
A.3. Recursive query for DS with MQType-Query for NS	12
Author's Address	13

1. Introduction

A commonly requested DNS [STD13] feature is the ability to receive multiple related resource records (RRs) in a single DNS response.

For example, it may be desirable to receive the A, AAAA, and HTTPS RRs for a domain name together, rather than having to issue multiple queries.

The DNS wire protocol in theory supported having multiple questions in a single packet, but in practice this does not work. In [RFC9619], RFC1035 is updated to only permit a single question in a QUERY (OpCode=0) request.

Sending QTYPE=ANY does not guarantee that all resource record sets (RRsets) will be returned. Section 4.1 of [RFC8482] specifies that responders may return a single RRset of their choosing.

This document provides a solution for those cases where only the QTYPE varies by specifying a new option for the Extension Mechanisms for DNS (EDNS) [RFC6891] that contains an additional list of QTYPE values that the client wishes to receive in addition to the single QTYPE appearing in the question section. A different EDNS option is used in response packets as protection against DNS middleboxes that echo EDNS options verbatim.

The specification described herein is applicable both for queries from a stub resolver to recursive servers, and from recursive resolvers to authoritative servers. It does not apply to Multicast DNS queries [RFC6762], which are already designed to allow requesting multiple records in a single query, but is applicable to DNS-Based Service Discovery (DNS-SD) [RFC6763].

2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

This document makes use of DNS terminology defined in [RFC9499].

3. Specification

3.1. Multiple QTYPE EDNS Options Format

The overall format of an EDNS option is shown for reference in Figure 1, per [RFC6891], followed by the option specific data.

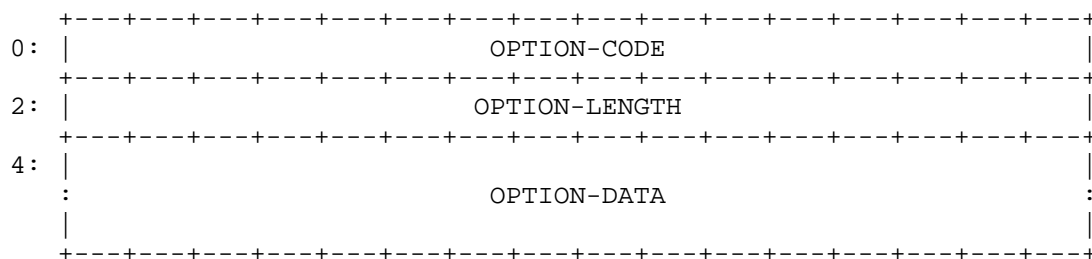


Figure 1: EDNS Option Format

OPTION-CODE: MQTYPE-Query (20) in queries and MQTYPE-Response (21) in responses.

OPTION-LENGTH: Size (in octets) of OPTION-DATA.

OPTION-DATA: Option specific, as depicted in Figure 2.

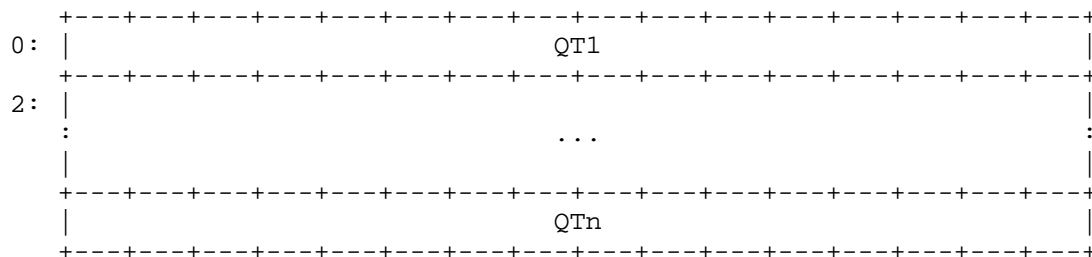


Figure 2: MQTYPE OPTION-DATA Format

A list of 2-octet values in network order (MSB first) each specifying a DNS RRTYPE that must be for a data RRTYPE as described in Section 3.1 of [RFC6895]. Individual values from the list (with unspecified index "x") are referred to as QT_x in the following sections.

3.2. Client Request Generation

DNS clients implementing this specification MUST generate packets that conform to the server request parsing rules described in Section 3.3.

The choice of when a client implementation should attempt to coalesce queries for multiple QTYPES using this method is implementation specific and not discussed further herein. However, careful considerations should be taken into account when coalescing queries on behalf of an application if such a feature is not explicitly

requested by the application. How an application interacts with an underlying name resolution library is internal to the implementation and is thus out of scope.

3.3. Server Request Parsing

In addition to the error cases discussion in Section 7 of [RFC6891], the server MUST return a FORMERR response if the server receives:

- * An MQTYPE-Query option in any inbound DNS message with an OpCode other than QUERY (0).
- * An MQTYPE-Response option in any inbound DNS message.
- * More than one MQTYPE-Query option in a query.
- * An MQTYPE-Query in a query that contains no primary question (i.e. QDCOUNT=0).
- * An MQTYPE-Query option in a query where the primary question is a non-data RRTYPE (e.g. ANY, AXFR).
- * An MQTYPE-Query option with an empty QT list.
- * An invalid QTx (e.g. one corresponding to a Meta RRTYPE).
- * A duplicate QTx (or one duplicating the primary QTYPE field) in a query.

3.4. Server Response Generation

A conforming server that receives an MQTYPE-Query option in a valid query MUST return an MQTYPE-Response option in its response, even if that response is truncated (TC=1). This is necessary to indicate that the server does support this extension. Refer to Section 3.3 for invalid queries.

The server MUST first start constructing a response for the primary (QNAME, QCLASS, QTYPE) tuple specified in the Question section per the existing DNS sections. The RCODE and all other flags (such as AA or AD) MUST be determined at this time.

If this initial response results in truncation (TC=1) then the additional queries specified in the MQTYPE-Query option MUST NOT be processed.

After the initial response is prepared, the server MUST attempt to combine the responses for individual (QNAME, QCLASS, QTx) combinations into the response for the first query. If a recursive server does not yet have those responses available it MAY first make appropriate outbound queries to populate its caches.

For each individual combination the server MUST evaluate the resulting RCODE and other flags and check that they all match the values generated from the primary query.

If any mismatch is detected the mismatching additional response MUST NOT be included in the final combined response and its QTx value MUST NOT be included in the MQTYPE-Response option's list. This might happen, for example, if the primary query resulted in a NOERROR response but a QTx query resulted in a SERVFAIL, or if the primary response has AA=0 but a QTx response has AA=1, such as might happen if the NS and DS records were both requested at the parent side of a zone cut.

The server MUST attempt to combine the remaining individual RRs into the same sections in which they would have appeared in a standalone query, i.e. as if each combination had been "the question" per Section 4.1 of [RFC1035].

The server MUST detect duplicate RRs and keep only a single copy of each RR in its respective section. Duplicates can occur, for example, in the Answer section if a CNAME chain is involved, or in the Authority section if multiple QTYPES don't exist, etc. Note that RRs can be legitimately duplicated in different sections such as for the (SOA, TYPE12345) combination at a zone apex where TYPE12345 is not present.

Handling of an MQTYPE-Query option MUST NOT itself trigger a truncated response. If response size (or other) limits do not allow all of the data obtained by querying for an additional QTx to be included in the final response in their entirety (i.e. as complete RRsets) then the server MUST NOT include the respective QTx in the MQTYPE-Response option's list and MAY stop processing further QTx combinations.

If all RRs for a single QTx combination fit into the message then the server MUST include the respective QTx in the MQTYPE-Response option's list to indicate that the given query type was completely processed.

Note that it is possible for the resulting MQTYPE-Response option to contain an empty list, but as described above the option must still be returned.

3.5. Client Response Processing

If the response to a query containing an MQTYPE-Query option does not contain an MQTYPE-Response option, or if it erroneously contains an MQTYPE-Query option, the client MUST treat the response as if this option is unsupported by the server and MUST process the primary response as if the MQTYPE-Query option had not been used.

In the above case, or if the server generates a FORMERR response, the client MUST issue additional standalone queries (that is, without using the MQTYPE-Query option) for all QTYPES for which an answer is still required.

If the MQTYPE-Response option is present more than once or if a QTx value is duplicated (or duplicates the primary QTYPE field) the client MUST treat the answer as invalid (equivalent to FORMERR).

The Question section and the list of types present in the MQTYPE-Response option indicates the list of (QNAME, QCLASS, qtypes) combinations which are completely answered and contained within the received response. The answers to all query combinations share the same RCODE and all other flags.

All RRs required by existing DNS specifications are expected to be present in the respective sections of the DNS message, including proofs of nonexistence where required. The client MUST NOT rely on any particular order of RRs in the message sections.

For the purposes of Section 5.4.1 of [RFC2181] any authoritative answers received MUST be ranked the same as the answer for the primary question.

Clients MUST take into account that individual RRs might originate from different DNS zones and that proofs of non-existence might have been produced by different signers.

Absence of QTx values which were requested by the client but are not present in the MQTYPE-Response option indicates that:

- * (for responses from recursive servers) the server does not have any records for that QTx value in cache, and/or
- * the individual responses could not be combined into one message because of RCODE or other flag mismatches, and/or
- * the server was unwilling to process the request (for example, because a limit was exceeded), and/or

- * the response size limit would be exceeded

The client MUST subsequently initiate separate standalone queries for all QTx values for which an answer is still required.

4. Security Considerations

The method documented here does not change any of the security properties of the DNS protocol itself.

It should however be noted that this method does increase the potential amplification factor when the DNS protocol is used as a vector for a denial-of-service attack. A further risk is being able to maliciously cause recursive servers to perform large amounts of additional work.

Implementors SHOULD therefore allow operators to configure limits on the number of QTx values specified and/or the resulting response size. The recommended values of those limits will depend on the environment in which this specification is used. In public DNS it is expected that a limit of four QTx values would be appropriate, but when used with DNS-SD or within private networks higher limits would be acceptable.

5. IANA Considerations

IANA has assigned the following values in the "DNS EDNS0 Option Codes (OPT)" registry within "Domain Name System (DNS) Parameters" registry group:

Value	Name	Status	Reference
20	MQTYPE-Query	Optional	RFC TBD
21	MQTYPE-Response	Optional	RFC TBD

Table 1: MQTYPE EDNS Option Numbers

Acknowledgements

The author wishes to thank the following for their feedback and reviews during the initial development of this document: Michael Graff, Olafur Gudmundsson, Matthijs Mekking, and Paul Vixie.

In addition the author wishes to thank the following for subsequent review during discussion in the DNSSD Working Group: Chris Box, Stuart Cheshire, Esko Dijk, Ted Lemon, David Schinazi and Petr Spacek.

References

Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/rfc/rfc2119>>.
- [RFC2181] Elz, R. and R. Bush, "Clarifications to the DNS Specification", RFC 2181, DOI 10.17487/RFC2181, July 1997, <<https://www.rfc-editor.org/rfc/rfc2181>>.
- [RFC6891] Damas, J., Graff, M., and P. Vixie, "Extension Mechanisms for DNS (EDNS(0))", STD 75, RFC 6891, DOI 10.17487/RFC6891, April 2013, <<https://www.rfc-editor.org/rfc/rfc6891>>.
- [RFC6895] Eastlake 3rd, D., "Domain Name System (DNS) IANA Considerations", BCP 42, RFC 6895, DOI 10.17487/RFC6895, April 2013, <<https://www.rfc-editor.org/rfc/rfc6895>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/rfc/rfc8174>>.
- [RFC9619] Bellis, R. and J. Abley, "In the DNS, QDCOUNT Is (Usually) One", RFC 9619, DOI 10.17487/RFC9619, July 2024, <<https://www.rfc-editor.org/rfc/rfc9619>>.
- [STD13] Internet Standard 13, <<https://www.rfc-editor.org/info/std13>>. At the time of writing, this STD comprises the following:
 - Mockapetris, P., "Domain names - concepts and facilities", STD 13, RFC 1034, DOI 10.17487/RFC1034, November 1987, <<https://www.rfc-editor.org/info/rfc1034>>.
 - Mockapetris, P., "Domain names - implementation and specification", STD 13, RFC 1035, DOI 10.17487/RFC1035, November 1987, <<https://www.rfc-editor.org/info/rfc1035>>.

Informative References

- [RFC6762] Cheshire, S. and M. Krochmal, "Multicast DNS", RFC 6762, DOI 10.17487/RFC6762, February 2013, <<https://www.rfc-editor.org/rfc/rfc6762>>.
- [RFC6763] Cheshire, S. and M. Krochmal, "DNS-Based Service Discovery", RFC 6763, DOI 10.17487/RFC6763, February 2013, <<https://www.rfc-editor.org/rfc/rfc6763>>.
- [RFC8482] Abley, J., Gudmundsson, O., Majkowski, M., and E. Hunt, "Providing Minimal-Sized Responses to DNS Queries That Have QTYPE=ANY", RFC 8482, DOI 10.17487/RFC8482, January 2019, <<https://www.rfc-editor.org/rfc/rfc8482>>.
- [RFC9499] Hoffman, P. and K. Fujiwara, "DNS Terminology", BCP 219, RFC 9499, DOI 10.17487/RFC9499, March 2024, <<https://www.rfc-editor.org/rfc/rfc9499>>.

Appendix A. Examples

The examples below are shown as might be reported by the ISC Dig utility. For the purposes of brevity irrelevant content is omitted.

A.1. Stub query for A with MQType-Query for AAAA + HTTPS

In this example a stub resolver has requested the A record for `www.example.com`, along with an MQTYPE-Query option requesting AAAA and HTTPS records. The stub resolver has also set the DO bit, indicating DNSSEC support.

The presence of the HTTPS QTYPE in the MQTYPE-Response option of the response coupled with its absence from the answer section indicates that the recursive server currently holds no data for this QTYPE. The corresponding type fields in the NSEC3 record further provide a cryptographic proof of non-existence for the HTTPS QTYPE and the SOA record also indicates a "negative answer".

```

;; ->>HEADER<- opcode: QUERY, status: NOERROR, id: 11111
;; flags: qr rd ra ad
;; QUERY: 1, ANSWER: 4, AUTHORITY: 4, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags: do; udp: 1232
; MQTYPE-Response: AAAA HTTPS

;; QUESTION SECTION:
;www.example.com.          IN   A

;; ANSWER SECTION:
www.example.com.    2849  IN   A      192.0.2.1
www.example.com.    2849  IN   RRSIG   A [...]
www.example.com.    3552  IN   AAAA    3fff::1234
www.example.com.    3552  IN   RRSIG   AAAA [...]

;; AUTHORITY SECTION:
example.com.        2830  IN   SOA      ns.example.com. [...]
example.com.        2830  IN   RRSIG    SOA 13 2 [...]
[...].example.com.  2830  IN   NSEC3    [...] A TXT AAAA RRSIG
[...].example.com.  2830  IN   RRSIG    NSEC3 [...]

```

Figure 3: A + AAAA + HTTPS

A.2. Stub query for DS with MQType-Query for DNSKEY

In this similar example, the primary QTYPE is for DS and the MQTYPE-Query field only contains DNSKEY.

Both the DS and DNSKEY records are returned, along with their corresponding RRSIG records.

```
;; ->>HEADER<- opcode: QUERY, status: NOERROR, id: 33333
;; flags: qr rd ra ad
;; QUERY: 1, ANSWER: 5, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags: do; udp: 1232
; MQTYPE-Response: DNSKEY

;; QUESTION SECTION:
;example.com.                IN      DS

;; ANSWER SECTION:
example.com.      625    IN    DNSKEY  256 3 13 [...]
example.com.      625    IN    DNSKEY  257 3 13 [...]
example.com.      625    IN    RRSIG   DNSKEY [...] example.com. [...]
example.com.     86185   IN    DS      370 13 2 [...]
example.com.     86185   IN    RRSIG   DS [...] com. [...]
```

Figure 4: Stub DS + DNSKEY

A.3. Recursive query for DS with MQType-Query for NS

In this instance, a recursive resolver is sending a DS record query to the parent zone's authoritative server and simultaneously requesting the NS records for the zone.

Since the DS record response is marked as authoritative (AA = 1) but the NS record data on the parent side of a zone cut is not authoritative (AA = 0) the server is unable to merge the responses, and the NS QTYPE is omitted from the MQTYPE-Response field.

```
;; ->>HEADER<- opcode: QUERY, status: NOERROR, id: 33333
;; flags: qr aa
;; QUERY: 1, ANSWER: 5, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags: do; udp: 1232
; MQTYPE-Response: [empty]

;; QUESTION SECTION:
;example.com.                IN    DS

;; ANSWER SECTION:
example.com.     86185   IN    DS      370 13 2 [...]
example.com.     86185   IN    RRSIG   DS [...] com. [...]
```

Figure 5: Recursive DS + NS

Author's Address

Ray Bellis
Internet Systems Consortium, Inc.
PO Box 360
Newmarket, NH 03857
United States of America
Phone: +1 650 423 1300
Email: ray@isc.org