deleg
Internet-Draft
Updates: 1034, 1035, 4035, 6672, 6840 (if approved)
Intended status: Standards Track
Expires: 23 April 2026

T. April
Google, LLC
P. paek
ISC
R. Weber
Akamai Technologies
D. Lawrence
Salesforce

20 October 2025

Extensible Delegation for DNS
draft-ietf-deleg-05

Abstract

This document proposes a new extensible method for the delegation of authority for a domain in the Domain Name System (DNS) using DELEG and DELEGI records.

A delegation in the DNS enables efficient and distributed management of the DNS namespace. The traditional DNS delegation is based on NS records which contain only hostnames of servers and no other parameters. The new delegation records are extensible, can be secured with DNSSEC, and eliminate the problem of having two sources of truth for delegation information.

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://github.com/ietf-wg-deleg/draft-ietf-deleg-base/tree/gh-pages. Status information for this document may be found at https://datatracker.ietf.org/doc/draft-ietf-deleg/.

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

Source for this draft and an issue tracker can be found at https://github.com/ietf-wg-deleg/draft-ietf-deleg-base/.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Copyright Notice

Table of Contents

1. Introduction 3
  1.1. Terminology 4
2. DELEG and DELEGI Resource Record Types 5
  2.1. Presentation Format 6
  2.2. RDATA Wire Format 6
  2.3. Overview of Differences between DELEG and DELEGI Semantics 7
3. Use of DELEG Records 8
  3.1. Resolvers 8
    3.1.1. Signaling DELEG Support 8
    3.1.2. Referral 9
    3.1.3. Parent-side types, QTYPE=DELEG 9
    3.1.4. Algorithm for "Finding the Best Servers to Ask" . . . 10
    3.1.5. Name Server Information for Delegation 12
    3.1.6. Populating the SLIST from DELEG and DELEGI Records . 13
  3.2. Authoritative Servers 14
    3.2.1. DELEG-aware Clients 14
    3.2.2. DELEG-unaware Clients 15
  3.3. DNSSEC Signers 17
  3.4. DNSSEC Validators 18

3.4.1. Clarifications on Nonexistence Proofs 18
3.4.2. Insecure Delegation Proofs 19
3.4.3. Referral downgrade protection 19
3.4.4. Chaining 19
4. Security Considerations 20
4.1. Preventing Over-work Attacks 20
4.2. Preventing Downgrade Attacks 20
5. IANA Considerations 21
5.1. Changes to Existing Registries 21
5.2. New Registry for Delegation Information 21
5.2.1. Procedure 21
5.2.2. Initial Contents 22
5.3. Temporary Assignments 23
6. References 23
6.1. Normative References 23
6.2. Informative References 24
Appendix A. Examples 25
A.1. Root zone file 25
A.2. Example.org zone file 27
A.3. Example.net zone file 27
A.4. Responses 27
A.4.1. DO bit clear, DE bit clear 27
A.4.2. DO bit set, DE bit clear 29
A.4.3. DO bit clear, DE bit set 30
A.4.4. DO bit set, DE bit set 31
A.5. DELEGI Interpretation 33
Acknowledgments 33
Authors' Addresses 33

1. Introduction

In the Domain Name System, responsibility for each subdomain within the domain name hierarchy can be delegated to different servers, which makes them authoritative for their portion of the namespace.

The original DNS record that does this, called an NS record, contains only the hostname of a single name server and no other parameters. The resolver needs to resolve these names into usable addresses and infer other required parameters, such as the transport protocol and any other protocol features. Moreover, the NS record set exists in two places--one at the delegation point, and the other, possibly different, in the child zone. The DNS Security Extensions (DNSSEC) protect only one copy, those in the child zone.

These properties of NS records limit resolvers to unencrypted UDP and TCP port 53, and this initial contact cannot be protected with DNSSEC. Even if these two problems were somehow solved, the NS record does not offer extensibility for any other parameters. This limitation is a barrier for the efficient introduction of new DNS technology.

The proposed DELEG and DELEGI resource record (RR) types remedy this problem by providing extensible parameters to indicate server capabilities and additional information, such as other transport protocols that a resolver may use.

The DELEG record creates a new delegation. It is authoritative in the parent side of delegation and thus can be signed with DNSSEC. This makes it possible to validate all delegation parameters, including those of future extensions.

The DELEGI record is an auxiliary record which does not create a delegation by itself but provides an optional layer of indirection. It can be used to share the same delegation information across any number of zones. DELEGI is treated like regular authoritative record.

The DELEG record can be used instead of or alongside a NS record to create a delegation. The combination of DELEG+NS is fully compatible with old resolvers, facilitating the incremental rollout of this new method of delegation.

Future documents can use the extensibility mechanism for more advanced features like connecting to a name server with an encrypted transport.

1.1. 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.

Terminology regarding the Domain Name System comes from [BCP219], with addition terms defined here:

* legacy delegation: A delegation that is done with an NS RRset

* legacy response: A response that does not use the DELEG protocol described in this document

* DELEG-aware: An authoritative server or resolver that follows the protocol defined in this document

* DELEG-unaware: An authoritative server or resolver that does not follow the protocol defined in this document

* non-DELEG specifications: DNS protocols that predate this protocol, or are written after this protocol is published but are not related to this protocol

2. DELEG and DELEGI Resource Record Types

The DELEG record, RR type TBD, and the DELEGI record, RR type TBD2 (different from that of DELEG), have the same wire and presentation formats, but their semantics are different as described in a following section. These records are defined for the IN class.

The record format is based on the extensible key=value list that was originally defined as "SvcParams" for the SVCB record type [RFC9460]. Unlike SVCB, the DELEG protocol does not have "SvcPriority" and "TargetName" fields. The keys in the DELEG protocol are different than those used in SVCB. To avoid confusion between the two protocols, the list of key=value parameters used by the DELEG protocol are called DelegInfos and will be tracked in their own IANA registry for Delegation Information.

The following rules are adapted from SVCB, but with changed names:

* The whole RDATA consists of a single list called "DelegInfos".

* DelegInfos consists of individual DelegInfo key=value pairs.

* Each DelegInfo pair has DelegInfoKey and a possibly optional DelegInfoValue.

* Each DelegInfo has a specified presentation format and wire encoding.

* Each DelegInfoKey has a presentation name and a registered key number.

* Each DelegInfoValue is in a format specific to its DelegInfoKey.

Implementations can reuse the same code to parse SvcParams and DelegInfos and only plug in a different list of key=value pairs for SVCB/HTTPS and DELEG/DELEGI record families.

The initial set of DelegInfoKeys and their formats are defined in Section 3.1.5.

2.1. Presentation Format

The RDATA presentation format of the DELEG and DELEGI resource records consists of a single list, DelegInfos.

The DelegInfos presentation format is defined exactly the same as SvcParams in Section 2.1 of [RFC9460]. The following rules are adapted from SVCB, but with changed names:

* DelegInfos is a whitespace-separated list with each DelegInfo consisting of a DelegInfoKey=DelegInfoValue pair, or a standalone DelegInfoKey.

* Individual element definitions are the same as [RFC9460]:

  - The DelegInfo syntax is the same as SvcParam, but it references DelegInfo elements instead of SvcParam elements.

  - DelegInfoKey syntax is the same as SvcParamKey.

  - The syntax for unknown keys in Section 2.1 of [RFC9460] applies.

  - The DelegInfoValue syntax is the same as SvcParamValue.

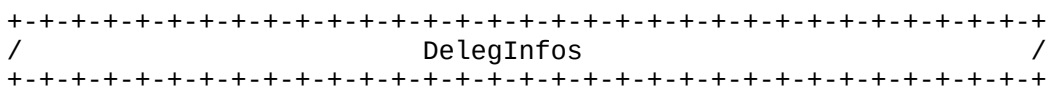
  - The rules from Appendix A of [RFC9460] apply.

* All the requirements in Section 2.1 of [RFC9460] apply.

DelegInfos MAY be zero-length; this is similar to what is allowed in SVCB records. A record with a zero-length DelegInfos field has no effect on the SLIST processing for resolvers.

2.2. RDATA Wire Format

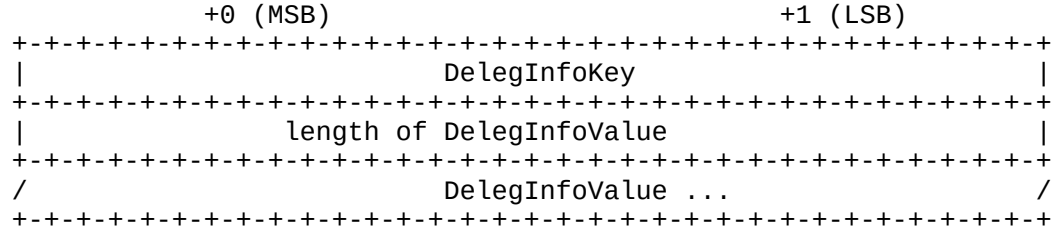
The RDATA portion of the DELEG and DELEGI resource record has variable length and entirely consists of a single "DelegInfos" element:



The format of the DelegInfos element is identical to the format of the SvcParams element defined in [RFC9460] Section 2.2, including the requirements for strictly increasing numeric order to keys and no key duplication allowed.

All the requirements in Section 2.2 of [RFC9460] apply.

The DelegInfos element is a sequence of individual DelegInfo elements. The wire format of an individual DelegInfo element is the same as for a SvcParam element, but it references DelegInfo elements instead of SvcParam elements.



The permissible lengths depend on the DelegInfoKey value. Some future keys may have no DelegInfoValue, which would be indicated with an explicit 0 length.

2.3. Overview of Differences between DELEG and DELEGI Semantics

The following is a brief summary of semantic differences between the DELEG and DELEGI types.

* DELEG creates a delegation for its owner name, similar to the NS RR type.

* DELEG and NS RR types can coexist at the same owner name.

* DELEG is authoritative in the parent zone of the delegated zone, similar to the DS RR type, and unlike the NS RR type.

* DELEG is signed by the parent zone of the delegated zone when using DNSSEC, similar to the DS RR type, and unlike the NS RR type.

* DELEG cannot be present at the apex of the delegated zone, similar to the DS RR type, and unlike the NS RR type.

* DELEG has special processing for being included in answers.

Conversely,

* DELEGI is like any normal RR and doesn’t require any special processing.

* DELEGI does not create a delegation for its owner name.

* DELEGI cannot coexist at the same owner name with DELEG or NS RR types.

* DELEGI DNSSEC signing and record placement rules are the same as for any ordinary RR type.

* DELEGI is used as the target of the DELEG protocol’s "include" mechanism, as described in section Section 3.1.6.

TODO: Add some introduction comparing how resolvers see legacy delegation (set of NS and A/AAAA records) and DELEG delegation (DELEG and DELEGI records with server-ipv4 and server-ipv6 keys)

3. Use of DELEG Records

The DELEG RRset MAY contain multiple records. A DELEG RRset MAY be present with or without NS or DS RRsets at the delegation point, though without NS records then DELEG-unaware software will not be able to resolve records in the the delegated zone.

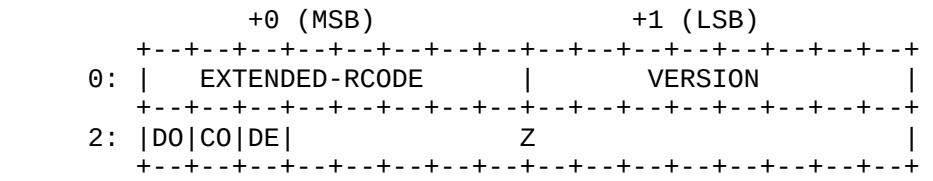
DELEG RRsets MUST NOT appear at a zone’s apex. The erroneous inclusion of DELEG RRset at zone’s apex will cause DNSSEC validation failures. Servers MAY refuse to load such an invalid zone, similar to the DS RR type.

3.1. Resolvers

3.1.1. Signaling DELEG Support

There will be both DELEG and NS needed for delegation for a long time. Both legacy delegation and the DELEG protocol enable recursive resolution. This document defines a new EDNS flag to signal that a resolver is DELEG-aware and therefore does not need NS records or glue information in a referral response.

A resolver that is DELEG-aware MUST signal in queries that it supports the DELEG protocol by setting a bit in the OPT RR TTL as described in [RFC6891]. This bit referred to as the "DELEG" (DE) bit, expected to be assigned by IANA at Bit 2 in the EDNS Header Flags registry, as follows:



Setting the DE bit to one in a query indicates the resolver understands the DELEG semantics and does not need NS records to follow a referral. The DE bit set to 0 indicates the resolver is not DELEG-aware, and therefore can only be served referrals with NS records and other data according to non-DELEG specifications.

3.1.2. Referral

The DELEG record creates a zone cut similar to the NS record.

If one or more DELEG records exist at a given delegation point, a DELEG-aware resolver MUST treat the name servers from those DELEG records as authoritative for the child zone. In such a case, a DELEG-aware resolver MUST NOT use NS records for the zone if they are present, even if resolution using DELEG records has failed. Such fallback from DELEG to NS would invalidate the security guarantees of the DELEG protocol.

If no DELEG record exists at a given delegation point, DELEG-aware resolvers MUST use NS records as specified by [RFC1034]. See Section 3.4 for more information about protection from downgrade attacks.

3.1.3. Parent-side types, QTYPE=DELEG

Record types defined as authoritative on the parent side of zone cut, currently the DS and DELEG types, retain the same special handling as described in Section 2.6 of [RFC4035].

DELEG-unaware resolvers can get different types of answers for QTYPE=DELEG queries based on the configuration of the server, such as whether it is DELEG-aware and whether it also is authoritative for subdomains. For example, a DELEG-unaware authoritative name server which has loaded DELEG records via the [RFC3597] unknown types mechanism would answer with them only if there were no NS records at the owner name, and answer with an NS delegation otherwise.

3.1.4. Algorithm for "Finding the Best Servers to Ask"

This document updates instructions for finding the best servers to ask. That information currently is covered in Section 5.3.3 of [RFC1034] and Section 3.4.1 of [RFC6672] with the text "2. Find the best servers to ask." Section 3.1.4.1 of [RFC4035] should have explicitly updated Section 5.3.3 of [RFC1034] for the DS RR type, but failed to do so. This document simply extends this existing behavior to DELEG RR type as well, and makes this special case explicit.

When a DELEG RRset exists for a delegation in a zone, DELEG-aware resolvers ignore any NS RRset for the delegated zone, whether from the parent or from the apex of the child.

Each delegation level can have a mixture of DELEG and NS RR types, and DELEG-aware resolvers MUST be able to follow chains of delegations which combines both types in arbitrary ways.

An example of a valid delegation tree:

```
; root zone with NS-only delegations
. SOA ...
test. NS ...

; test. zone with NS+DELEG delegations
test. SOA ...
sld.test. NS ...
sld.test. DELEG ...

; sld.test. zone with NS-only delegation
sld.test. SOA ...
nssub.sld.test. NS ...

; nssub.sld.test. zone with DELEG-only delegation
delegsub.sub.sld.test. DELEG ...
```

TODO: after the text below, refer back to this figure and show the order that a DELEG-aware resolver would take when there is a failure to find any good DELEG addresses at sub.sld.test, then any usable name servers at sub.sld.test, and then maybe a good DELEG record at test.

The terms SNAME and SLIST used here are defined in Section 5.3.2 of [RFC1034]:

SNAME is the domain name we are searching for.

SLIST is a structure which describes the name servers and the zone which the resolver is currently trying to query. Neither [RFC1034] nor this document define how a resolver uses SLIST; they only define how to populate it.

A DELEG-aware SLIST needs to be able to hold two types of information, delegations defined by NS records and delegations defined by DELEG records. DELEG and NS delegations can create cyclic dependencies and/or lead to duplicate entries which point to the same server. Resolvers need to enforce suitable limits to prevent damage even if someone has incorrectly configured some of the data used to create an SLIST.

This leads to a modifications of the description from earlier documents for DELEG-aware resolvers can find the best servers to ask. That description becomes:

1. Determine deepest possible zone cut which can potentially hold the answer for a given (query name, type, class) combination:

    1. Start with SNAME equal to QNAME.

    2. If QTYPE is a type that is authoritative at the parent side of a zone cut (currently, DS or DELEG), remove the leftmost label from SNAME. For example, if the QNAME is "test.example." and the QTYPE is DELEG or DS, set SNAME to "example.".

2. Look for locally-available DELEG and NS RRsets, starting at current SNAME.

    1. For a given SNAME, check for the existence of a DELEG RRset. If it exists, the resolver MUST use its content to populate SLIST. However, if the DELEG RRset is known to exist but is unusable (for example, if it is found in DNSSEC BAD cache), the resolver MUST NOT instead use an NS RRset; instead, the resolver MUST treat this case as if no servers were available.

    2. If a given SNAME is proven to not have a DELEG RRset but does have an NS RRset, the resolver MUST copy the NS RRset into SLIST.

3. If SLIST is now populated, stop walking up the DNS tree. However, if SLIST is not populated, remove the leftmost label from SNAME and go back to the first step, using the newly shortened SNAME. Do not go back to the first step if doing so would exceed the amount of work that the resolver is configured to do when processing names; see Section 4.1.

The rest of Step 2's description is not affected by this document.

Resolvers MAY respond to "QNAME=. / QTYPE=DELEG" queries in the same fashion as they respond to "QNAME=. / QTYPE=DS" queries.

3.1.5. Name Server Information for Delegation

The DELEG and DELEGI records have four keys that describe information about name servers. The purpose of this information is to populate the SLIST with IP addresses of the name servers for a zone. The types of information defined in this document are:

* server-ipv4: an unordered collection of IPv4 addresses for name servers

* server-ipv6: an unordered collection of IPv6 addresses for name servers

* server-name: an unordered collection of hostnames of name servers; the addresses must be fetched

* include-delegi: an unordered collection of domain names that point to a DELEGI RRsets, which in turn have more information about the delegation

These keys MUST have a non-empty DelegInfoValue.

The presentation values for server-ipv4 and server-ipv6 are comma-separated list of one or more IP addresses of the appropriate family in standard textual format [RFC5952] [RFC4001]. The wire formats for server-ipv4 and server-ipv6 are a sequence of IP addresses, in network byte order, for the respective address family.

The presentation values for server-name and include-delegi are an unordered collection of fully-qualified domain names and relative domain names, separated by commas. The wire format for server-name and include-delegi are each a concatenated set of a wire-format domain names, where the root label provides the separation between names. The names in the wire format MUST NOT be compressed.

TODO: Describe how escaping works for server-name and include-delegi. This will be the same as the escaping mechanism defined in SVCB.

A DELEG or DELEGI record that has a non-empty DelegInfos MUST have one, and only one, set of server information, chosen from the following:

* one server-ipv4 key
* one server-ipv6 key
* a pair consisting of one server-ipv4 key and one server-ipv6 key
* one server-name key
* one include-delegi key

This restriction only applies to a single DELEG or DELEGI record; a DELEG or DELEGI RRset can have records with different server information keys.

When using server-name, the addresses for all the names in the set must be fetched using normal DNS resolution. This means the names in the value of the server-name key or the include-delegi key MUST NOT be inside the delegated domain.

With this initial DELEG specification, servers are still expected to be reached on the standard DNS port for both UDP and TCP, 53. While a future specification is expected to address other transports using other ports, its eventual semantics are not covered here.

3.1.6. Populating the SLIST from DELEG and DELEGI Records

Each individual DELEG record inside a DELEG RRset, or each individual DELEGI record in a DELEGI RRset, can cause the addition of zero or more entries to SLIST.

A resolver processes each individual DELEG record within a DELEG RRset, or each individual DELEGI record in a DELEGI RRset, using the following steps:

1. If a record has more than one type of server information key (excluding the IPv4/IPV6 case), or has multiple server information keys of the same type, that record is malformed. Stop processing this record.

2. If server-ipv4 and/or server-ipv6 keys are present inside the record, copy all of the address values into SLIST. Stop processing this record.

3. If a server-name key is present in the record, resolve each name in the value into IPv4 and/or IPv6 addresses. Copy these addresses into SLIST. Stop processing this record.

4. If an include-delegi key is present in the record, resolve each name in the value using the DELEGI RR type. Recursively apply the algorithm described in this section, after checking that the maximum loop count described in Section 4.1 has not been reached.

5. If none of the above applies, SLIST is not modified by this particular record.

A DELEG-aware resolver MAY implement lazy filling of SLIST, such as by deferring processing remaining records if SLIST already has what the resolver considers a sufficiently large pool of addresses to contact.

The order in which to try the servers in the final SLIST is outside the scope of this document.

3.2. Authoritative Servers

The DELEG RR type defines a zone cut in similar way as the NS RR type. Behavior defined for zone cuts in existing non-DELEG specifications apply to zone cuts created by the DELEG record. A notable example of this is that the occlusion (usually accidentally) created by NS records in a parent zone would also be created by DELEG records in a parent zone.

DELEG-aware authoritative servers act differently when handling queries from DELEG-unaware clients (those with DE=0) than from DELEG-aware clients (those with DE=1).

The server MUST copy the value of the DE bit from the query into the response, to signal that it is a DELEG-aware server.

3.2.1. DELEG-aware Clients

When the client indicates that it is DELEG-aware by setting DE=1 in the query, DELEG-aware authoritative servers treat DELEG records as zone cuts, and the servers are authoritative on the parent side of the zone cut. This new zone cut has priority over a legacy delegation.

3.2.1.1. DELEG-aware Clients Requesting QTYPE=DELEG

An explicit query for the DELEG RR type at a delegation point behaves much like query for the DS RR type: the server answers authoritatively from the parent zone. All non-DELEG specifications for special handling queries with QTYPE=DS apply equally to QTYPE=DELEG. In summary, the server either provides an authoritative DELEG RRset or declares its non-existence, with relevant DNSSEC proofs when requested and available.

3.2.1.2. Delegation with DELEG

If the delegation has a DELEG RRset, the authoritative server MUST put the DELEG RRset into the Authority section of the referral. In this case, the server MUST NOT include the NS RRset in the Authority section. Include the covering RRSIG following the normal DNSSEC procedure for answers with authoritative zone data.

Similarly, rules for DS RRset inclusion in referrals apply as specified by the DNSSEC protocol.

3.2.1.3. DELEG-aware Clients with NS RRs Present but No DELEG RRs

If the delegation does not have a DELEG RRset, the authoritative server MUST put the NS RRset into the authority section of the referral. The absence of the DELEG RRset MUST be proven as specified by the DNSSEC protocol for authoritative data.

Similarly, rules for DS RRset inclusion into referrals apply as specified by the DNSSEC protocol. Please note, in practice the same process and records are used to prove the non-existence of both DELEG and DS RRsets.

3.2.2. DELEG-unaware Clients

A general principle for DELEG-aware authoritative servers is that they respond to a DELEG-unaware client by following non-DELEG specifications.

DELEG-unaware clients do not recognize DELEG records as a zone cut and are not aware of the special handling rules for DELEG records. They understand a DELEG RRset as an ordinary unknown RR type.

In summary, DELEG records are not returned in referral responses to DELEG-unaware clients, and DELEG-unaware clients do not consider DELEG records authoritative on the parent side of a zone cut.

An authoritative server responding to DELEG-unaware clients has to handle three distinct situations:

* No DELEG RRset is present. In this case, the authoritative server follows the non-DELEG specifications.

* An NS RRset and a DELEG RRset are both present. In this case, the authoritative server uses the NS RRset when constructing referral responses, following the non-DELEG specifications. See also Section 3.3 and Appendix A.

* A DELEG RRset is present, but an NS RRset is not. See Section 3.2.2.1.

3.2.2.1. DELEG-unaware Clients with DELEG RRs Present but No NS RRs

Authoritative servers may receive requests from DELEG-unaware clients for which the child zone is authoritative and is delegated with DELEG RRs only (that is, without any NS RRs). Such a zone is by definition not resolvable for DELEG-unaware clients. From the perspective of a DELEG-unaware client, the zone cut created by the DELEG RRs is invisible. In such a situation, the authoritative server should respnd in a way to limit confusion and/or colateral damage for the DELEG-unaware client.

The authoritative server is RECOMMENDED to supplement its responses to DELEG-unaware resolvers with an [RFC8914] Extended DNS Error using the (IANA-TBD) value "New Delegation Only" from the Extended DNS Error Codes registry.

A DELEG-aware authoritative server implementation has two options:

1. When faced with a client that sent a query with DE=0 that would lead to a delegation, but the zone has no NS records, an authoritative server MAY reply with an RCODE of SERVFAIL and nothing in the Answer and Authority sections.

   This response has negative side effects, namely that most resolvers will then query the remaining authoritative servers to see if any of them would give a different answer. The advantage of this approach is simplicity of implementation and it is easy to understand.

2. Because of the negative side effects of the previous option, an authoritative server SHOULD instead send an answer that accurately describes the situation to a DELEG-unaware resolver.

The NSEC chain and the type bitmap generated according to Section 3.3 leads to exactly one possible answer which is valid according to non-DELEG specifications. See Appendix A for several examples of such answers.

Please note different interpretation of DELEG RR type and zone cut definitions between DELEG-aware authoritative server and DELEG-unaware client. It is confusing.

TODO: List as many of the possible situations that need to be considered for variant 2, and the proposed the single appropriate response for each situation. This list will probably change for the next few iterations of this draft.

3.2.2.2. DELEG-unaware Clients Requesting QTYPE=DELEG

From the perspective of DELEG-unaware clients, the DELEG RR type does not have special semantics and should behave like an old ordinary RR type such as TXT. Thus, queries with DE=0 and QTYPE=DELEG MUST result in a legacy response which can be validated by DELEG-unaware client.

* If there is an NS RRset, this will be a legacy referral to the child zone. From the perspective of a DELEG-unaware client, the DELEG RR is effectively occluded by NS RRset. The DELEG-unaware resolver can then obtain a final answer which can be validated from the child zone in similar fashion as described in [RFC4035] section 3.1.4.1.

* If there is no NS RRset but there is a DELEG RRset, this will be a normal authoritative response with the DELEG RRset, following non-DELEG specifications.

* If there is no NS RRset and no DELEG RRset, this will be a standard negative response following non-DELEG specifications.

TODO: Should we have an example with auth having parent+child zone at the same time, and DE=0 QTYPE=DELEG query?

3.3. DNSSEC Signers

The DELEG record is authoritative on the parent side of a zone cut and needs to be signed as such. Existing rules from the DNSSEC specifications apply.

In summary: for DNSSEC signing, treat the DELEG RR type the same way as the DS RR type.

DELEG RR type defines a zone cut in similar way as NS RR type. This has several consequences which stem from existing non-DELEG specifications:

* All owner names below zone cut are occluded and thus not present in NSEC chains.

* All RRsets which are not permissible at the parent side of zone cut are occluded too and not represented in NSEC chain type bitmap.

See examples in Appendix A.1 and Appendix A.4.2.3.

In order to protect validators from downgrade attacks this draft introduces a new DNSKEY flag ADT (Authoritative Delegation Types). In zones which contain a DELEG RRset, this flag MUST be set to 1 in at least one of the DNSKEY records published in the zone.

3.4. DNSSEC Validators

DELEG awareness introduces additional requirements on validators.

3.4.1. Clarifications on Nonexistence Proofs

This document updates Section 4.1 of [RFC6840] to include "NS or DELEG" types in the type bitmap as indication of a delegation point, and generalizes applicability of ancestor delegation proof to all RR types that are authoritative at the parent (that is, both DS and DELEG). The text in that section is updated as follows:

An "ancestor delegation" NSEC RR (or NSEC3 RR) is one with:

* the NS and/or DELEG bit set,

* the Start of Authority (SOA) bit clear, and

* a signer field that is shorter than the owner name of the NSEC RR, or the original owner name for the NSEC3 RR.

Ancestor delegation NSEC or NSEC3 RRs MUST NOT be used to assume nonexistence of any RRs below that zone cut, which include all RRs at that (original) owner name, other than types authoritative at the parent-side of a zone cut (DS and DELEG), and all RRs below that owner name regardless of type.

3.4.2. Insecure Delegation Proofs

This document updates Section 4.4 of [RFC6840] to include securing DELEG records, and explicitly states that Opt-Out is not applicable to the DELEG protocol. The first paragraph of that section is updated to read:

Section 5.2 of [RFC4035] specifies that a validator, when proving a delegation is not secure, needs to check for the absence of the DS and SOA bits in the NSEC (or NSEC3) type bitmap; this was clarified in Section 4.1 of [RFC6840]. This document updates [RFC4035] and [RFC6840] to specify that the validator MUST also check for the presence of the NS or the DELEG bit in the matching NSEC (or NSEC3) RR (proving that there is, indeed, a delegation). Alternately, the validator must make sure that the delegation with an NS record is covered by an NSEC3 RR with the Opt-Out flag set. Opt-Out is not applicable to DELEG RR type because DELEG records are authoritative at the parent side of a zone cut in the same way that DS RR types are.

3.4.3. Referral downgrade protection

If the zone is DNSSEC-secure, and if any DNSKEY of the zone has the ADT flag set to 1, a DELEG-aware validator MUST prove the absence of a DELEG RRset in referral responses from this particular zone.

Without this check, an attacker could strip the DELEG RRset from a referral response and replace it with an unsigned (and potentially malicious) NS RRset. A referral response with an unsigned NS and signed DS RRsets does not require additional proofs of nonexistence according to non-DELEG DNSSEC specification, and it would have been accepted as a delegation without the DELEG RRset.

3.4.4. Chaining

A Validating Stub Resolver that is DELEG-aware has to use a Security-Aware Resolver that is DELEG-aware and, if it is behind a forwarder, that forwarder has to be security-aware and DELEG-aware as well.

[RFC9606] specifies a DNS resource record type RESINFO to allow resolvers to publish information about their capabilities and policies. This can be used to inform DNS clients that DELEG is supported by the DNS resolver.

A resolver which supports [RFC9606] SHOULD add the "deleg" key if it supports DELEG protocol.

Note that, per the rules for the keys defined in Section 6.4 of [RFC6763], if there is no '=' in a key, then it is a boolean attribute, simply identified as being present, with no value.

4. Security Considerations

TODO: Add more here

4.1. Preventing Over-work Attacks

Resolvers MUST prevent situations where accidental misconfiguration of zones or malicious attacks cause them to perform too much work when resolving. This document describes two sets of actions that, if not controlled, could lead to over-work attacks.

Long chains of include-delegi information (Section 3.1.5), and those with circular chains of include-delegi information, can be burdensome. To prevent this, the resolver SHOULD NOT follow more than 3 include-delegi chains in an RRset when populating SLIST. Note that include-delegi chains can have CNAME steps in them; in such a case, a CNAME step is counted the same as a DELEGI step when determining when to stop following a chain.

4.2. Preventing Downgrade Attacks

TODO: this section is a bit redundant with "Referral Downgrade Protection" above; harmonize them.

During the rollout of the DELEG protocol, the operator of an authoritative server can upgrade the server software to be DELEG-aware before changing any DNS zones. Such deployment should work and provide DELEG-aware clients with correct DELEG-aware answers. However, the deployment will not be protected from downgrade attacks against the DELEG protocol.

To protect DNSSEC-secure DNS zones that use DELEG delegations, the delegating zone needs to have at least one DNSKEY with the ADT flag set to 1. Failure to set this flag in a DNSKEY record in the zone allows an attacker to remove the DELEG RRset from referrals which contain the DS RRset, and replace the original signed DELEG RRset with an arbitrary unsigned NS set. Doing so would be a downgrade from the strong protection offered by DNSSEC for DELEG. That is, the DELEG protocol when used with upgraded DNSKEY records gives the same protection to DELEG that the zone's DS RR set has. Without DELEG, there are no security guarantees for the NS RR set on the parent side of the zone cut.

Please note that a full DNSKEY rollover is not necessary to achieve the downgrade protection for DELEG. Any single DNSKEY with the ADT flag set to 1 is sufficient; the zone can introduce an otherwise unused record into the DNSKEY RRset. This DNSKEY RR can even use an unknown signing algorithm and zero-length key material to minimize size increase of the DNSKEY RRset.

5. IANA Considerations

5.1. Changes to Existing Registries

IANA is requested to allocate the DELEG RR and the DELEGI RR in the Resource Record (RR) TYPEs registry, with the meaning "enhanced delegation information" and referencing this document.

IANA is requested to assign a new bit in the DNSKEY RR Flags registry ([RFC4034]) for the ADT bit (N), with the description "Authoritative Delegation Types" and referencing this document. For compatibility reasons, we request the bit 14 to be used. This value has been proven to work whereas bit 0 was proven to break in practical deployments (because of bugs).

IANA is requested to assign a bit from the EDNS Header Flags registry ([RFC6891]), with the abbreviation DE, the description "DELEG enabled", and referencing this document.

IANA is requested to assign a value from the Extended DNS Error Codes ([RFC8914]), with the Purpose "New Delegation Only" and referencing this document.

IANA is requested to add the name "deleg" to DNS Resolver Information Keys registry ([RFC9606]), with the description of "The presence of the key indicates that DELEG protocol is supported." and referencing this document.

5.2. New Registry for Delegation Information

IANA is requested to create the "DELEG Delegation Information" registry. This registry defines the namespace for delegation information keys, including string representations and numeric key values.

5.2.1. Procedure

A registration MUST include the following fields:

Number: Wire-format numeric identifier (range 0-65535) Name: Unique presentation name Meaning: A short description Reference: Location of specification or registration source Change Controller: Person or entity, with contact information if appropriate

To enable code reuse from SVCB parsers, the requirements for registered Name exactly copy requirements set by [RFC9460] section 14.3.1: The characters in the registered Name field entry MUST be lowercase alphanumeric or "-". The name MUST NOT start with "key" or "invalid".

The registration policy for new entries is Expert Review ([RFC8126]). The designated expert MUST ensure that the reference is stable and publicly available and that it specifies how to convert the delegation information’s presentation format to wire format. The reference MAY be any individual’s Internet-Draft or a document from any other source with similar assurances of stability and availability. An entry MAY specify a reference of the form "Same as (other key name)" if it uses the same presentation and wire formats as an existing key.

This arrangement supports the development of new parameters while ensuring that zone files can be made interoperable.

5.2.2. Initial Contents

The "DELEG Delegation Information" registry should be populated with the following initial registrations:

Number: 1
Name: server-ipv4
Meaning: An unordered collection of IPv4 addresses of name servers
Reference: {{nameserver-info}} of this document
Change Controller: IETF

Number: 2
Name: server-ipv6
Meaning: An unordered collection of IPv6 addresses of name servers
Reference: {{nameserver-info}} of this document
Change Controller: IETF

Number: 3
Name: server-name
Meaning: An unordered collection of hostnames of name servers
Reference: {{nameserver-info}} of this document
Change Controller: IETF

Number: 4
Name: include-delegi
Meaning: An unordered collection of domain names of DELEGI records
Reference: {{nameserver-info}} of this document
Change Controller: IETF

The registration for numbers 65280-65534 is reserved for private use.
The registration for number 65535 is reserved.

5.3. Temporary Assignments

This section gives the values that can be used for interoperability testing before IANA makes permanent assignments. The section will be removed when IANA makes permanent assignments.

* DELEG RR type code is 61440

* DELEGI RR type code is 65433

* DELEG EDNS DE flag bit is 2

* DNSKEY ADT (Authoritative Delegation Types) flag bit is 14

6. References

6.1. Normative References

[RFC1034] Mockapetris, P., "Domain names - concepts and facilities", STD 13, RFC 1034, DOI 10.17487/RFC1034, November 1987, <https://www.rfc-editor.org/rfc/rfc1034>.

[RFC4034] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "Resource Records for the DNS Security Extensions", RFC 4034, DOI 10.17487/RFC4034, March 2005, <https://www.rfc-editor.org/rfc/rfc4034>.

[RFC4035] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "Protocol Modifications for the DNS Security Extensions", RFC 4035, DOI 10.17487/RFC4035, March 2005, <https://www.rfc-editor.org/rfc/rfc4035>.

[RFC6672] Rose, S. and W. Wijngaards, "DNAME Redirection in the DNS", RFC 6672, DOI 10.17487/RFC6672, June 2012, <https://www.rfc-editor.org/rfc/rfc6672>.

[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>.

[RFC6840] Weiler, S., Ed. and D. Blacka, Ed., "Clarifications and Implementation Notes for DNS Security (DNSSEC)", RFC 6840, DOI 10.17487/RFC6840, February 2013, <https://www.rfc-editor.org/rfc/rfc6840>.

[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>.

[RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 8126, DOI 10.17487/RFC8126, June 2017, <https://www.rfc-editor.org/rfc/rfc8126>.

[RFC8914] Kumari, W., Hunt, E., Arends, R., Hardaker, W., and D. Lawrence, "Extended DNS Errors", RFC 8914, DOI 10.17487/RFC8914, October 2020, <https://www.rfc-editor.org/rfc/rfc8914>.

[RFC9460] Schwartz, B., Bishop, M., and E. Nygren, "Service Binding and Parameter Specification via the DNS (SVCB and HTTPS Resource Records)", RFC 9460, DOI 10.17487/RFC9460, November 2023, <https://www.rfc-editor.org/rfc/rfc9460>.

[RFC9606] Reddy.K, T. and M. Boucadair, "DNS Resolver Information", RFC 9606, DOI 10.17487/RFC9606, June 2024, <https://www.rfc-editor.org/rfc/rfc9606>.

6.2. Informative References

[BCP219] Best Current Practice 219,
<https://www.rfc-editor.org/info/bcp219>.
At the time of writing, this BCP comprises the following:

Hoffman, P. and K. Fujiwara, "DNS Terminology", BCP 219,
RFC 9499, DOI 10.17487/RFC9499, March 2024,
<https://www.rfc-editor.org/info/rfc9499>.

[I-D.tapril-ns2]
April, T., "Parameterized Nameserver Delegation with NS2
and NS2T", Work in Progress, Internet-Draft, draft-tapril-
ns2-01, 13 July 2020,
<https://datatracker.ietf.org/doc/html/draft-tapril-
ns2-01>.

[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>.

[RFC3597] Gustafsson, A., "Handling of Unknown DNS Resource Record
(RR) Types", RFC 3597, DOI 10.17487/RFC3597, September
2003, <https://www.rfc-editor.org/rfc/rfc3597>.

[RFC4001] Daniele, M., Haberman, B., Routhier, S., and J.
Schoenwaelder, "Textual Conventions for Internet Network
Addresses", RFC 4001, DOI 10.17487/RFC4001, February 2005,
<https://www.rfc-editor.org/rfc/rfc4001>.

[RFC5952] Kawamura, S. and M. Kawashima, "A Recommendation for IPv6
Address Text Representation", RFC 5952,
DOI 10.17487/RFC5952, August 2010,
<https://www.rfc-editor.org/rfc/rfc5952>.

[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>.

Appendix A. Examples

A.1. Root zone file

The following example shows an excerpt from a signed root zone. It shows the delegation point for "example." and "test."

The "example." delegation has DELEG and NS records. The "test." delegation has DELEG but no NS records.

TODO: Add examples that have server-name and include-delegi being sets of more than one name.

TODO: Examples that show DELEGI records in ns2.example.net and ns3.example.org.

```
example.   DELEG server-ipv4=192.0.2.1 server-ipv6=2001:DB8::1
example.   DELEG server-name=ns2.example.net.,ns3.example.org.
example.   RRSIG DELEG 13 1 300 20260101000000 (
                        20250101000000 33333 . SigExampleDELEG/ )

example.   NS    a.example.
example.   NS    b.example.net.
example.   NS    c.example.org.

example.   DS    44444 13 2 ABCDEF01234567...
example.   RRSIG DS 13 1 300 20260101000000 (
                        20250101000000 33333 . SigExampleDS )

example.   NSEC  net. NS DS RRSIG NSEC DELEG
example.   RRSIG NSEC 13 1 300 20260101000000 (
                        20250101000000 33333 . SigExampleNSEC+/ )

; unsigned glue for legacy (NS) delegation
; it is NOT present in NSEC chain
a.example. A     192.0.2.1
a.example. AAAA  2001:DB8::1
```

The "test." delegation point has a DELEG record and no NS or DS records.

Please note: This is an example of unnecessarily complicated setup to demonstrate capabilities of DELEG and DELEGI RR types.

```
test.      DELEG server-ipv6=3fff::33
test.      DELEG include-delegi=Acfg.example.org.
test.      DELEG include-delegi=config2.example.net.
test.      RRSIG DELEG 13 1 300 20260101000000 (
                        20250101000000 33333 . SigTestDELEG )

test.      NSEC  . RRSIG NSEC DELEG
test.      RRSIG NSEC 13 1 300 20260101000000 (
                        20250101000000 33333 . SigTestNSEC/ )

; a forgotten glue from legacy (NS) delegation
; it is NOT present in NSEC chain and it is occluded
a.test.    A     192.0.2.1
```

Delegations to org and net zones omitted for brevity.

A.2. Example.org zone file

The following example shows an excerpt from an unsigned example.org zone.

```
Acfg.example.org.    DELEGI server-ipv6=2001:DB8::6666
Acfg.example.org.    DELEGI server-name=c.example.org.
Acfg.example.org.    DELEGI include-delegi=subcfg.example.org.

c.example.org.       AAAA   3fff::33

subcfg.example.org.  DELEGI server-ipv4=203.0.113.1 server-ipv6=3fff::2
```

A.3. Example.net zone file

The following example shows an excerpt from an unsigned example.net zone.

```
b.example.net.       A      198.51.100.1

config2.example.net. DELEGI server-name=b.example.org.
```

A.4. Responses

The following sections show referral examples:

A.4.1. DO bit clear, DE bit clear

A.4.1.1. Query for foo.example

```
   ;; Header: QR RCODE=NOERROR
   ;;

   ;; Question
   foo.example.  IN MX

   ;; Answer
   ;; (empty)

   ;; Authority
   example.   NS    a.example.
   example.   NS    b.example.net.
   example.   NS    c.example.org.

   ;; Additional
   a.example. A     192.0.2.1
   a.example. AAAA  2001:DB8::1
```

A.4.1.2. Query for foo.test

```
   ;; Header: QR AA RCODE=NXDOMAIN
   ;;

   ;; Question
   foo.test.   IN MX

   ;; Answer
   ;; (empty)

   ;; Authority
   .   SOA ...

   ;; Additional
   ;; OPT with Extended DNS Error: New Delegation Only
```

A.4.1.3. Query for a.test

A forgotten glue record under the "test." delegation point is occluded by DELEG RRset.

```
   ;; Header: QR AA RCODE=NXDOMAIN
   ;;

   ;; Question
   a.test.   IN A

   ;; Answer
   ;; (empty)

   ;; Authority
   .   SOA ...

   ;; Additional
   ;; OPT with Extended DNS Error: New Delegation Only
```

A.4.2. DO bit set, DE bit clear

A.4.2.1. Query for foo.example

```
   ;; Header: QR DO RCODE=NOERROR
   ;;

   ;; Question
   foo.example.   IN MX

   ;; Answer
   ;; (empty)

   ;; Authority

   example.   NS    a.example.
   example.   NS    b.example.net.
   example.   NS    c.example.org.
   example.   DS    44444 13 2 ABCDEF01234567...
   example.   RRSIG DS 13 1 300 20260101000000 (
                           20250101000000 33333 . SigExampleDS )
   ;; Additional
   a.example. A     192.0.2.1
   a.example. AAAA  2001:DB8::1
```

A.4.2.2. Query for foo.test

```
;; Header: QR DO AA RCODE=NXDOMAIN
;;

;; Question
foo.test.      IN MX

;; Answer
;; (empty)

;; Authority
.          SOA ...
.          RRSIG SOA ...
test.      NSEC  . RRSIG NSEC DELEG
test.      RRSIG NSEC 13 1 300 20260101000000 (
                        20250101000000 33333 . SigTestNSEC/ )

;; Additional
;; OPT with Extended DNS Error: New Delegation Only
```

A.4.2.3. Query for a.test

A forgotten glue record under the "test." delegation point is occluded by DELEG RRset. This is indicated by NSEC chain which "skips" over the owner name with A RRset.

```
;; Header: QR DO AA RCODE=NXDOMAIN
;;

;; Question
a.test.      IN A

;; Answer
;; (empty)

;; Authority
.          SOA ...
.          RRSIG SOA ...
test.      NSEC  . RRSIG NSEC DELEG
test.      RRSIG NSEC 13 1 300 20260101000000 (
                        20250101000000 33333 . SigTestNSEC/ )

;; Additional
;; OPT with Extended DNS Error: New Delegation Only
```

A.4.3. DO bit clear, DE bit set

A.4.3.1. Query for foo.example

```
;; Header: QR DE RCODE=NOERROR
;;

;; Question
foo.example.  IN MX

;; Answer
;; (empty)

;; Authority
example.   DELEG server-ipv4=192.0.2.1 server-ipv6=2001:DB8::1
example.   DELEG server-name=ns2.example.net.,ns3.example.org.

;; Additional
;; (empty)
```

A.4.3.2. Query for foo.test

```
;; Header: QR AA RCODE=NOERROR
;;

;; Question
foo.test.   IN MX

;; Answer
;; (empty)

;; Authority
test.      DELEG server-ipv6=3fff::33
test.      DELEG include-delegi=Acfg.example.org.
test.      DELEG include-delegi=config2.example.net.

;; Additional
;; (empty)
```

Follow-up example in Appendix A.5 explains ultimate meaning of this response.

A.4.4. DO bit set, DE bit set

A.4.4.1. Query for foo.example

```
   ;; Header: QR DO DE RCODE=NOERROR
   ;;

   ;; Question
   foo.example.  IN MX

   ;; Answer
   ;; (empty)

   ;; Authority
   example.   DELEG server-ipv4=192.0.2.1 server-ipv6=2001:DB8::1
   example.   DELEG server-name=ns2.example.net.,ns3.example.org.
   example.   RRSIG DELEG 13 1 300 20260101000000 (
                           20250101000000 33333 . SigExampleDELEG/ )
   example.   DS    44444 13 2 ABCDEF01234567...
   example.   RRSIG DS 13 1 300 20260101000000 (
                           20250101000000 33333 . SigExampleDS )

   ;; Additional
   a.example. A     192.0.2.1
   a.example. AAAA  2001:DB8::1
```

A.4.4.2. Query for foo.test

```
   ;; Header: QR DO DE AA RCODE=NOERROR
   ;;

   ;; Question
   foo.test.      IN MX

   ;; Answer
   ;; (empty)

   ;; Authority
   test.      DELEG server-ipv6=3fff::33
   test.      DELEG include-delegi=Acfg.example.org.
   test.      DELEG include-delegi=config2.example.net.
   test.      RRSIG DELEG 13 1 300 20260101000000 (
                           20250101000000 33333 . SigTestDELEG )
   test.      NSEC  . RRSIG NSEC DELEG
   test.      RRSIG NSEC 13 1 300 20260101000000 (
                           20250101000000 33333 . SigTestNSEC/ )

   ;; Additional
   ;; (empty)
```

Follow-up example in Appendix A.5 explains the ultimate meaning of this response.

A.5. DELEGI Interpretation

In the examples above, the test. DELEG record uses indirection and points to other domain names with DELEGI, A, and AAAA records. During resolution, a resolver will gradually build set of name servers to contact, as defined in Section 3.1.6.

To visualize end result of this process we represent full set of name servers in form of a 'virtual' DELEG RRset.

```
test. DELEG server-ipv4=198.51.100.1
test. DELEG server-ipv4=203.0.113.1
test. DELEG server-ipv6=2001:DB8::6666
test. DELEG server-ipv6=3fff::2
; IPv6 address 3fff::33 was de-duplicated (input RRsets listed it twice)
test. DELEG server-ipv6=3fff::33
```

Implementations are free to use arbitrary representation for this data as it is not directly exposed via DNS protocol.

Acknowledgments

This document is heavily based on past work done by Tim April in [I-D.tapril-ns2] and thus extends the thanks to the people helping on this which are: John Levine, Erik Nygren, Jon Reed, Ben Kaduk, Mashooq Muhaimen, Jason Moreau, Jerrod Wiesman, Billy Tiemann, Gordon Marx and Brian Wellington.

Work on DELEG protocol has started at IETF 118 Hackaton. Hackaton participants: Christian Elmerot, David Blacka, David Lawrence, Edward Lewis, Erik Nygren, George Michaelson, Jan Velk, Klaus Darilion, Libor Peltan, Manu Bretelle, Peter van Dijk, Petr paek, Philip Homburg, Ralf Weber, Roy Arends, Shane Kerr, Shumon Huque, Vandan Adhvaryu, Vladimr unt, Andreas Schulze.

Other people joined the effort after the initial hackaton: Ben Schwartz, Bob Halley, Paul Hoffman, Miek Gieben ...

RESINFO extension was contributed by Florian Obser.

Authors' Addresses

Tim April
Google, LLC
Email: ietf@tapril.net

Petr paek
ISC
Email: pspacek@isc.org

Ralf Weber
Akamai Technologies
Email: rweber@akamai.com

David C Lawrence
Salesforce
Email: tale@dd.org