deleg
Internet-Draft
Updates: 1034, 1035, 4035, 6672, 6840 (if approved)
Intended status: Standards Track
Expires: 6 December 2026

P. paek
ISC
R. Weber
Akamai Technologies
D. Lawrence
Salesforce

4 June 2026

Extensible Delegation for DNS
draft-ietf-deleg-09

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 DELEGPARAM 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

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 at the apex of the delegated zone, which might not match the NS records at the delegation. The DNS Security Extensions (DNSSEC) protect only one copy, those in the apex.

These properties of NS records limit resolvers to unencrypted messages on UDP and TCP port 53, and this initial contact cannot be protected with DNSSEC. These limitations are a barrier for the efficient introduction of new DNS technology.

The proposed DELEG and DELEGPARAM resource record (RR) types remedy this problem by providing extensible parameters to indicate authoritative name 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 at the delegation point and thus can be signed with DNSSEC. This makes it possible to validate all delegation parameters, including those of future extensions.

The DELEGPARAM record is an auxiliary record which does not create a delegation provides an optional layer of indirection. It can be used to share the same delegation information across any number of zones, simplifying operations management by reducing the number of situations for which the delegation information for a domain would need to be changed at the delegation point. For example, if the customers of a DNS operator point their delegations to a DELEGPARAM record managed by the DNS operator, then the operator can make changes without requiring the customers to have to update the delegation point.

The DELEG record can be used alongside, or even instead of, an 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.

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 additional terms defined here:

* legacy delegation: A delegation that is done with an NS RRset
* DELEG-aware: A DNS software that follows the protocol defined in this document
* DELEG-unaware: A DNS software 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
* Delegation NS RRset: An NS RRset that delegates authority of subdomain to a set of authoritative servers
* Authoritative NS RRset: An NS RRset at the apex of a zone.

2. Protocol Overview

This section is a brief overview of the protocol. It is meant for people who want to understand the protocol before they dive deeper into the specifics.

When a DELEG-aware resolver sends queries, it sets the DE bit in the EDNS0 header to 1 in queries to authoritative servers, as a signal that it is DELEG-aware (Section 4).

DELEG-unaware authoritative servers intrinsically ignore this signal.

A DELEG-aware authoritative server uses that signal to determine the type of response it will send. If the response is not a referral, the authoritative server doesn’t change anything about how it responds (Section 5.2.1.3). If the response is a referral, the authoritative server checks if there is a DELEG RRset for the queried zone. If so, it returns the DELEG RRset instead of any NS RRset in the response (Section 5.2.1).

Records in the DELEG RRset for a zone describe how to find name servers for that zone (Section 3). The RDATA for DELEG records has key=value pairs (Section 3.4).

* "server-ipv4" and "server-ipv6" keys contain one or more IP addresses for the delegated name servers

* "server-name" key contains one or more hostnames for the delegated name servers; the addresses must be fetched separately

* "include-delegparam" key contains one or more domain names which in turn have more information about the delegation

* "mandatory" key contains a list of other keys which must be present in the same record, and which the resolver must understand in order to use that record

The DELEG-aware resolver uses the information in the DELEG RRset to form the list of best servers to ask about the original zone (Section 5.1.3). If the DELEG RRset contains "include-delegparam", the resolver queries those hostnames for DELEGPARAM RRsets. DELEGPARAM records have the same format as DELEG records; thus, they can have the same key=value pairs.

The DELEG protocol changes how zones are signed (Section 5.3) and validated (Section 5.4). The changes are primarily because DELEG RRsets are authoritative at the delegation point and thus are signed and validated as authoritative data, similar to DS records.

A zone might be delegated with only DELEG records but no NS records. Such a zone would be invisible to DELEG-unaware resolvers.

In order to protect validators from downgrade attacks, this document introduces a new DNSKEY flag called ADT (Authoritative Delegation Types), described in Section 5.4.3.

There are many parts of the DELEG protocol that are not included in this brief overview. For example, DELEG-aware authoritative servers have choices to make depending both on the request and the contents of the zone file. For those readers who learn better from examples than the definitive text, see Appendix A.

3. DELEG and DELEGPARAM Resource Record Types

The DELEG record, RR type TBD, and the DELEGPARAM 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 also 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 are 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 a 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 the SVCB/HTTPS and DELEG/DELEGPARAM record families.

The initial set of DelegInfoKeys and their formats are defined in Section 3.4.

3.1. Presentation Format

The RDATA presentation format of the DELEG and DELEGPARAM 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.

- The 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.

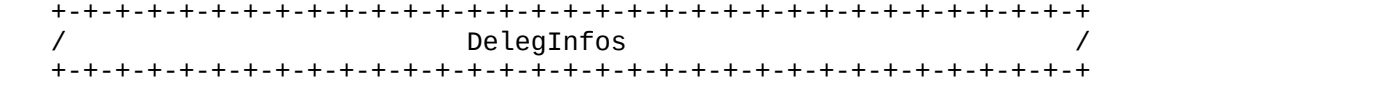
- The rules from Appendix A of [RFC9460] apply.

* 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.

3.2. RDATA Wire Format

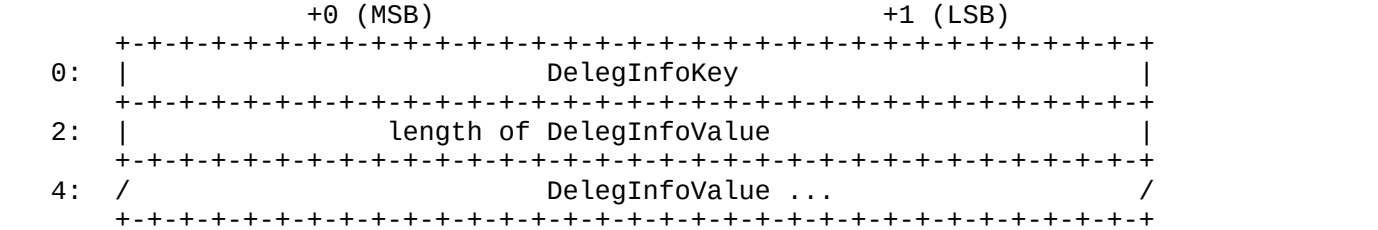
The RDATA portion of the DELEG and DELEGPARAM resource record is variable length and entirely consists of a single "DelegInfos" element:



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 and MAY be empty. 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.

3.3. Semantics

The following is a brief summary of semantic differences between the DELEG and DELEGPARAM 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 at the delegation point, similar to the DS RR type, and unlike the NS RR type.

* DELEG is signed 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,

* DELEGPARAM is an ordinary RR and doesn’t require any special processing.

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

* DELEGPARAM cannot exist at a delegation point.

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

* DELEGPARAM is used as the target of the DELEG protocol’s "include-delegparam" mechanism, as described in section Section 5.1.4.

Note that neither DELEG nor DELEGPARAM trigger Additional Section processing like NS does. The significance of this difference is addressed more in the next section.

3.4. Name Server Information for Delegation

The DELEG and DELEGPARAM records have four keys that describe information about name servers. The purpose of this information is to populate the SLIST (see Section 5.1.4) 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 separately

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

These keys MUST have a non-empty DelegInfoValue.

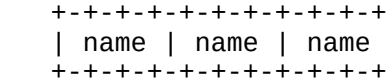
The presentation values for server-ipv4 and server-ipv6 are comma-separated lists of one or more IP addresses of the appropriate family in standard textual format [RFC5952] [RFC4001]. The wire formats for server-ipv4 and server-ipv6 are a sequence of IP addresses, in network byte order, for the respective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

The presentation values for server-name and include-delegparam are an unordered collection of fully-qualified domain names and relative domain names, separated by commas. Relative names in the presentation format are interpreted according to the origin rules in Section 5.1 of [RFC1035]. Parsing the comma-separated list is specified in Section A.1 of [RFC9460].

The DELEG protocol allows the use of all valid domain names, as defined in [RFC1035] and Section 11 of [RFC2181]. The presentation format for names with special characters requires both double-escaping by applying rules of Section 5.1 of [RFC1034] together with the escaping rules from Section A.1 of [RFC9460].

TODO: add an example that requires this escaping.

The wire format for server-name and include-delegparam are each a concatenated unordered collection of wire-format domain names, where the root label provides the separation between names:



The names in the wire format MUST NOT be compressed, per [RFC3597].

For interoperability with the resolver algorithm defined in section Section 5.1.4, a DELEG or DELEGPARAM record that has a non-empty DelegInfos MUST have one, and only one, set of server information keys, 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-delegparam key

This restriction only applies to a single DELEG or DELEGPARAM record; a DELEG or DELEGPARAM RRset can have records with different server information keys. Authoritative servers MAY refuse to load zones which have a disallowed combination of keys in a single record.

When using server-name or include-delegparam, the addresses for the names in the set must be fetched as if they were referenced by NS records. Because of the lack of Additional Section processing, there are no "glue" records provided for these names, so they cannot be for names 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.

3.5. Metadata keys

This specification defines a key which serves as a protocol extensibility mechanism, but is not directly used for contacting DNS servers.

Any DELEG or DELEGPARAM record can have key named "mandatory" which is similar to the key of the same name in [RFC9460].

The presentation format for the value MUST be a comma-separated list of one or more valid DelegInfoKeys, either by their registered name or in the unknown-key format.

The wire format for the value is a sequence of DelegInfoKey numeric values in network byte order, concatenated, in strictly increasing numeric order.

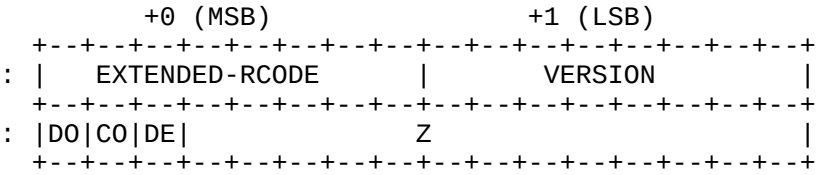
The "mandatory" key is optional, but when it is present, the RR in which it appears MUST also contain all of the DelegInfoKeys referenced in its DelegInfoValue. Resolvers MUST handle non-compliant RRs as specified in Section 5.1.4.

A resolver MUST NOT use an RR with a "mandatory" key in the resolution process unless all of the DelegInfoKeys referenced by the "mandatory" DelegInfoValue are supported in the resolver's implementation. See Section 5.1.4.

4. Signaling DELEG Support

This document defines a new EDNS flag to signal that an initiator and responder are DELEG-aware.

This flag is referred to as the "DELEG" (DE) bit, expected to be assigned by IANA as Bit 2 in the EDNS Header Flags registry. It is part of OPT RR TTL as described in [RFC6891], as follows:



If a query has the DE bit set to 1, and the responder is DELEG-aware, the responder MUST set the DE bit in the response to 1, independent of whether the response includes any DELEG or DELEGPARAM records.

5. 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.

Both the DELEG protocol and legacy delegations (that is, NS records) will be used for delegation for a long time. Both legacy delegations and the DELEG protocol enable recursive resolution. A DELEG-aware resolver therefore does not need the NS records or glue information in a DELEG referral response, and MUST NOT get them; see Section 7.2.

5.1. Resolvers

A resolver that is DELEG-aware MUST signal in queries that it supports the DELEG protocol by setting the DE bit to 1 in (see Section 4). This indicates that 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. Other special scenarios with DE=0 queries to DELEG-aware authorities are addressed in Section 5.2.

5.1.1. Referral

The DELEG record creates a delegation point 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 delegated zone. In such a case, a DELEG-aware resolver MUST NOT use NS records for the zone if they are learned, even if resolution using DELEG records has failed. Such fallback from DELEG to NS would invalidate the security guarantees of the DELEG protocol; see Section 7.2.

If no DELEG record exists at a given delegation point, DELEG-aware resolvers MUST use NS records as specified by [RFC1034].

5.1.2. Delegation point types, QTYPE=DELEG

Record types defined as authoritative at a delegation point, 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. See Section 5.2.2.2 for more information.

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

This document updates instructions for finding the best servers to ask. It was 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.". These instructions were informally updated by section 4.2 of [RFC4035] for the DS RR type but the algorithm change was not made explicit. This document simply extends this existing behavior from the DS RR type to the 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 delegation point or from the apex of the delegated zone.

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]. Quote:

* 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.

This document defines SLIST to be a set. Each individual value MUST be represented only once in the final SLIST even if it was encountered multiple times during SLIST construction.

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 SHOULD enforce suitable limits to prevent runaway processing even if someone has incorrectly configured some of the data used to create an SLIST; this is the same recommendation to bound the amount of work as is made in Section 5.3.3 of [RFC1034].

Step 2 of Section 5.3.3 of [RFC1034] is just "2. Find the best servers to ask." For DELEG-aware resolvers, this description becomes:

=====

2. Find the best servers to ask:

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

2.1.1. Start with SNAME equal to QNAME.

2.1.2. If QTYPE is a type that is authoritative at the delegation point (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.".

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

2.2.1. 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, or content of individual RRs is unusable for any reason), the resolver MUST NOT instead use an NS RRset; instead, the resolver MUST treat this case as if SLIST is populated with unreachable servers.

2.2.2. 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.

2.2.3. If SLIST is now populated, stop walking up the DNS tree.

2.2.4. However, if SLIST is not populated, remove the leftmost label from SNAME and go back to step 2.2, using the newly shortened SNAME.

=====

The rest of Step 2's description in Section 5.3.3 of [RFC1034] is not affected by this document.

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

5.1.4. Populating the SLIST from DELEG and DELEGPARAM Records

Each individual DELEG record inside a DELEG RRset, or each individual DELEGPARAM record in a DELEGPARAM 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 DELEGPARAM record in a DELEGPARAM RRset, using the following steps:

1. Discard all DelegInfo elements with DelegInfoKey values that are not supported by the resolver implementation. If no DelegInfo elements remain after this filtering, stop processing the record. Otherwise, continue using only the supported DelegInfo elements.

2. If a DelegInfo element with the "mandatory" DelegInfoKey is present, check its DelegInfoValue. The DelegInfoValue is a list of keys which MUST have corresponding DelegInfo elements in the record after the filtering in the previous steps. If any of the listed DelegInfo elements is not present, stop processing this record.

3. If a record has more than one type of server information key (excluding the IPv4/IPv6 case, see Section 3.4), or if it has multiple server information keys of the same type, that record is malformed. Stop processing this record.

4. If any DNS name referenced by server-name key or the include-delegparam key is equal to or is a subdomain of the delegated domain (i.e. the DELEG record owner), that record is malformed. Stop processing this record.

   This check MUST be performed against the original owner name of the DELEG record even if the currently-processed record is a DELEGPARAM record that was included by the original DELEG record. The purpose of this check is to ensure deterministic behavior. Not performing this check would allow delegations to be reachable only with certain cache content and/or a specific algorithm for server selection from SLIST.

5. 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.

6. 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.

7. If an include-delegparam key is present in the record, resolve each name in the value using the DELEGPARAM RR type. Recursively apply the algorithm described in this section, after checking that the maximum loop count described in Section 7.1 has not been reached.

8. 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 of remaining records, or even individual names or query types, 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.

5.2. 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 delegation NS records would also be created by DELEG records at a delegation point (see Appendix A.4.1.3). Rules for setting Authoritative Answer (AA) bit in answers also remain unchanged: the DELEG RR type has the same special treatment as DS RR

type.

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). See Section 4 and Section 5.1.

5.2.1. 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 delegations, and the servers are authoritative. This new zone cut has priority over a legacy delegation.

5.2.1.1. DELEG-aware Clients Requesting QTYPE=DELEG

An explicit query for the DELEG RR type at a delegation point behaves much like a query for the DS RR type: the server answers authoritatively from the delegating zone. All non-DELEG specifications for the special handling of 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.

5.2.1.2. 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.

Non-DELEG DNSSEC specifications for RRSIG inclusion in answers with authoritative RRsets ({!RFC4035} section 3.1.1) MUST be followed. Similarly, rules for DS RRset inclusion in referrals apply as specified by the DNSSEC protocol.

5.2.1.3. 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.

5.2.2. 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 delegation point 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 at a delegation point.

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 5.3 and Appendix A.

* A DELEG RRset is present, but an NS RRset is not. This is addressed in the next section.

5.2.2.1. 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. The authoritative server should respond in a way that makes sense to DELEG-unaware clients.

The current, primary use case for zone owners that have zones to have DELEG records but no NS records is that they want resolution of those zones only if the resolver uses future features of the DELEG protocol, such as encrypted DNS transports.

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.

When there is no NS records for a delegated zone, a DELEG-aware authoritative server MUST respond to DELEG-unaware clients with an answer that accurately describes the situation to a DELEG-unaware resolver. For a query of the delegated zone itself, the response has an RCODE of NOERROR; for a query that has more labels than the delegated zone, the response has an RCODE of NXDOMAIN; this is no different than what is already specified by algorithms in [RFC1034] and subsequent updates. NSEC and DS records are returned following the existing rules in [RFC4035].

5.2.2.2. 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 response which can be validated by a DELEG-unaware client.

* If there is an NS RRset, this will be a legacy referral. 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 delegated 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? What about QTYPE=ANY?

5.3. DNSSEC Signers

The DELEG record is authoritative at the delegation point 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.

The DELEG RR type defines a zone cut in similar way as the 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 delegation point 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 (see Section 7.2) this draft introduces a new DNSKEY flag ADT (Authoritative Delegation Types, see Section 5.4.3). To achieve downgrade resistance, DNSSEC-signed zones which contain a DELEG RRset MUST set ADT flag to 1 in at least one of the DNSKEY records published in the zone.

5.4. DNSSEC Validators

DELEG awareness introduces additional requirements on validators.

5.4.1. 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 a delegation point (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 delegation point (DS and DELEG), and all RRs below that owner name regardless of type.

5.4.2. 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 delegation point in the same way that DS RR types are.

5.4.3. Referral downgrade protection

If the zone is DNSSEC-secure, and if any DNSKEY of the zone has the ADT flag (Section 5.4.3) 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 (Section 7.2). The reason for this is that according to non-DELEG DNSSEC specification, a referral response with an unsigned NS and signed DS RRsets does not require additional proofs of nonexistence.

5.4.4. Positive responses

An existing DELEG RRset is authoritative in, and signed by, the delegating zone, similarly to a DS RRset (see Section 5.3).

A validator SHOULD NOT treat a positive response with a DELEG RRset as DNSSEC-bogus only because all DNSKEYs in the zone have the ADT flag set to 0. Such a zone would not be protected from downgrade attacks (Section 7.2) but this behavior is consistent with other non-DELEG DNSSEC specifications: validators are not expected to detect inconsistencies in data if a chain of trust can be established.

5.4.5. Chaining

A Validating Stub Resolver that is DELEG-aware MUST only use security-aware resolvers that are DELEG-aware. A DELEG-aware Validating Resolver that uses forwarders MUST only use DELEG-aware and security-aware forwarders. Otherwise DNSSEC-secure zones might fail to validate (see Appendix A.4.2.2) and DNSSEC-insecure zones might observe inconsistent answers (see Section 6).

[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. A resolver that uses forwarders MAY use a RESINFO query to determine if the configured forwarders are DELEG-aware.

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.

6. Operational Considerations

When DELEG is deployed, new operational considerations will apply. While the majority of these relate to the operation of DELEG-aware servers or resolvers, there is a more general set of operational practices which will need to apply because not all resolvers will be DELEG-aware. This section gives an overview of some of those considerations.

6.1. NS Not Required by Protocol

A zone delegated exclusively using DELEG records is not resolvable by non-DELEG aware resolvers. In that case the zone is not required to have NS RRset in the apex of the delegated zone. Software to manage zone content or check the validity of zones needs to be updated to allow zones without an NS RRset at the apex.

6.2. NS Maybe Required in Practice

Although DELEG removes the protocol requirement for NS records, resolver support for DELEG will not be universal for a long time after this protocol is first deployed. The deployment of DELEG-only delegation creates a new situation in which DNS servers that are authoritative for a particular set of domains provide partly or completely different answers. Where "split DNS" or "split-horizon DNS" RFC9499 differences depend on the source of the query, resolution of DELEG-only delegations will depend on whether or not the resolver is aware of and using DELEG. Compare examples of DELEG-only delegation and respective answers for DELEG-unaware client in Appendix A.4.2.2 and DELEG-aware client in Appendix A.4.4.2.

For any part of the namespace that is intended to be globally reachable, operators should avoid DELEG-only delegations, as some resolvers will be unaware of DELEG. For other parts of the namespace, operators should take care to ensure that any variability in responses introduced maps correctly to the client capabilities.

DELEG-only delegation is appropriate only where all intended users are known to use DELEG-capable resolvers. This might be the case when a zone operator wants a zone be reachable only over secure transport, for example. The decision to drop NS records should be guided by operational measurements of resolver adoption of the DELEG protocol.

6.3. NS and DELEG Combined

This document explicitly allows zones to be delegated using DELEG records without also using NS records; delegating a zone with both DELEG and NS records is also allowed. Software that manages delegations or checks the validity of zones need to be updated to allow delegations with all combinations of (with, without) * (NS, DELEG) records.

If both NS and DELEG records are present, zone managers might want to check consistency across both RRsets, subject to local policy. This specification treats both NS and DELEG RRsets as completely independent on the protocol level, but it does not prohibit a provisioning system from generating one record type from the other.

6.4. Authoritative Deployment

Before adding a first DELEG record into a DNS zone, these steps need to be taken, in this order:

1. If zone checkers are used: ensure that the zone checkers are DELEG-aware.

2. Ensure that all authoritative servers serving (and transfering) the zone are DELEG-aware.

3. If a zone is DNSSEC-signed: ensure that the signer is DELEG-aware.

4. If a zone is DNSSEC-signed: ensure that at least one DNSKEY record has the ADT flag set to 1. Failure to do so results in loss of downgrade resistence of the DELEG protocol for this zone; see Section 7.2.

6.4.1. Enabling ADT Flag

According to the DNSSEC specification, changing flags of a DNSKEY record changes its Key Tag and thus requires a key rollover. For this reason, the ADT flag cannot be simply enabled on an existing key without other changes to the record. Operators are advised to set the ADT flag at the time of generating a new key, as part of a regular key rollover using established procedures. A zone can safely have keys with the ADT flag set to 1 even if the zone does not have any DELEG records. Turning on the ADT flag can be done months or even years before a first DELEG record is introduced into the zone.

Downgrade protection is effective if any DNSKEY with ADT flag set to 1 is present, even if this key does not sign any RRset. In other words, it is sufficient to pre-publish new key, as described in stage 2 of Pre-Publish Zone Signing Key Rollover, section 4.1.1.1 of [RFC6781].

An extremely conservative approach might be:

1. Lower DNSKEY TTL to shorten time to rollback.
2. Add a new DNSKEY with ADT flag set to 1, but do not sign any RRsets with this key.
3. Monitor deployment for issues.
4. Experiment with adding DELEG records at this point, even if the key rollover is not finished. If there is a problem, withdraw the new, otherwise unused key.
5. Finish the key rollover.
6. Restore the original DNSKEY TTL.

Such an approach requires changing only to DNSKEY RRset and resigning it. Consequently, the time required to withdraw the new DNSKEY record is limited only by DNSKEY TTL + time to sign + time to transfer modified DNSKEY RRset.

6.5. Interaction with Dynamic DNS Upate ([RFC2136])

DELEG records can be updated like other regular zone data at the delegation point, similar to NS records, as dynamic updates work on zone data and not queries. A DELEG-only delegation would not need an NS record in the delegated zone, but the NS record in the delegated zone can not be deleted because of section RFC2136 section 7.13. This should cause no immediate problems as dynamic DNS updates with

DELEG are most useful at the delegation point. DELEGPARAM will be handled like any other record.

7. Security Considerations

TODO: More people should check this section is complete!

7.1. 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-delegparam information (Section 3.4), and those with circular chains of include-delegparam information, can be burdensome. To prevent this, the resolver SHOULD NOT follow more than 3 include-delegparam chains in an RRset when populating SLIST. Note that include-delegparam chains can have CNAME steps in them; in such a case, a CNAME step is counted the same as a DELEGPARAM step when determining when to stop following a chain.

TODO: No other DNS spec specifies hard maximum number of indirections. Perhaps we should not specify it either? After all DELEG value can contain a name in NS-only delegation and then we get into realm of 'count DELEG but NS is uncounted' and other fun combinations like this. Perhaps this is better dealt with for the whole DNS protocol within draft-fujiwara-dnsop-dns-upper-limit-values?

7.2. Preventing Downgrade Attacks

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 contain 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 RRset has. Without DELEG, there are no security guarantees for delegation NS Records.

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.

7.3. DELEG Is Stronger Than NS

DELEG RRtype has stronger protection (by DNSSEC) than delegation NS records and glue records. A zone that does not need to be resolvable by DELEG-unaware clients (see {operational-considerations}), and is delegated only with DELEG records, will have a smaller attack surface compared to a zone delegated with both DELEG and NS records.

The additional attack surface of legacy delegations stems from the possibility of replacing NS and glue records in referrals with arbitrary values, which is not detectable by DNSSEC (by design in [RFC4035] Section 2.2).

For example, this allows redirecting a referral to names and/or addresses under an attacker's control. Even for DNSSEC-secure zones, an attacker can use this ability to continuously proxy queries and responses, observe traffic, and also monitor the network addresses involved, which might be a privacy concern for roaming clients.

The feasibility and impact of such attacks depend on the threat model, which is outside the scope of this document.

8. IANA Considerations

8.1. Changes to Existing Registries

All new allocations should reference this document.

IANA is requested to assign two types in the Resource Record (RR) TYPEs registry ([RFC6895]):

* TYPE DELEG, Meaning "Extensible Delegation", Value equal to 61440.

* TYPE DELEGPARAM, Meaning "Extensible Delegation Indirection", Value TBA1 inside one of the ranges marked as "data TYPEs".

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

IANA is requested to assign a bit in the EDNS Header Flags registry ([RFC6891]): Bit TBA2, Flag DE, with the description "DELEG enabled".

IANA is requested to assign a value in the Extended DNS Error Codes registry ([RFC8914]): INFO-CODE TBA3, with the Purpose "New Delegation Only".

IANA is requested to create this assignment in the DNS Resolver Information Keys registry ([RFC9606]): Name "deleg", Description "The presence of the key indicates that DELEG protocol is supported."

8.2. 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.

8.2.1. 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.

8.2.2. Initial Contents

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

Number: 0
Name: mandatory
Meaning: Mandatory keys in this RR
Reference: {{mandatory}} of this document
Change Controller: IETF

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 domain names of name servers
Reference: {{nameserver-info}} of this document
Change Controller: IETF

Number: 4
Name: include-delegparam
Meaning: An unordered collection of domain names of DELEGPARAM 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.

8.3. 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
* DELEGPARAM RR type code is 65433
* DELEG EDNS DE flag bit is 2
* DNSKEY ADT (Authoritative Delegation Types) flag bit is 14

9. References

9.1. 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>.

[RFC1035] Mockapetris, P., "Domain names - implementation and specification", STD 13, RFC 1035, DOI 10.17487/RFC1035, November 1987, <https://www.rfc-editor.org/rfc/rfc1035>.

[RFC2136] Vixie, P., Ed., Thomson, S., Rekhter, Y., and J. Bound, "Dynamic Updates in the Domain Name System (DNS UPDATE)", RFC 2136, DOI 10.17487/RFC2136, April 1997, <https://www.rfc-editor.org/rfc/rfc2136>.

[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>.

[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>.

[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>.

[RFC6781] Kolkman, O., Mekking, W., and R. Gieben, "DNSSEC Operational Practices, Version 2", RFC 6781, DOI 10.17487/RFC6781, December 2012, <https://www.rfc-editor.org/rfc/rfc6781>.

[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>.

[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>.

[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>.

9.2. 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.peltan-edns-presentation-format]
Peltan, L. and T. Carpay, "EDNS Presentation and JSON
Format", Work in Progress, Internet-Draft, draft-peltan-
edns-presentation-format-03, 19 April 2024,
<https://datatracker.ietf.org/doc/html/draft-peltan-edns-
presentation-format-03>.

[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>.

[RFC4001] Daniele, M., Haberman, B., Routhier, S., and J.
Schoenwaelder, "Textual Conventions for Internet Network
Addresses", RFC 4001, DOI 10.17487/RFC4001, March 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>.

[RFC8427] Hoffman, P., "Representing DNS Messages in JSON",
RFC 8427, DOI 10.17487/RFC8427, July 2018,
<https://www.rfc-editor.org/rfc/rfc8427>.

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 include-delegparam being sets of more than one name.

```
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    ns1.example.
example.   NS    ns2.example.net.
example.   NS    ns3.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
ns1.example. A     192.0.2.1
ns1.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 an unnecessarily complicated setup to demonstrate the capabilities of the DELEG and DELEGPARAM RR types.

```
test.      DELEG server-ipv6=3fff::33
test.      DELEG include-delegparam=Acfg.example.org.,cname.example.org.
test.      DELEG include-delegparam=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 record 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.    DELEGPARAM server-ipv6=2001:DB8::6666
Acfg.example.org.    DELEGPARAM server-name=ns3.example.org.
Acfg.example.org.    DELEGPARAM include-delegparam=subcfg.example.org.

; unhelpful but technically legal CNAME
cname.example.org.   CNAME      Acfg.example.org.

ns3.example.org.     AAAA       3fff::33

subcfg.example.org.  DELEGPARAM 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.

```
ns2.example.net.     A          198.51.100.1

config2.example.net. DELEGPARAM 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    ns1.example.
example.   NS    ns2.example.net.
example.   NS    ns3.example.org.

;; Additional
ns1.example. A     192.0.2.1
ns1.example. AAAA  2001:DB8::1
```

A.4.1.2. Query for foo.test

See Section 5.2.2.1.

```
;; 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    ns1.example.
   example.   NS    ns2.example.net.
   example.   NS    ns3.example.org.
   example.   DS    44444 13 2 ABCDEF01234567...
   example.   RRSIG DS 13 1 300 20260101000000 (
                          20250101000000 33333 . SigExampleDS )
   ;; Additional
   ns1.example. A     192.0.2.1
   ns1.example. AAAA  2001:DB8::1
```

A.4.2.2. Query for foo.test

See Section 5.2.2.1.

DELEG-unaware validators would treat this answer as DNSSEC-secure.

DELEG-aware validators would treat it as DNSSEC-bogus because the DELEG bit in NSEC type bitmap would trigger downgrade attack detection (see Section 5.4.3).

```
;; 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 RCODE=NOERROR
;;

;; Question
foo.test.   IN MX

;; Answer
;; (empty)

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

;; Additional
;; (empty)
```

A follow-up example in Appendix A.5 explains the 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
   ;; (empty)
```

A.4.4.2. Query for foo.test

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

   ;; Question
   foo.test.      IN MX

   ;; Answer
   ;; (empty)

   ;; Authority
   test.      DELEG server-ipv6=3fff::33
   test.      DELEG include-delegparam=Acfg.example.org.
   test.      DELEG include-delegparam=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)
```

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

A.5. DELEGPARAM Interpretation

In the examples above, the test. DELEG record uses indirection and points to other domain names with DELEGPARAM, A, AAAA, and CNAME records. During resolution, a resolver will gradually build set of name servers to contact, as defined in Section 5.1.4.

To visualize the 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
```

Note the "cname.example.org." value in "include-delegparam=Acfg.example.org.,cname.example.org." DELEG record has no visible effect. The included name loops via CNAME back to "Acfg.example.org." which is already included. This would have caused duplication of all values if each SLIST was not treated as a set.

Implementations are free to use alternative representations for this data, as it is not directly exposed via DNS protocol.

A.6. Failure Cases

Several examples of misconfigured delegations which cannot be resolved follow.

Self-references to names without any addresses:

```
1p.invalid. DELEG include-delegparam=params.invalid.,sub.params.invalid.
2n.invalid. DELEG server-name=ns1.invalid.,ns2.sub.invalid.
```

Cycles:

```
c1.invalid. DELEG server-name=ns1.c2.invalid.
c2.invalid. DELEG include-delegparam=params.c1.invalid.,c3.invalid.
c3.invalid. CNAME c2.invalid.
```

Syntactically valid DELEG records without any Section 3.4 keys:

```
00.invalid. DELEG key65280=\032\037\041\045
00.invalid. DELEG key65281="char-string with whitespace"
```

A delegation missing the value for a mandatory key:

```
m1.invalid. DELEG mandatory=key65534
```

Records which are not even allowed in zone file (see also [RFC9460] appendix D.3) but might be sent in wire format:

```
m2.invalid. DELEG mandatory
ik.invalid. DELEG invalid
```

Appendix B. Test Vectors

TODO: In what format? Machine readable would be a win. Perhaps a combination of [RFC8427] and [I-D.peltan-edns-presentation-format]?

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, Ray Hunter, Hvard Eidnes, Ted Hardie, Michael Richardson, Florian Obser, Evan Hunt, ...

Authors' Addresses

Petr paek
ISC
Email: pspacek@isc.org

Ralf Weber
Akamai Technologies
Email: rweber@akamai.com

David C Lawrence
Salesforce

Email: tale@dd.org