

Network Working Group  
Internet-Draft  
Intended status: Informational  
Expires: 23 March 2022

B. Dickson  
GoDaddy  
19 September 2021

DS Algorithms for Securing NS and Glue  
draft-dickson-dnsop-ds-hack-02

Abstract

This Internet Draft proposes a mechanism to encode relevant data for NS records on the parental side of a zone cut by encoding them in DS records based on a new DNSKEY algorithm.

Since DS records are signed by the parent, this creates a method for validation of the otherwise unsigned delegation records.

Notably, support for updating DS records in a parent zone is already present (by necessity) in the Registry-Registrar-Registrant (RRR) provisioning system, EPP. Thus, no changes to the EPP protocol are needed, and no changes to registry database or publication systems upstream of the DNS zones published by top level domains (TLDs).

This NS validation mechanism is beneficial if the name server `_names_` need to be validated prior to use.

Status of This Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <https://datatracker.ietf.org/drafts/current/>.

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

This Internet-Draft will expire on 23 March 2022.

Copyright Notice

Copyright (c) 2021 IETF Trust and the persons identified as the document authors. All rights reserved.

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

## Table of Contents

1. Introduction . . . . .	2
2. Conventions and Definitions . . . . .	3
3. Background . . . . .	3
3.1. Attack Example . . . . .	3
4. New DNSKEY Algorithm . . . . .	4
4.1. Algorithm {TBD1} . . . . .	4
4.1.1. Example . . . . .	4
5. Validation Using These DS Records . . . . .	5
6. Protection of glue records . . . . .	5
7. Security Considerations . . . . .	5
8. IANA Considerations . . . . .	5
9. Normative References . . . . .	5
10. Informative References . . . . .	6
Appendix A. Acknowledgments . . . . .	6
Author's Address . . . . .	6

## 1. Introduction

Currently, any query for delegation NS records over an unprotected transport path returns results which do not have protection from tampering by an active on-path attacker, or against successful cache poisoning attacks. This is because the parent NS records are being authoritative, and thus do not have RRSIGs. The child NS records with the same owner name are authoritative, but the parent NS records are what get used for delegations.

There is new privacy work that relies on the name server names in the delegation RDATA. Unsigned records are vulnerable to modification by on-path attackers and to cache poisoning by off-path attackers. That privacy work uses the name for TLS validation, and the only source of the name server name is the NS record in the delegation.

This document is about protecting the RDATA of NS record, not the privacy issues per se.

Note that the use of an encrypted transport (such as DoT [RFC7858]) to the parent would be an alternative approach, but in the absence of encrypted transport, the current approach is recommended.

If an attacker alters the NS records returned, or poisons the resolver's cache for the unsigned delegation NS, the recursive resolver could be directed to a server operated by an attacker.

## 2. Conventions and Definitions

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.

## 3. Background

The methods developed for adding security to the Domain Name System, collectively referred to as DNSSEC, had as a primary requirement that they be backward compatible. The original specifications for DNS used the same Resource Record Type (RRTYPE) on both the parent and child side of a zone cut (the NS record). The main goal of DNSSEC was to ensure data integrity by using cryptographic signatures. However, owing to this overlap in the NS record type where the records above and below the zone cut have the same owner name created an inherent conflict, as only the child zone is authoritative for these records.

The result is that the parent side of the zone cut has records needed for DNS resolution which are not signed and not validatable.

This has no security (data validation) impact on DNS zones which are fully DNSSEC signed (anchored at the IANA DNS Trust Anchor), but does impact unsigned zones regardless of where the transition from secure to insecure occurs.

### 3.1. Attack Example

Suppose a resolver queries for the NS records for "example.com", at the name servers for the "com" TLD. Suppose this domain has been published in the com zone as "example.com NS ns1.example.net".

The response is not protected against MITM attacks. An on-path attacker can substitute its own name, "ns1.attacker.example". The resolver would then send its queries to the attacker.

Note that this vulnerability to MITM is present even if the domain "example.net" (the domain serving records for "ns1.example.net") is DNSSEC signed, and the resolver intends to use TLS to make queries for names within the child zone, "example.com".

Substituting the name server name is sufficient to prevent the resolver from validating the TLS connection. It can validate the received TLS certificate, but would do expect the certificate to be for "ns1.attacker.example".

#### 4. New DNSKEY Algorithm

This new DNSKEY algorithm conforms to the structure requirements from [RFC4034], but is not itself used as actual DNSKEY algorithm. It is assigned a value from the DNSKEY algorithm table. No DNSKEY records are published in the child zone using this algorithm.

This DNSKEY is used only as the input to the corresponding DS hashes published in the parent zone.

Note that this method is orthogonal to the specific choice of DS hashes. Examples here refer to the what is published currently in the IANA tables for recommended DNSSEC parameters, including recommended choices. Any valid supported hash for DS records MAY be used.

##### 4.1. Algorithm {TBD1}

This algorithm is used to validate the NS records of the delegation for the owner name.

The original NS records are canonicalized according to the DNSSEC signing process [RFC4034] section 6, including removing any label compression, and normalizing the character cases to lower case. The RDATA field of the record is hashed using the selected digest algorithm(s), e.g. SHA2-256 for DS digest algorithm 2.

Note that only the RDATA from the wire format of the original NS record is used in constructing the DS record.

##### 4.1.1. Example

Consider the delegation in the COM zone:

```
example.com NS ns1.Example.Net  
example.com NS ns2.Example.Net
```

The input to the digest for each NS record is the uncompressed wire format of their respective RVALUES.

The Key Tag is calculated per [RFC4034] using this value as the RDATA.

The resulting combination of NS and DS records are:

```
example.com NS ns1.Example.Net
example.com NS ns2.Example.Net
; example.com DS KeyTag=FOO Algorithm={TBD1}
;   DigestType=2 Digest=sha2-256(wireformat("ns1.example.net"))
example.com DS KeyTag=FOO Algorithm={TBD1} DigestType=2 Digest=...
; example.com DS KeyTag=FOO Algorithm={TBD1}
;   DigestType=2 Digest=sha2-256(wireformat("ns2.example.net"))
example.com DS KeyTag=FOO Algorithm={TBD1} DigestType=2 Digest=...
```

#### 5. Validation Using These DS Records

These new DS records are used to validate corresponding delegation records and glue. Each NS record must have a matching DS record. The expected DS record RDATA is constructed, and a matching DS record with identical RDATA MUST be present. Any NS record without matching valid DS record MUST be ignored.

\* NS records are validated using {TBD1}. The RDATA consists of only the RDATA from the NS record.

#### 6. Protection of glue records

For the issue of glue records (parent side A/AAAA records which are not signed), please see the proposal [I-D.dickson-dnsop-glueless].

#### 7. Security Considerations

As outlined earlier in FIXME, there could be security issues in various use cases.

The target domain containing each name server name is presumed (and required) to be DNSSEC signed.

#### 8. IANA Considerations

This document has no IANA actions. (FIXME - update this doc to specify the required IANA actions - add TBD1 to the DNSKEY algorithm table)

#### 9. Normative References

- [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/info/rfc4034>>.
- [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/info/rfc8174>>.

## 10. Informative References

- [I-D.dickson-dnsop-glueless]  
Dickson, B., "Operating a Glueless DNS Authority Server", Work in Progress, Internet-Draft, draft-dickson-dnsop-glueless-00, 17 September 2021, <<https://datatracker.ietf.org/doc/html/draft-dickson-dnsop-glueless-00>>.
- [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/info/rfc2119>>.
- [RFC7858]    Hu, Z., Zhu, L., Heidemann, J., Mankin, A., Wessels, D., and P. Hoffman, "Specification for DNS over Transport Layer Security (TLS)", RFC 7858, DOI 10.17487/RFC7858, May 2016, <<https://www.rfc-editor.org/info/rfc7858>>.

## Appendix A. Acknowledgments

Thanks to everyone who helped create the tools that let everyone use Markdown to create Internet Drafts, and the RFC Editor for xml2rfc.

Thanks to Dan York for his Tutorial on using Markdown (specifically mmark) for writing IETF drafts.

Thanks to YOUR NAME HERE for contributions, reviews, etc.

## Author's Address

Brian Dickson  
GoDaddy

Email: [brian.peter.dickson@gmail.com](mailto:brian.peter.dickson@gmail.com)

Network Working Group  
Internet-Draft  
Intended status: Informational  
Expires: 26 March 2022

B. Dickson  
GoDaddy  
22 September 2021

Operating a Glueless DNS Authority Server  
draft-dickson-dnsop-glueless-02

Abstract

This Internet Draft proposes a method for protecting authority servers against MITM and poisoning attacks, using a domain naming strategy to not require glue A/AAAA records and use of DNSSEC.

This technique assumes the use of validating resolvers.

MITM and poisoning attacks should only be effective/possible against unsigned domains.

However, until all domains are signed, this guidance is relevant, in that it can limit the attack surface of unsigned domains.

This guidance should be combined with [I-D.dickson-dnsop-ds-hack]

Status of This Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <https://datatracker.ietf.org/drafts/current/>.

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

This Internet-Draft will expire on 26 March 2022.

Copyright Notice

Copyright (c) 2021 IETF Trust and the persons identified as the document authors. All rights reserved.

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

## Table of Contents

1. Introduction	2
2. Conventions and Definitions	2
3. Background	2
4. Proposed Solutions	3
5. Terminology:	3
6. Recommendations	4
7. Security Considerations	7
8. IANA Considerations	8
9. Normative References	8
10. Informative References	8
Appendix A. Acknowledgments	8
Author's Address	8

## 1. Introduction

DNS Security Extensions (DNSSEC) are additions to the DNS protocol which provide data integrity and authenticity protections, but do not provide privacy.

## 2. Conventions and Definitions

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.

## 3. Background

Use of DNSSEC requires upgrades to software for authoritative servers, resolvers, and optionally clients, in order to benefit from these protections. It also requires that DNS operators actually sign their zones and secure the corresponding delegations at the parent.

When a given domain is unsigned or not securely delegated, those protections to the zone contents are not available.



Any such insecure domain is trivially able to be altered by an on-path attacker.

An off-path attacker is limited to use of cache poisoning attacks.

However, some class of cache poisoning attacks target unsigned delegation data. These records consist of the necessary NS records, and when necessary, "glue" records for IP addresses corresponding to these NS records.

The impact to successful cache poisoning of delegation records is that the attacker may substitute their own name servers for the legitimate name server. In other words, the attacker is able to promote itself to being effectively on-path, and trivially modify unsigned domain results.

#### 4. Proposed Solutions

This work does not propose any protocol changes. It provides guidance on strategies and techniques for name server naming.

There are two kinds of delegation records that require protection against off-path attackers, for unsigned domains.

For protecting NS records used in delegations, there is a new proposal for use of a new DS record. See [I-D.dickson-dnsop-ds-hack] for details.

The present draft addresses the "glue" records, by recommending methods to make them mostly unnecessary. If there is no delegation glue data, an attacker cannot poison that data. The resolver cache would contain only authoritative address records associated with NS names. Authoritative data cannot be pre-empted by such poisoning attacks, since those are only able to replace less trusted glue records.

Additional recommendations are made to reduce the chances for errors caused by DNS operators when changing delegation records, by avoiding re-use of name server names which require glue address records.

#### 5. Terminology:

The following terms are used to disambiguate domains and server names:

- \* Registered domain - end-user (registrant) domain

- In the parent zone, the registered domain is the left-hand side of the NS record
- \* Registered domain name server - the name of the name server serving the registered domain
- In the parent zone, the registered domain name server is the right-hand side of the NS record

## 6. Recommendations

The following practice is RECOMMENDED for unsigned domains:

- \* Do not use in-bailiwick registered domain name servers for unsigned domains.
- \* Instead, use out-of-zone names for the registered domain name servers of unsigned domains.

Example:

Do NOT do the following (delegations requiring glue):

\$ORIGIN example.

```
// Records in example TLD, with relative names
unsigned-domain NS ns1.unsigned-domain
unsigned-domain NS ns2.unsigned-domain
// glue
// "strictly necessary glue"
// always required for successful resolution
ns1.unsigned-domain A (IP address)
ns1.unsigned-domain AAAA (IP address)
ns2.unsigned-domain A (IP address)
ns2.unsigned-domain AAAA (IP address)
```

Instead, do the following (glueless delegations):

\$ORIGIN example.

```
// Records in example TLD, with relative names
// This is the minimum "glueless" set-up
// NS target name is not a "registered" host
// NS target is not used for glue for any domains
unsigned-domain NS ns1.nameserver-signed-domain
unsigned-domain NS ns2.nameserver-signed-domain
//
// Delegation to signed domain containing name server names
// (This domain serves the address records of name servers
// such as the glueless example above)
nameserver-signed-domain NS ns1.nameserver-signed-domain
nameserver-signed-domain NS ns2.nameserver-signed-domain
nameserver-signed-domain DS (DS record data)
// However, this domain needs to be resolvable, and needs glue
// glue records for this delegation
ns1.nameserver-signed-domain A (IP address)
ns1.nameserver-signed-domain AAAA (IP address)
ns2.nameserver-signed-domain A (IP address)
ns2.nameserver-signed-domain AAAA (IP address)
```

The following practice is RECOMMENDED:

- \* For any name server domain (domain containing addresses and related data for name servers used by registered domains), use distinct dedicated name servers for the domain itself
  - I.e. avoid sharing name servers between the name server domain and any registered domains
- \* Consider making the name server domain itself fully glueless, with an out-of-zone name server (using a tertiary domain)

- \* For this tertiary domain, also consider using separating the in-bailiwick name servers, from the names used for serving the name server domain
  - Limiting the in-bailiwick NS names ensures that changes and updates to the tertiary domain don't affect any other domains
  - Depending on parent zone policy (e.g. TLD database policy), renaming or renumbering name servers may affect delegations using them (NS entries)
  - A single domain with non-reused NS names guarantees side effects of this sort are not possible
- \* Overhead of tertiary domain and not re-using (or sharing) name server names in the tertiary domain:
  - Additional lookups are required on the initial reference to get the addresses of name servers for the main glueless domain
  - Subsequent (new) queries for the IP addresses of glueless name servers only require single queries

Example:

Entries in the example TLD  
\$ORIGIN example.

```
//  
// Same unsigned domain uses the same name servers  
// However, the name server is in its own glueless domain  
unsigned-registrant-domain NS ns1.signed-nameserver-domain  
unsigned-registrant-domain NS ns2.signed-nameserver-domain  
//  
signed-nameserver-domain NS ns1.tertiary-domain  
signed-nameserver-domain NS ns2.tertiary-domain  
signed-nameserver-domain DS (DS record data)  
//  
tertiary-domain NS special-ns1.tertiary-domain  
tertiary-domain NS special-ns2.tertiary-domain  
tertiary-domain DS (DS record data)  
// glue for special-ns1 and -2  
// special-ns1 and -2 are used only for/by tertiary-domain  
special-ns1.tertiary-domain A (IP address)  
special-ns1.tertiary-domain AAAA (IP address)  
special-ns2.tertiary-domain A (IP address)  
special-ns2.tertiary-domain AAAA (IP address)
```

Zone file for signed-nameserver-domain.example:

```
$ORIGIN signed-nameserver-domain.example.  
@ SOA (soa record data)  
// glueless NS are used  
@ NS ns1.tertiary-domain  
@ NS ns2.tertiary-domain  
// actual glueless address records for "real" name server names  
ns1 A (IP address)  
ns1 AAAA (IP address)  
ns2 A (IP address)  
ns2 AAAA (IP address)  
// etc etc etc
```

```
Zone file for tertiary-domain.example:  
$ORIGIN tertiary-domain.example.  
@ SOA (soa record data)  
//  
// This is the only non-glueless NS in use  
// (NB: matches glue address records in the parent)  
@ NS special-ns1  
@ NS special-ns2  
special-ns1 A (IP address)  
special-ns1 AAAA (IP address)  
special-ns2 A (IP address)  
special-ns2 AAAA (IP address)  
//  
// actual address records for "real" name server name  
// (only used by signed-nameserver-domain)  
// (These match glue records in the parent zone)  
ns1 A (IP address)  
ns1 AAAA (IP address)  
ns2 A (IP address)  
ns2 AAAA (IP address)
```

## 7. Security Considerations

This guidance is useful in preventing off-path attackers from poisoning DNS cache entries necessary for delegations.

However, an on-path attacker is still able to manipulate DNS responses sent over UDP or unencrypted TCP.

This guidance is not a substitute for use of DNSSEC for DNS domains. The only mechanism that can protect against on-path attackers is cryptographic protection DNSSEC signing of domains is both necessary and sufficient to provide data integrity protection.

Use of an encrypted transport is may be effective at preventing MITM attacks (i.e. DNS over TLS from resolver to authoritative server, aka ADoT), but does not provide provable data integrity.

Encrypted transport may be used in combination with DNSSEC signed zones and glueless name server domains.

Encrypted transport does not incrementally improve the data integrity or protection against MITM. DNSSEC is sufficient alone for this purpose. However, encrypted transport does add privacy protection against passive observers.

## 8. IANA Considerations

This document has no IANA actions.

## 9. Normative References

[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/info/rfc8174>>.

## 10. Informative References

[I-D.dickson-dnsop-ds-hack]  
Dickson, B., "DS Algorithms for Securing NS and Glue", Work in Progress, Internet-Draft, draft-dickson-dnsop-ds-hack-00, 11 August 2021, <<https://datatracker.ietf.org/doc/html/draft-dickson-dnsop-ds-hack-00>>.

[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/info/rfc2119>>.

## Appendix A. Acknowledgments

Thanks to everyone who helped create the tools that let everyone use Markdown to create Internet Drafts, and the RFC Editor for xml2rfc.

Thanks to Dan York for his Tutorial on using Markdown (specifically mmark) for writing IETF drafts.

Thanks to YOUR NAME HERE for contributions, reviews, etc.

## Author's Address

Brian Dickson  
GoDaddy

Email: [brian.peter.dickson@gmail.com](mailto:brian.peter.dickson@gmail.com)

Network Working Group  
Internet-Draft  
Intended status: Informational  
Expires: 13 May 2022

B. Dickson  
GoDaddy  
9 November 2021

Authenticated DNS over TLS to Authoritative Servers  
draft-dickson-dprive-adot-auth-06

Abstract

This Internet Draft proposes a mechanism for DNS resolvers to discover support for TLS transport to authoritative DNS servers, to validate this indication of support, and to authenticate the TLS certificates involved.

This requires that the name server `_names_` are in a DNSSEC signed zone.

This also requires that the delegation of the zone served is protected by [I-D.dickson-dnsop-ds-hack], since the NS names are the keys used for discovery of TLS transport support.

Additional recommendations relate to use of various techniques for efficiency and scalability, and new EDNS options to minimize round trips and for signaling between clients and resolvers.

Status of This Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <https://datatracker.ietf.org/drafts/current/>.

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

This Internet-Draft will expire on 13 May 2022.

Copyright Notice

Copyright (c) 2021 IETF Trust and the persons identified as the document authors. All rights reserved.



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

## Table of Contents

1. Introduction . . . . .	3
2. Conventions and Definitions . . . . .	3
3. Background . . . . .	3
4. Purpose . . . . .	3
4.1. New DNS Elements . . . . .	4
5. Requirements, and Limitations . . . . .	4
6. DNS Records To Publish for ADoT . . . . .	5
6.1. Server DNS Transport Support Signaling . . . . .	5
6.1.1. Examples . . . . .	5
6.2. DANE TLSA Records for ADoT (TLSADOT) . . . . .	6
6.2.1. Example . . . . .	6
6.3. Signaling DNS Transport for a Name Server . . . . .	7
6.3.1. Examples . . . . .	7
6.4. Signaling DNS Transport for a Domain . . . . .	7
6.4.1. Examples . . . . .	8
7. Validation Using DS Records, DNST Records, TLSADOT Records, and DNSSEC Validation . . . . .	8
7.1. Complete Example . . . . .	9
7.1.1. DNS Record Data . . . . .	9
7.1.2. Discussion Point - Wildcard-like Records . . . . .	11
7.1.3. Resolver Iterative Queries For Final TLS Query . . . . .	11
8. Signaling Resolver Support and Client Desire for ADoT . . . . .	14
8.1. Server Side Support Signaling . . . . .	15
8.2. Client Side Desire Signaling . . . . .	15
9. Security Considerations . . . . .	15
10. IANA Considerations . . . . .	15
11. Normative References . . . . .	15
12. Informative References . . . . .	16
Appendix A. Acknowledgments . . . . .	17
Author's Address . . . . .	17

## 1. Introduction

The Domain Name System (DNS) predates any concerns over privacy, including the possibility of pervasive surveillance. The original transports for DNS were UDP and TCP, unencrypted. Additionally, DNS did not originally have any form of data integrity protection, including against active on-path attackers.

DNSSEC (DNS Security extensions) added data integrity protection, but did not address privacy concerns. The original DNS over TLS [RFC7858] and DNS over HTTPS [RFC8484] specifications were limited to client-to-resolver traffic.

The remaining privacy component is recursive-to-authoritative servers. This Internet Draft is designed to provide a solution to this problem.

## 2. Conventions and Definitions

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.

## 3. Background

The result is that the parental side of the zone cut has records needed for DNS resolution which are not signed and not validatable.

## 4. Purpose

Authoritative DNS over TLS is intended to provide the following for communications from recursive resolvers to authoritative servers:

- \* Enable discovery of support for ADoT
- \* Validate the name server name
- \* Validate the server's TLS certificate
- \* Provide channel security using TLS

#### 4.1. New DNS Elements

The following are new protocol components, which are either included in this document, or are in other documents. Some are strictly required, while others are strongly suggested components to allow better scalability and performance. Some of the new elements are aliases to already documented standards, for purposes of these improvements. DNST refers to [I-D.dickson-dprive-dnst]

Element	New/Alias/ OPT	Format/ Base	Required	Description
DNST	New	Flags	Y	DNS Transport - support for DoT
TLSADOT	Alias	TLSA	Y	TLSA without prefixing
ADOTD	New	OPT RR (flag)	N	Signal desire for ADOT (client-resolver)
ADOTA	New	OPT RR (flag)	N	Signal availability of ADOT (resolver-client)
NSECD	New	OPT RR (flag)	N	Signal desire for NSEC(3) for [RFC8198]
NSV	New	DNSKEY Alg	Y	Protect NS - see [I-D.dickson-dnsop-ds-hack]

Table 1

#### 5. Requirements, and Limitations

This protocol depends on correct configuration and operation of the respective components, and that those are maintained according to Best Current Practices:

- \* Use of DS records [I-D.dickson-dnsop-ds-hack] for the protection of the delegation to the authoritative name servers
- \* Use of "glueless" zone(s) for name server names' zone [I-D.dickson-dnsop-glueless]
- \* DNSSEC signing of the zone serving the authoritative name servers' names [@RFC4034;@RFC4035;RFC5155]

- \* Proper management of key signing material for DNSSEC
- \* Ongoing management of RRSIGs on a timely basis (avoiding RRSIG expiry)
- \* Ensuring TLSADOT records are kept synchronized with the TLS certificates used
- \* Proper management of TLS private keys for TLS certificates used

There are external dependencies that impact the system security of any DNSSEC zone, which are inherently unavoidable in establishing this scheme. Specifically, the original DS record enrollment and any updates to the DS records involved in DNSSEC delegations are presumed secure and outside of the scope of the DNS protocol per se.

Other risks relate to normal information security practices, including access controls, role based access, audits, multi-factor authentication, multi-party controls, etc. These are out of scope for this protocol itself.

## 6. DNS Records To Publish for ADoT

ADoT is a property of DNS servers. The signaling is done at the server level, using a DNS record with the same owner name as the server itself (i.e. where the A and AAAA records for the server are published).

### 6.1. Server DNS Transport Support Signaling

In order to support ADoT for a DNS server, it is necessary to publish a record specifying explicit DoT support. This record also indicates other supported transports for the DNS server, e.g. the standard ports (TCP and UDP port 53).

The record type is "DNST" (DNS Transport), which is a single resource record consisting of flags for different supported transport types.

The zone serving the record MUST be DNSSEC signed. The absence of the DNST RRTYPE is proved by the NSEC(3) record, or else the DNST RRTYPE plus RRSIG is returned in response to a query for this record if it exists.

#### 6.1.1. Examples

Suppose the name server ns1.example.net supports only the normal DNS ports, and the name server ns2.example.net supports both the normal ports and ADoT. The zone example.net would include the records:

```
ns1.example.net. IN DNST UDP TCP
ns2.example.net. IN DNST UDP TCP DOT
```

And similarly, if another zone with many name server names wanted to have a policy of all-ADoT support (i.e. every name server supports ADoT), they would each be encoded as:

```
ns1.example2.net DNST UDP TCP DOT
ns2.example2.net DNST UDP TCP DOT
ns3.example2.net DNST UDP TCP DOT
ns4.example2.net DNST UDP TCP DOT
```

## 6.2. DANE TLSA Records for ADoT (TLSADOT)

The presence of ADoT requires additionally that a TLSA [RFC6698] record be provided. A new RRTYPE is to be created for this as an alias of TLSA, with mnemonic of "TLSADOT" (TLS ADOT Certificate). This record will be published at the location NS\_NAME, where NS\_NAME is the name of the name server. Any valid TLSA RDATA is permitted. The use of Certificate Usage types PKIX-TA and PKIX-EE is NOT RECOMMENDED since PKIX requires web PKI interactions. DANE types only require DNSSEC support. The use of Certificate Usage types DANE-TA records may provide more flexibility in provisioning and validation. On the other hand, DANE-EE is more secure, with fewer consequences for private key loss and certificate revocation. Per [RFC7218][RFC7671] the RECOMMENDED Selector and Matching types for this are SPKI and SHA2-256, giving the recommended TLSADOT record type of DANE-TA SPKI SHA2-256.

### 6.2.1. Example

In the above example, ns2.example.net supports DNS over TLS, and thus would need to have a TLSADOT record. The zone would include:

```
ns2.example.net. IN TLSADOT DANE-TA SPKI SHA2-256 (hash data)
```

If there were another zone containing many DNS server names, example2.net, it would be relatively simple to replicate otherwise identical records which use the same signing cert (rather than end-entity cert) in the TLSADOT record.

This would look like the following:

```
ns1.example2.net IN TLSADOT DANE-TA SPKI SHA2-256 (hash data)
ns2.example2.net IN TLSADOT DANE-TA SPKI SHA2-256 (hash data)
ns3.example2.net IN TLSADOT DANE-TA SPKI SHA2-256 (hash data)
ns4.example2.net IN TLSADOT DANE-TA SPKI SHA2-256 (hash data)
ns1.example2.net IN A IP4_ADDRESS1
ns2.example2.net IN A IP4_ADDRESS2
ns3.example2.net IN A IP4_ADDRESS3
ns4.example2.net IN A IP4_ADDRESS4
ns1.example2.net IN AAAA IP6_ADDRESS1
ns2.example2.net IN AAAA IP6_ADDRESS2
ns3.example2.net IN AAAA IP6_ADDRESS3
ns4.example2.net IN AAAA IP6_ADDRESS4
```

### 6.3. Signaling DNS Transport for a Name Server

This transport signaling MUST only be trusted if the name server names for the domain containing the relevant name servers' names are protected with [I-D.dickson-dnsop-ds-hack] and validated. The name servers must also be in a DNSSEC signed zone (i.e. securely delegated where the delegation has been successfully DNSSEC validated).

The specific DNS transport that a name server supports is indicated via use of an RRSSet of RRTYPE "DNST".

#### 6.3.1. Examples

We re-use the same examples from above, indicating whether or not individual authoritative name servers support DoT:

```
ns1.example.net. IN DNST UDP TCP DOTDNST
ns2.example.net. IN DNST UDP TCP DOTDNST
```

And similarly, if another zone with many name server names wanted to have a policy of all-ADoT support (i.e. every name server supports ADoT), this could be encoded as:

```
ns1.example2.net DNST UDP TCP DOT
ns2.example2.net DNST UDP TCP DOT
ns3.example2.net DNST UDP TCP DOT
ns4.example2.net DNST UDP TCP DOT
```

### 6.4. Signaling DNS Transport for a Domain

A domain inherits the signaled transport for the name servers serving the domain.

This transport signaling MUST only be trusted for use of ADoT if the delegated name server names for the domain are protected with [I-D.dickson-dnsop-ds-hack].

The delegation to NS names "A" and "B", along with the DS record protecting/encoding "A" and "B", results in the DNS transport that is signaled for "A" and "B" being applied to the domain being delegated. This transport will include ADoT IFF the transport for "A" and "B" has included ADoT via DNS records.

#### 6.4.1. Examples

No additional configuration is needed, beyond use of authority servers which signal DoT support. The following examples assumes the previous DNS records are provisioned:

```
example.com NS ns1.example.net. // does not support ADoT
example.com NS ns2.example.net. // supports ADoT
```

```
example2.com NS ns1.example2.net. // all support ADoT
example2.com NS ns2.example2.net. // all support ADoT
```

In this example, ns1 does not have ADoT support (since the DNST record excludes the DOT flag), while ns2 does support ADoT (since it includes DOT).

### 7. Validation Using DS Records, DNST Records, TLSADOT Records, and DNSSEC Validation

These records are used to validate corresponding delegation records, DNST records, and TLSADOT records, as follows:

- \* Initial domain NS records are validated using [I-D.dickson-dnsop-ds-hack]
- \* All DS records implementing [I-D.dickson-dnsop-ds-hack] must be DNSSEC validated prior to use
- \* Once the NS names have been validated, and the delegations to the appropriate name servers are validated, the DNST records for the NS name are obtained to identify the DNS transport methods supported.
- \* If ADoT is among the supported transports, the TLSADOT record for the name server is obtained, and used for verification of the TLS certificate when making the TLS connection.

## 7.1. Complete Example

### 7.1.1. DNS Record Data

Suppose a client requests resolution for the IP address of "sensitive-name.example.com". Suppose the client's resolver has a "cold" cache without any entries beyond the standard Root Zone and relevant TLD name server records.

Suppose the following entries are present at their respective TLD authority servers, delegating to the respective authority servers:

```
// (Single NS for brevity only, please use 2 NS minimum )
// Unsigned delegations to various single-operator servers
example2.com NS ns1.example2.net. // all support ADoT
example3.com NS ns2.example2.net. // all support ADoT
example4.com NS ns3.example2.net. // all support ADoT
example5.com NS ns4.example2.net. // all support ADoT

// Zone serving NS data for single-operator's servers
example2.net NS ns1.infra2.example
example2.net NS ns2.infra2.example
example2.net DS (DS record data)
// glueless name servers are used

// Special zone serving NS data for previous zone
infra2.example NS ns1-glue.infra2.example
infra2.example NS ns2-glue.infra2.example
infra2.example DS (DS record data)
// Note use of glue for only this zone's delegation
ns1-glue.infra2.example A (glue A data)
ns1-glue.infra2.example AAAA (glue AAAA data)
ns2-glue.infra2.example A (glue A data)
ns2-glue.infra2.example AAAA (glue AAAA data)
```

Suppose the following additional entries are in the respective authority servers for the ADoT signaling/certs:



```
example2.net SOA ( SOA record data )
// glueless name servers are used
example2.net NS ns1.infra2.example
example2.net NS ns2.infra2.example
//
// DNS Transport for discovery of support
ns1.example2.net DNST UDP TCP DOT
ns2.example2.net DNST UDP TCP DOT
ns3.example2.net DNST UDP TCP DOT
ns4.example2.net DNST UDP TCP DOT
//
// TLSADOT signing cert
ns1.example2.net IN TLSADOT DANE-TA SPKI SHA2-256 (hash data)
ns2.example2.net IN TLSADOT DANE-TA SPKI SHA2-256 (hash data)
ns3.example2.net IN TLSADOT DANE-TA SPKI SHA2-256 (hash data)
ns4.example2.net IN TLSADOT DANE-TA SPKI SHA2-256 (hash data)
//
// Addresses of name servers serving customer zones
// E.g. example2.com to example5.com served on these
ns1.example2.net IN A IP4_ADDRESS1
ns2.example2.net IN A IP4_ADDRESS2
ns3.example2.net IN A IP4_ADDRESS3
ns4.example2.net IN A IP4_ADDRESS4
ns1.example2.net IN AAAA IP6_ADDRESS1
ns2.example2.net IN AAAA IP6_ADDRESS2
ns3.example2.net IN AAAA IP6_ADDRESS3
ns4.example2.net IN AAAA IP6_ADDRESS4
//
// plus RRSIGs and NSEC(3) records and their RRSIGs

infra2.example SOA ( SOA record data )
infra2.example NS ns1-glue.infra2.example
infra2.example NS ns2-glue.infra2.example
ns1-glue.infra2.example A (same as glue A data)
ns1-glue.infra2.example AAAA (same as glue AAAA data)
ns2-glue.infra2.example A (same as glue A data)
ns2-glue.infra2.example AAAA (same as glue AAAA data)
//
// name server info for example2.net zone
ns1.infra2.example A (glueless A data)
ns1.infra2.example AAAA (glueless AAAA data)
ns2.infra2.example A (glueless A data)
ns2.infra2.example AAAA (glueless AAAA data)
//
// plus RRSIGs and NSEC(3) records and their RRSIGs
```

### 7.1.2. Discussion Point - Wildcard-like Records

Wildcard records have RRTYPE(s), but are only instantiated when an owner name does not exist.

If wildcards were instantiated whenever the 3-tuple (name, class, type) did not exist, use of wildcard records for DNST and TLSADOT would be a logical choice.

The discussion point is as follows:

- \* Would it make sense to support a wildcard-like behavior for covering many owner names which did not have explicit DNST and/or TLSADOT records of their own?
- \* If so, when/how would that be signalled?
  - It could be explicit, using a separate RRTYPE to flag the need to use the parent name (zone apex) for the required RRTYPE.
    - o This would support use of NSEC(3) records to check for the flag
    - o A resolver could use the flag to optimize cache usage for the parent record. Once the parent is in the cache, the flag in the NSEC(3) for the owner name would trigger use of the cached parent record.
  - It could be implicit, meaning the absence of the explicit record type results in the need to search for the record type at another name (e.g. zone apex).
    - o The lack of explicit record could be detected from NSEC(3) records
    - o The implicit flag would be handled the same as the explicit flag case above.
- \* The TLSADOT record at the parent zone would only be viable for DANE-TA or PKIX-TA types.

### 7.1.3. Resolver Iterative Queries For Final TLS Query

(In the following, use of wildcard-type records and semantics is assumed, but not explicitly described currently. Literal wildcard record labels ("\*") are used as a placeholder, pending the above Discussion Point's resolution.)

The following are the necessary queries to various servers necessary to do a private TLS-protected lookup.

Several examples are provided in order, from a presumed cold cache state. Root Priming and TLD queries are presumed to already have been complete.

1. Query for sensitive-name.example2.com:

1. Query for NS for example2.com => get NS ns1.example2.net plus DS => validate the DS and proceed
2. Query for NS for example2.net => get NS ns1/ns2.infra2.example plus DS => validate the DS and proceed
3. Query for NS for infra2.example2.net => get NS ns1-glue/ns2-glue.infra2.example plus DS plus glue A/AAAA => validate the DS and proceed
4. Query with NSEC3 for A for ns1/ns2.infra2.example => get A for ns1/ns2.infra2.example plus RRSIGs plus NSEC(3) plus RRSIG => validate the RRSIGs and proceed
5. Query with NSEC3 for A for ns1.example2.net => get A for ns1.example2.net plus RRSIG plus NSEC(3) plus RRSIG => validate the RRSIGs and proceed
6. Query with NSEC3 for DNST for ns1.example2.net => get DNST for \*.example2.net plus RRSIG plus special wildcard NSEC(3)s plus RRSIGs => validate the RRSIGs and proceed
7. Query with NSEC3 for TLSADOT for ns1.example2.net => get TLSADOT for \*.example2.net plus RRSIG plus special wildcard NSEC(3)s plus RRSIGs => validate the RRSIGs and proceed
8. Query over TLS for sensitive-name.example2.com (to ns1.example2.net, match TLS cert chain against DANE-TA cert, only query once TLS established)

2. Query for sensitive-name.example3.com:

1. Query for NS for example2.com => get NS ns1.example2.net plus DS => validate the DS and proceed
2. Query with NSEC3 for A for ns1.example2.net => get A for ns1.example2.net plus RRSIG plus NSEC(3) plus RRSIG => validate the RRSIGs and proceed

3. NB: already have wildcards for DNST and TLSADOT plus NSEC3 proving no non-wildcards exist for ns1.example2.net for those types, synthesize DNST and TLSADOT records)
4. Query over TLS for sensitive-name.example2.com (to ns1.example2.net, match TLS cert chain against DANE-TA cert, only query once TLS established)
3. Query for sensitive-name.example4.com:
  1. Query for NS for example2.com => get NS ns1.example2.net plus DS => validate the DS and proceed
  2. Query with NSEC3 for A for ns1.example2.net => get A for ns1.example2.net plus RRSIG plus NSEC(3) plus RRSIG => validate the RRSIGs and proceed
  3. NB: already have wildcards for DNST and TLSADOT plus NSEC3 proving no non-wildcards exist for ns1.example2.net for those types, synthesize DNST and TLSADOT records)
  4. Query over TLS for sensitive-name.example2.com (to ns1.example2.net, match TLS cert chain against DANE-TA cert, only query once TLS established)
4. Query for sensitive-name.example5.com:
  1. Query for NS for example2.com => get NS ns1.example2.net plus DS => validate the DS and proceed
  2. Query with NSEC3 for A for ns1.example2.net => get A for ns1.example2.net plus RRSIG plus NSEC(3) plus RRSIG => validate the RRSIGs and proceed
  3. NB: already have wildcards for DNST and TLSADOT plus NSEC3 proving no non-wildcards exist for ns1.example2.net for those types, synthesize DNST and TLSADOT records)
  4. Query over TLS for sensitive-name.example2.com (to ns1.example2.net, match TLS cert chain against DANE-TA cert, only query once TLS established)
5. Query for sensitive-name2.example2.com:
  1. (Already have delegation entry for example2.com in cache.)
  2. (Already have A for ns1.example2.net in cache.)

3. (Already have all TLS info in the cache.)
4. Query over TLS for sensitive-name.example2.com (to ns1.example2.net, match TLS cert chain against DANE-TA cert, only query once TLS established)

Once the initial query or queries for a name server zone has been done, if that zone uses wildcards for DNST and TLSADOT, the only queries needed for a new name server are the A and/or AAAA records. Once the initial query for a name server has been done, all of the address and TLS information is available in the cache, and the DOT query can be made upon receipt of the TLD delegation record. Once the initial query for a second-level domain has been done, the TLD delegation and all of the address and TLS information is available in the cache, and the DOT query can be made immediately.

Once a cache is populated with wildcards from the name server domain, additional delegation queries require no more trips than those needed for normal UDP queries:

1. Query for delegation from TLD, and validate the response
2. Query for the name server's address(es), and validate the response
3. Send the query to the authoritative server for the domain with the sensitive name (over TLS or over UDP/TCP, depending on transport supported by the authoritative server)

Once a cache is populated with name server addresses and wildcards, additional delegation queries require no more trips than those needed for normal UDP queries:

1. Query for delegation from TLD, and validate the response
2. Send the query to the authoritative server for the domain with the sensitive name (over TLS or over UDP/TCP, depending on transport supported by the authoritative server)

#### 8. Signaling Resolver Support and Client Desire for ADoT

The following presume some new OPT sub-types, to be added to the IANA action section or to be split out as separate drafts. The sub-type mnemonics are "ADOTA" (available) and "ADOTD" (desired), each with an enumerated set of values and mnemonic codes. Respectively those are: "Always", "Upon Request", and "Never"; and "Force", "If Available", and "Never".

### 8.1. Server Side Support Signaling

A DNS server (e.g. recursive resolver or forwarder) MAY signal to clients that it offers the use of ADoT. The mechanism used is to set the EDNS option "ADOTA". The values for this option are "Always", "Upon Request", and "Never". The value "Always" indicates the server will always attempt to use ADoT without regards to client requests for ADoT. The value "Upon Request" indicates that the server will ONLY use ADoT for upstream queries if the client requests that ADoT be used. These values have no effect on answers served from the resolver's cache. (The "Never" case is unusual, in that it signals the server understands the option, but does not perform ADoT. Generally this would be used to allow a client to track changes in the status, if the client is interested in uses of ADoT.)

### 8.2. Client Side Desire Signaling

A DNS client (e.g. stub or forwarder) MAY signal the desire to have the resolver use ADoT. The mechanism used is to set the EDNS option "ADOTD". The values for this option are "Force", "If Available", and "Never". The value "Force" indicates the server should attempt to use ADoT, and return a failure code of XXXX and an EDE value of YYYY if the authoritative server does not offer ADoT, or any other ADoT failure occurs. The value "If Available" indicates that the server should use ADoT for upstream queries if it is available, but SHOULD NOT allow any downgrades if the authoritative server signals that ADoT is available. These values have no effect on answers served from the resolver's cache. (The "Never" case is unusual, in that it signals the client understands the option, but does not perform ADoT. Generally this would be used to allow a server to track changes in the client base, so the server operator can make informed decisions about enabling ADoT.)

## 9. Security Considerations

As outlined above, there could be security issues in various use cases.

## 10. IANA Considerations

This document may or may not have any IANA actions. (e.g. if the RRTYPES, EDNS subtypes, DNSKEY algorithms, etc., are defined in other documents, no IANA actions are needed.)

## 11. Normative References

- [RFC6698] Hoffman, P. and J. Schlyter, "The DNS-Based Authentication of Named Entities (DANE) Transport Layer Security (TLS) Protocol: TLSA", RFC 6698, DOI 10.17487/RFC6698, August 2012, <<https://www.rfc-editor.org/info/rfc6698>>.
- [RFC7218] Gudmundsson, O., "Adding Acronyms to Simplify Conversations about DNS-Based Authentication of Named Entities (DANE)", RFC 7218, DOI 10.17487/RFC7218, April 2014, <<https://www.rfc-editor.org/info/rfc7218>>.
- [RFC7671] Dukhovni, V. and W. Hardaker, "The DNS-Based Authentication of Named Entities (DANE) Protocol: Updates and Operational Guidance", RFC 7671, DOI 10.17487/RFC7671, October 2015, <<https://www.rfc-editor.org/info/rfc7671>>.
- [RFC7858] Hu, Z., Zhu, L., Heidemann, J., Mankin, A., Wessels, D., and P. Hoffman, "Specification for DNS over Transport Layer Security (TLS)", RFC 7858, DOI 10.17487/RFC7858, May 2016, <<https://www.rfc-editor.org/info/rfc7858>>.
- [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/info/rfc8174>>.
- [RFC8198] Fujiwara, K., Kato, A., and W. Kumari, "Aggressive Use of DNSSEC-Validated Cache", RFC 8198, DOI 10.17487/RFC8198, July 2017, <<https://www.rfc-editor.org/info/rfc8198>>.
- [RFC8484] Hoffman, P. and P. McManus, "DNS Queries over HTTPS (DoH)", RFC 8484, DOI 10.17487/RFC8484, October 2018, <<https://www.rfc-editor.org/info/rfc8484>>.

## 12. Informative References

- [I-D.dickson-dnsop-ds-hack]  
Dickson, B., "DS Algorithms for Securing NS and Glue", Work in Progress, Internet-Draft, draft-dickson-dnsop-ds-hack-02, 19 September 2021, <<https://datatracker.ietf.org/doc/html/draft-dickson-dnsop-ds-hack-02>>.
- [I-D.dickson-dnsop-glueless]  
Dickson, B., "Operating a Glueless DNS Authority Server", Work in Progress, Internet-Draft, draft-dickson-dnsop-glueless-02, 22 September 2021, <<https://datatracker.ietf.org/doc/html/draft-dickson-dnsop-glueless-02>>.

[I-D.dickson-dprive-dnst]

Dickson, B., "Resource Record for Signaling Transport for DNS to Authority Servers", Work in Progress, Internet-Draft, draft-dickson-dprive-dnst-00, 24 October 2021, <<https://datatracker.ietf.org/doc/html/draft-dickson-dprive-dnst-00>>.

[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/info/rfc2119>>.

#### Appendix A. Acknowledgments

Thanks to everyone who helped create the tools that let everyone use Markdown to create Internet Drafts, and the RFC Editor for xml2rfc.

Thanks to Dan York for his Tutorial on using Markdown (specifically mmark) for writing IETF drafts.

Thanks to YOUR NAME HERE for contributions, reviews, etc.

#### Author's Address

Brian Dickson  
GoDaddy

Email: [brian.peter.dickson@gmail.com](mailto:brian.peter.dickson@gmail.com)



Network Working Group  
Internet-Draft  
Intended status: Standards Track  
Expires: 27 April 2022

B. Dickson  
GoDaddy  
24 October 2021

Resource Record for Signaling Transport for DNS to Authority Servers  
draft-dickson-dprive-dnst-00

Abstract

This Internet Draft proposes an RRTYPE to signal explicit support for transport types for DNS service. This new RRTYPE is "DNST". The available transports to signal are TCP and UDP on port 53 (DNS), and DoT (DNS over TLS) transport using TCP port 853.

Status of This Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <https://datatracker.ietf.org/drafts/current/>.

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

This Internet-Draft will expire on 27 April 2022.

Copyright Notice

Copyright (c) 2021 IETF Trust and the persons identified as the document authors. All rights reserved.

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

## Table of Contents

1. Introduction . . . . .	2
2. Conventions and Definitions . . . . .	2
3. Background . . . . .	2
4. Remove Before Publication . . . . .	2
5. DNS Transport RRTYPE . . . . .	3
6. Restrictions . . . . .	3
7. Wire Format . . . . .	3
8. Presentation Format . . . . .	3
9. Additional Processing . . . . .	4
10. Security Considerations . . . . .	4
11. IANA Considerations . . . . .	4
12. Normative References . . . . .	4
13. Informative References . . . . .	4
Appendix A. Acknowledgments . . . . .	4
Author's Address . . . . .	5

## 1. Introduction

## 2. Conventions and Definitions

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.

## 3. Background

DNS over TLS is defined in [RFC7858]. However, there is no explicit signaling for when DoT should be used. Without explicit signaling, there is no protection against downgrade attacks by an on-path attacker.

## 4. Remove Before Publication

Notes on design decisions, including the decision NOT to use an SVCB-compatible format:

- \* NS records MUST point to non-CNAME records. Thus, there is no need for the SVCB "Alias-form" behavior. DNST does not support aliasing,
- \* DNST allows for explicit rejection of default transport (UDP/53 and TCP/53)
- \* DNST allows explicit signaling of DoT

- \* There is no need for alternate port numbers for UDP or TCP port 53, or for DoT port 853.
- \* There is no need for DoH, since the expected clients are limited to DNS resolvers.

## 5. DNS Transport RRTYPE

The solution to this problem is to introduce a method for explicit signaling for when DoT is available. When combined with TLSA [RFC6698] records for the corresponding DNS server name, any client wishing to use DoT is able to know that it is available, and can detect and avoid any attempts at transport downgrade.

This document defines the RRTYPE value {TBD} with mnemonic name DNST ("DNS Transport"). This consists of a set of flags indicating supported transport for the DNS server at the owner name. The flag bits represent transports:

- \* UDP on port 53
- \* TCP on port 53
- \* DoT (DNS over TLS) on port 853

## 6. Restrictions

The DNST record may occur anywhere, including at the apex of a DNS zone, and may co-exist with any other type that also permits other types.

## 7. Wire Format

The RDATA wire format is an 8-bit octet of flag bits.

| UDP | TCP | DOT | 5 unused bits |

## 8. Presentation Format

OWNER CLASS TTL DNST [UDP] [TCP] [DOT]

At least one of the transport types must be present.

## 9. Additional Processing

The authoritative server MAY/SHOULD return both the DNST record(s) and any/all A and AAAA records with the same owner name. This reduces the number of queries the resolver would otherwise have to make (i.e. two additional queries for A and AAAA record types).

## 10. Security Considerations

The DNST record MUST be in a DNSSEC-signed zone. This ensures protection against downgrade attacks on the transport signaling.

## 11. IANA Considerations

IANA is directed to add a new record to the DNS RRTYPES table to add the entry "DNST" with value "TBD", referencing this document.

## 12. Normative References

- [RFC6698] Hoffman, P. and J. Schlyter, "The DNS-Based Authentication of Named Entities (DANE) Transport Layer Security (TLS) Protocol: TLSA", RFC 6698, DOI 10.17487/RFC6698, August 2012, <<https://www.rfc-editor.org/info/rfc6698>>.
- [RFC7858] Hu, Z., Zhu, L., Heidemann, J., Mankin, A., Wessels, D., and P. Hoffman, "Specification for DNS over Transport Layer Security (TLS)", RFC 7858, DOI 10.17487/RFC7858, May 2016, <<https://www.rfc-editor.org/info/rfc7858>>.
- [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/info/rfc8174>>.

## 13. Informative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.

## Appendix A. Acknowledgments

Thanks to everyone who helped create the tools that let everyone use Markdown to create Internet Drafts, and the RFC Editor for xml2rfc.

Thanks to Dan York for his Tutorial on using Markdown (specifically mmark) for writing IETF drafts.

Thanks to YOUR NAME HERE for contributions, reviews, etc.

Author's Address

Brian Dickson  
GoDaddy

Email: [brian.peter.dickson@gmail.com](mailto:brian.peter.dickson@gmail.com)

dprive  
Internet-Draft  
Intended status: Informational  
Expires: 30 July 2022

D.K. Gillmor  
ACLU  
J. Salazar  
A19  
26 January 2022

Unilateral Opportunistic Deployment of Encrypted Recursive-to-  
Authoritative DNS  
draft-dkgjsal-dprive-unilateral-probing-02

Abstract

This draft sets out steps that DNS servers (recursive resolvers and authoritative servers) can take unilaterally (without any coordination with other peers) to defend DNS query privacy against a passive network monitor. The steps in this draft can be defeated by an active attacker, but should be simpler and less risky to deploy than more powerful defenses. The draft also introduces (but does not try to specify) the semantics of signalling that would permit defense against an active attacker.

The goal of this draft is to simplify and speed deployment of opportunistic encrypted transport in the recursive-to-authoritative hop of the DNS ecosystem. With wider easy deployment of the underlying transport on an opportunistic basis, we hope to facilitate the future specification of stronger cryptographic protections against more powerful attacks.

Status of This Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <https://datatracker.ietf.org/drafts/current/>.

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

This Internet-Draft will expire on 30 July 2022.

## Copyright Notice

Copyright (c) 2022 IETF Trust and the persons identified as the document authors. All rights reserved.

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

## Table of Contents

1. Introduction . . . . .	3
1.1. Requirements Language . . . . .	3
1.2. Terminology . . . . .	3
2. Priorities . . . . .	3
2.1. Minimizing Negative Impacts . . . . .	4
2.2. Protocol Choices . . . . .	4
3. Guidance for Authoritative Servers . . . . .	5
3.1. Pooled Authoritative Servers Behind a Single IP Address . . . . .	5
3.2. Authentication . . . . .	5
3.3. Server Name Indication . . . . .	6
3.4. Resource Exhaustion . . . . .	6
4. Guidance for recursive resolvers . . . . .	7
4.1. Overall recursive resolver Settings . . . . .	7
4.2. Recursive Resolver Requirements . . . . .	8
4.3. Authoritative Server Encrypted Transport Connection State . . . . .	8
4.3.1. Separate State for Each of the Recursive Resolver's Own IP Addresses . . . . .	10
4.4. Maintaining Authoritative State by IP Address . . . . .	10
4.5. Probing Policy . . . . .	11
4.5.1. Sending a query over Do53 . . . . .	11
4.5.2. Receiving a response over Do53 . . . . .	12
4.5.3. Initiating a connection over encrypted transport . . . . .	12
4.5.4. Establishing an encrypted transport connection . . . . .	14
4.5.5. Failing to establish an encrypted transport connection . . . . .	15
4.5.6. Encrypted transport failure . . . . .	15
4.5.7. Handling clean shutdown of encrypted transport connection . . . . .	16
4.5.8. Sending a query over encrypted transport . . . . .	17
4.5.9. Receiving a response over encrypted transport . . . . .	17

4.5.10. Resource Exhaustion . . . . .	18
4.5.11. Maintaining connections . . . . .	19
5. Signalling for Stronger Defense . . . . .	19
5.1. Combining Signals with Opportunistic Probing . . . . .	20
6. IANA Considerations . . . . .	20
7. Privacy Considerations . . . . .	20
7.1. Server Name Indication . . . . .	20
8. Security Considerations . . . . .	20
9. Acknowledgements . . . . .	21
10. References . . . . .	21
10.1. Normative References . . . . .	21
10.2. Informative References . . . . .	21
Appendix A. Document Considerations . . . . .	22
A.1. Document History . . . . .	23
A.1.1. Substantive changes from -01 to -02 . . . . .	23
A.1.2. Substantive changes from -00 to -01 . . . . .	23
Authors' Addresses . . . . .	23

## 1. Introduction

### 1.1. Requirements Language

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] and [RFC8174]) when, and only when, they appear in all capitals, as shown here.

### 1.2. Terminology

- \* "unilateral" means capable of opportunistic probing deployment without external coordination with any of the other parties
- \* Do53 refers to traditional cleartext DNS over port 53 ([RFC1035])
- \* DoQ refers to DNS-over-QUIC ([I-D.ietf-dprive-dnsquic])
- \* DoT refers to DNS-over-TLS ([RFC7858])
- \* DoH refers to DNS-over-HTTPS ([RFC8484])
- \* Encrypted transports refers to DoQ, DoT, and DoH collectively

## 2. Priorities

This document aims to provide guidance to implementers who want to simply enable protection against passive network observers.



In particular, it focuses on mechanisms that can be adopted unilaterally by recursive resolvers and authoritative servers, without any explicit coordination with the other parties. This guidance provides opportunistic security (see [RFC7435]) -- encrypting things that would otherwise be in the clear, without interfering with or weakening stronger forms of security.

## 2.1. Minimizing Negative Impacts

It also aims to minimize potentially negative impacts caused by the probing of encrypted transports -- for the systems that adopt these guidelines, for the parties that they communicate with in the "second hump" of the DNS camel, and for uninvolved third parties. The negative impacts that we specifically try to minimize are:

- \* excessive bandwidth use
- \* excessive computational resources (CPU and memory in particular)
- \* amplification attacks (where DNS resolution infrastructure is wielded as part of a DoS attack)

## 2.2. Protocol Choices

While this document focuses specifically on strategies used by DNS servers, it does not go into detail on the specific protocols used, as those protocols --- in particular, DoT and DoQ --- are described in other documents.

This document does not pursue the use of DoH in this context, because a DoH client needs to know the path part of a DoH endpoint URL, and there are currently no mechanisms for a DNS resolver to predict the path on its own, in an opportunistic or unilateral fashion, without incurring in excessive use of resources. For instance, a recursive resolver in theory could guess the full path to a queried IP address by trying all the URL paths that the client has in records and see if one of those works, but even though it can be expected that this would work 99% of the time with fewer than 100 probes, this technique would likely incur in excessive resource consumption potentially leading to vulnerabilities and amplification attacks. The authors of this draft particularly welcome ideas and contributions from the community that lead to a suitable mechanism for unilaterally probing for DoH-capable authoritative servers, for later consideration in this or other drafts.

### 3. Guidance for Authoritative Servers

An authoritative server SHOULD implement and deploy DNS-over-TLS (DoT) on TCP port 853.

An authoritative server MAY implement and deploy DNS-over-QUIC (DoQ) on UDP port 853.

#### 3.1. Pooled Authoritative Servers Behind a Single IP Address

Some authoritative DNS servers are structured as a pool of authoritatives standing behind a load-balancer that runs on a single IP address, forwarding queries to members of the pool.

In such a deployment, individual members of the pool typically get updated independently from each other.

A recursive resolver following the guidance in Section 4 that interacts with such a pool likely does not know that it is a pool. If some members of the pool are updated to follow this guidance while others are not, the recursive client might see the pool as a single authoritative server that sometimes offers and sometimes refuses encrypted transport.

To avoid incurring additional minor timeouts for such a recursive resolver, the pool operator SHOULD either:

- \* ensure that all members of the pool enable the same encrypted transport(s) within the span of a few seconds, or
- \* ensure that the load balancer maps client requests to pool members based on client IP addresses.

Similar concerns apply to authoritative servers responding from an anycast IP address. As long as the pool of servers is in a heterogenous state, any flapping route that switches a given client IP address to a different responder risks incurring an additional timeout. Frequent changes of routing for anycast listening IP addresses are also likely to cause problems for TLS, TCP, or QUIC connection state as well, so stable routes are important to ensure that the service remains available and responsive.

#### 3.2. Authentication

For unilateral deployment, an authoritative server does not need to offer any particular form of authentication.

The simplest deployment would simply provide a self-issued, regularly-updated X.509 certificate. This mechanism is supported by many TLS and QUIC clients, and will be acceptable for any opportunistic connection.

Possible alternate forms of server authentication include:

- \* an X.509 Certificate issued by a widely-known certification authority associated with the common NS names used for this authoritative server
- \* DANE authentication (potentially including the TLS handshake)

### 3.3. Server Name Indication

An authoritative DNS server that wants to handle unilateral queries MAY rely on Server Name Indication (SNI) to select alternate server credentials. However, such a server MUST NOT serve resource records that differ based on SNI (or on the lack of SNI) provided by the client, as a probing recursive resolver that offers SNI might or might not have used the right server name to get the records it's looking for.

### 3.4. Resource Exhaustion

A well-behaved recursive resolver may keep an encrypted connection open to an authoritative server, to amortize the costs of connection setup for both parties.

However, some authoritative servers may have insufficient resources available to keep many connections open concurrently.

To keep resources under control, authoritative servers should proactively manage their encrypted connections. Section 6.5 of [I-D.ietf-dprive-dnsquic] ("Connection Handling") offers useful guidance for servers managing DoQ connections. Section 3.4 of [RFC7858] offers useful guidance for servers managing DoT connections.

An authoritative server facing unforeseen resource exhaustion SHOULD cleanly close open connections from recursive resolvers based on the authoritative's preferred prioritization.

In the case of unanticipated resource exhaustion, a reasonable prioritization scheme would be to close connections in this order, until resources are back in control:

- \* connections with no outstanding queries, ordered by idle time (longest idle time gets closed first)
- \* connections with outstanding queries, ordered by age of outstanding query (oldest outstanding query gets closed first)

When resources are especially tight, the authoritative server may also decline to accept new connections over encrypted transport.

#### 4. Guidance for recursive resolvers

This section outlines a probing policy suitable for unilateral adoption by any recursive resolver. Following this policy should not result in failed resolutions or significant delay.

##### 4.1. Overall recursive resolver Settings

A recursive resolver implementing this draft must set system-wide values for some default parameters. These parameters may be set independently for each supported encrypted transport, though a simple implementation may keep the parameters constant across encrypted transports.

Name	Description	Suggested Default
persistence	How long should the recursive resolver remember successful encrypted transport connections?	3 days (259200 seconds)
damping	How long should the recursive resolver remember unsuccessful encrypted transport connections?	1 day (86400 seconds)
timeout	How long should the recursive resolver wait for an initiated encrypted connection to complete?	4 seconds

Table 1: recursive resolver system parameters per encrypted transport

This document uses the notation E-foo to refer to the foo parameter for the encrypted transport E.

For example DoT-persistence would indicate the length of time that the recursive resolver will remember that an authoritative server had a successful connection over DoT.

This document also assumes that the resolver maintains a list of outstanding cleartext queries destined for the authoritative resolver's IP address X. This list is referred to as Do53-queries[X]. This document does not attempt to describe the specific operation of sending and receiving cleartext DNS queries (Do53) for a recursive resolver. Instead it describes a "bolt-on" mechanism that extends the recursive resolver's operation on a few simple hooks into the recursive resolver's existing handling of Do53.

Implementers or deployers of DNS recursive resolvers that follow the strategies in this document are encouraged to report their preferred values of these parameters.

#### 4.2. Recursive Resolver Requirements

To follow this guidance, a recursive resolver MUST implement at least one of either DoT or DoQ in its capacity as a client of authoritative nameservers.

A recursive resolver SHOULD implement the client side of DNS-over-TLS (DoT). A recursive resolver MAY implement the client side of DNS-over-QUIC (DoQ).

DoT queries from the recursive resolver MUST target TCP port 853, with an ALPN of dot. DoQ queries from the recursive resolver MUST target UDP port 853, with an ALPN of doq.

While this document focuses on the recursive-to-authoritative hop, a recursive resolver implementing these strategies SHOULD also accept queries from its clients over some encrypted transport (current common transports are DoH or DoT).

#### 4.3. Authoritative Server Encrypted Transport Connection State

The recursive resolver SHOULD keep a record of the state for each authoritative server it contacts, indexed by the IP address of the authoritative server and the encrypted transports supported by the recursive resolver.

Each record should contain the following fields for each supported encrypted transport, each of which would initially be null:

Name	Description	Retain Across Reset
session	The associated state of any existing, established session (the structure of this value is dependent on the encrypted transport implementation). If session is not null, it may be in one of two states: pending or established	N
initiated	Timestamp of most recent connection attempt	Y
completed	Timestamp of most recent completed handshake	Y
status	Enumerated value of success or fail or timeout, associated with the completed handshake	Y
resumptions	A stack of resumption tickets (and associated parameters) that could be used to resume a prior successful connection	Y
queries	A queue of queries intended for this authoritative server, each of which has additional status early, unsent, or sent	N
last-activity	A timestamp of the most recent activity on the connection	N

Table 2: recursive resolver state per authoritative IP, per encrypted transport

Note that the session fields in aggregate constitute a pool of open connections to different servers.

With the exception of the session, queries, and last-activity fields, this cache information should be kept across restart of the server unless explicitly cleared by administrative action.

This document uses the notation E-foo[X] to indicate the value of field foo for encrypted transport E to IP address X.

For example, DoT-initiated[192.0.2.4] represents the timestamp when the most recent DoT connection packet was sent to IP address 192.0.2.4.

#### 4.3.1. Separate State for Each of the Recursive Resolver's Own IP Addresses

Note that the recursive resolver should record this per-authoritative-IP state for each IP address it uses as it sends its queries. For example, if a recursive resolver can send a packet to authoritative servers from IP addresses 192.0.2.100 and 192.0.2.200, it should keep two distinct sets of per-authoritative-IP state, one for each source address it uses. Keeping these state tables distinct for each source address makes it possible for a pooled authoritative server behind a load balancer to do a partial rollout while minimizing accidental timeouts (see Section 3.1).

#### 4.4. Maintaining Authoritative State by IP Address

In designing a probing strategy, the recursive resolver could record its knowledge about any given authoritative server with different strategies, including at least:

- \* the authoritative server's IP address,
- \* the authoritative server's name (the NS record used), or
- \* the zone that contains the record being looked up.

This draft encourages the first strategy, to minimize timeouts or accidental delays.

A timeout (accidental delay) is most likely to happen when the recursive client believes that the authoritative server offers encrypted transport, but the actual server reached declines encrypted transport (or worse, filters the incoming traffic and does not even respond with an ICMP port closed message).

By associating state with the IP address, the recursive client is most able to avoid reaching a heterogenous deployment.

For example, consider an authoritative server named ns0.example.com that is served by two installations (with two A records), one at 192.0.2.7 that follows this guidance, and one at 192.0.2.8 that is a legacy (cleartext port 53-only) deployment. A recursive client who

associates state with the NS name and reaches .7 first will "learn" that ns0.example.com supports encrypted transport. A subsequent query over encrypted transport dispatched to .8 would fail, potentially delaying the response.

By associating the state with the authoritative IP address, the client can minimize the number of accidental delays introduced (see also Section 4.3.1 and Section 3.1).

#### 4.5. Probing Policy

When a recursive resolver discovers the need for an authoritative lookup to an authoritative DNS server using IP address X, it retrieves the records associated with X from its cache.

The following sections presume that the time of the discovery of the need for lookup is time T0.

If any of the records discussed here are absent, they are treated as null.

The recursive resolver must know to decide whether to initially send a query over Do53, or over any of the supported encrypted transports (DoT or DoQ).

Note that a resolver might initiate this query via any or all of the known transports. When multiple queries are sent, the initial packets for each connection can be sent concurrently, similar to "Happy Eyeballs" ([RFC8305]). However, unlike Happy Eyeballs, when one transport succeeds, the other connections do not need to be terminated, but can instead be continued to establish whether the IP address X is capable of corresponding on the relevant transport.

##### 4.5.1. Sending a query over Do53

For any of the supported encrypted transports E, if either of the following holds true, the resolver SHOULD NOT send a query to X over Do53:

- \* E-session[X] is in the established state, or
- \* E-status[X] is success, and  $(T - E-completed[X]) < persistence$

Otherwise, if there is no outstanding session for any encrypted transport, and the last successful encrypted transport connection was long ago, the resolver sends a query to X over Do53. When it does so, it inserts a handle for the query in Do53-queries[X].



#### 4.5.2. Receiving a response over Do53

When a successful response R is received in cleartext from authoritative server X for a query Q that was sent over Do53, the recursive resolver should:

- \* If Q is in Do53-queries[X]:
  - Return R to the requesting client
- \* Remove Q from Do53-queries[X]
- \* For each supported encrypted transport E:
  - If Q is in E-queries[X]:
    - o Remove Q from E-queries[X]

But if R is unsuccessful (e.g. SERVFAIL):

- \* If Q is in Do53-queries[X]:
  - Remove Q from Do53-queries[X]
- \* if Q is not in any of \*-queries[X]:
  - Return SERVFAIL to the client

#### 4.5.3. Initiating a connection over encrypted transport

If any E-session[X] is in the established, the recursive resolver SHOULD NOT initiate a new connection to X over any other transport, but should instead send a query through the existing session (see Section 4.5.8). FIXME: What if there's a preferred transport, but the established session does not correspond to that preferred transport?

Otherwise, the timer should examine and possibly refresh its state for encrypted transport E to authoritative IP address X:

- \* if E-session[X] is in state pending, and
- \* T - E-initiated[X] > E-timeout, then
  - set E-session[X] to null and
  - set E-status[X] to timeout

When resources are available to attempt a new encrypted transport, the resolver should only initiate a new connection to X over E as long as one of the following holds true:

- \* E-status[X] is success, or
- \* E-status[X] is fail or timeout and  $(T - E-completed[X]) > damping$ , or
- \* E-status[X] is null and E-initiated[X] is null

When initiating a session to X over encrypted transport E, if E-resumptions[X] is not empty, one ticket should be popped off the stack and used to try to resume a previous session. Otherwise, the initial Client Hello handshake should not try to resume any session.

When initiating a connection, the resolver should take the following steps:

- \* set E-initiated[X] to T0
- \* store a handle for the new session (which should have pending state) in E-session[X]
- \* insert a handle for the query that prompted this connection in E-queries[X], with status unsent or early, as appropriate (see below).

#### 4.5.3.1. Early Data

Modern encrypted transports like TLS 1.3 offer the chance to store "early data" from the client into the initial Client Hello in some contexts. A resolver that initiates a connection over a encrypted transport according to this guidance in a context where early data is possible SHOULD send the DNS query that prompted the connection in the early data, according to the sending guidance in Section 4.5.8.

If it does so, the status of Q in E-queries[X] should be set to early instead of unsent.

#### 4.5.3.2. Resumption Tickets

When initiating a new connection (whether by resuming an old session or not), the recursive resolver SHOULD request a session resumption ticket from the authoritative server. If the authoritative server supplies a resumption ticket, the recursive resolver pushes it into the stack at E-resumptions[X].

#### 4.5.3.3. Server Name Indication

For modern encrypted transports like TLS 1.3, most client implementations expect to send a Server Name Indication (SNI) in the Client Hello.

There are two complications with selecting or sending SNI in this unilateral probing:

- \* Some authoritative servers are known by more than one name; selecting a single name to use for a given connection may be difficult or impossible.
- \* In most configurations, the contents of the SNI field is exposed on the wire to a passive adversary. This potentially reveals additional information about which query is being made, based on the NS of the query itself.

To avoid additional leakage and complexity, a recursive resolver following this guidance SHOULD NOT send SNI to the authoritative when attempting encrypted transport.

If the recursive resolver needs to send SNI to the authoritative for some reason not found in this document, it is RECOMMENDED that it implements Encrypted Client Hello ([I-D.ietf-tls-esni]) to reduce leakage.

#### 4.5.3.4. Authoritative Server Authentication

A recursive resolver following this guidance MAY attempt to verify the server's identity by X.509 certificate or DANE. When doing so, the identity would presumably be based on the NS name used for a given query.

However, since this probing policy is unilateral and opportunistic, the client connecting under this policy MUST accept any certificate presented by the server. If the client cannot verify the server's identity, it MAY use that information for reporting, logging, or other analysis purposes. But it MUST NOT reject the connection due to the authentication failure, as the result would be falling back to cleartext, which would leak the content of the session to a passive network monitor.

#### 4.5.4. Establishing an encrypted transport connection

When an encrypted transport connection actually completes (e.g., the TLS handshake completes) at time T1, the resolver sets E-completed[X] to T1 and does the following:

If the handshake completed successfully:

- \* update E-session[X] so that it is in state established
- \* set E-status[X] to success
- \* for each query Q in E-queries[X]:
  - if early data was accepted and Q is early,
    - o set the status of Q to sent
  - otherwise:
    - o send Q through the session (see Section 4.5.8), and set the status of Q to sent
- \* set E-last-activity[X] to T1

#### 4.5.5. Failing to establish an encrypted transport connection

If, at time T2 an encrypted transport handshake completes with a failure (e.g. a TLS alert),

- \* set E-session[X] to null
- \* set E-status[X] to fail
- \* set E-completed[X] to T2
- \* for each query Q in E-queries[X]:
  - if Q is not present in any other \*-queries[X] or in Do53-queries[X], add Q to Do53-queries[X] and send query Q to X over Do53.

Note that this failure will trigger the recursive resolver to fall back to cleartext queries to the authoritative server at IP address X. It will retry encrypted transport to X once the damping timer has elapsed.

#### 4.5.6. Encrypted transport failure

Once established, an encrypted transport might fail for a number of reasons (e.g., decryption failure, or improper protocol sequence).

If this happens:

- \* set E-session[X] to null
- \* set E-status[X] to fail
- \* for each query Q in E-queries[X]:
  - if Q is not present in any other \*-queries[X] or in Do53-queries[X], add Q to Do53-queries[X] and send query Q to X over Do53. FIXME: should a resumption ticket be used here for this previously successful connection?

Note that this failure will trigger the recursive resolver to fall back to cleartext queries to the authoritative server at IP address X. It will retry encrypted transport to X once the damping timer has elapsed.

FIXME: are there specific forms of failure that we might handle differently? For example, What if a TCP timeout closes an idle DoT connection? What if a QUIC stream ends up timing out but other streams on the same QUIC connection are going through? Do the described scenarios cover the case when an encrypted transport's port is made unavailable/closed?

#### 4.5.7. Handling clean shutdown of encrypted transport connection

At time T3, the recursive resolver may find that authoritative server X cleanly closes an existing outstanding connection (most likely due to resource exhaustion, see Section 3.4).

When this happens:

- \* set E-session[X] to null
- \* for each query Q in E-queries[X]:
  - if Q is not present in any other \*-queries[X] or in Do53-queries[X], add Q to Do53-queries[X] and send query Q to X over Do53.

Note that this premature shutdown will trigger the recursive resolver to fall back to cleartext queries to the authoritative server at IP address X. Any subsequent query to X will retry the encrypted connection promptly.

#### 4.5.8. Sending a query over encrypted transport

When sending a query to an authoritative server over encrypted transport at time T4, the recursive resolver should take a few reasonable steps to ensure privacy and efficiency.

When sending query Q, the recursive resolver should ensure that its state in E-queries[X] is set to sent.

The recursive resolver also sets E-last-activity[X] to T4.

In addition, the recursive resolver should consider the following guidance:

##### 4.5.8.1. Avoid EDNS client subnet

To protect the privacy of the client, the recursive resolver SHOULD NOT send EDNS(0) Client Subnet information to the authoritative server ([RFC7871]) unless explicitly authorized to do so by the client.

##### 4.5.8.2. Pad to standard policy

To increase the anonymity set for each query, the recursive resolver SHOULD use EDNS(0) padding according to policies described in [RFC8467].

##### 4.5.8.3. Send queries in separate channels

When multiple queries are multiplexed on a single encrypted transport to a single authoritative server, the recursive resolver MUST offer distinct query ID fields for every outstanding query on a connection, and MUST be capable of receiving responses out of order.

To the extent that the encrypted transport can avoid head-of-line blocking (e.g. QUIC can use a separate stream per query) the recursive resolver SHOULD avoid head-of-line blocking.

#### 4.5.9. Receiving a response over encrypted transport

When a response R for query Q arrives at the recursive resolver over encrypted transport E from authoritative server with IP address X at time T5, if Q is in E-queries[X], the recursive resolver takes the following steps:

- \* Remove R from E-queries[X]
- \* Set E-last-activity[X] to T5

- \* If R is successful:
  - send R to the requesting client
  - For each supported encrypted transport N other than E:
    - o If Q is in N-queries[X]:
      - + Remove Q from N-queries[X]
  - If Q is in Do53-queries[X]:
    - o Remove Q from Do53-queries[X]
- \* Otherwise (R is unsuccessful, e.g., SERVFAIL):
  - If Q is not in Do53-queries[X] or any other \*-queries[X]:
    - o Return SERVFAIL to the requesting client FIXME: What response should be sent to the clients in the case that extended DNS errors are used in an authoritative's response?

#### 4.5.10. Resource Exhaustion

To keep resources under control, a recursive resolver should proactively manage outstanding encrypted connections. Section 6.5 of [I-D.ietf-dprive-dnssoquic] ("Connection Handling") offers useful guidance for clients managing DoQ connections. Section 3.4 of [RFC7858] offers useful guidance for clients managing DoT connections.

Even with sensible connection management, a recursive resolver doing unilateral probing may find resources unexpectedly scarce, and may need to close some outstanding connections.

In such a situation, the recursive resolver SHOULD use a reasonable prioritization scheme to close outstanding connections.

One reasonable prioritization scheme would be:

- \* close outstanding established sessions based on E-last-activity[X] (oldest timestamp gets closed first)

Note that when resources are limited, a recursive resolver following this guidance may also choose not to initiate new connections for encrypted transport.

#### 4.5.11. Maintaining connections

Some recursive resolvers looking to amortize connection costs, and to minimize latency MAY choose to synthesize queries to a particular resolver to keep a encrypted transport session active.

A recursive resolver that adopts this approach should try to align the synthesized queries with other optimizations. For example, a recursive resolver that "pre-fetches" a particular resource record to keep its cache "hot" can send that query over an established encrypted transport session.

### 5. Signalling for Stronger Defense

This draft does not contemplate the specification of any form of coordinated signalling between authoritative servers and recursive resolvers, as such measures would not be unilateral.

However, the draft highlights the needs of a signaling mechanism for stronger defense.

We highlight the following questions for other specifications to solve:

- \* What does the signal need to contain?
  - type of transport? (DoQ? DoT? DoH?)
  - error reporting if secure, authenticated connection fails (how to report? similar to TLSRPT?)
  - whether to hard-fail if encrypted communication isn't available
  - cryptographic authentication of authoritative server (e.g. pubkeys) vs. names vs. domain?
- \* How should the signal be presented?
  - SVCB RR or "surprising" DS RR
- \* How should the signal be scoped?
  - per-nameserver (by NS), per-nameserver (by IP address, via in-addr.arpa), or per-domain?



### 5.1. Combining Signals with Opportunistic Probing

FIXME: How do the signals get combined with the above opportunistic probing policy? Can we specify that without needing to specify the signalling mechanism itself?

## 6. IANA Considerations

IANA does not need to do anything for implementers to adopt the guidance found in this draft.

## 7. Privacy Considerations

### 7.1. Server Name Indication

A recursive resolver querying an authoritative server over DoT or DoQ that sends Server Name Indication (SNI) in the clear in the cryptographic handshake leaks information about the intended query to a passive network observer.

In particular, if two different zones refer to the same nameserver IP addresses via differently-named NS records, a passive network observer can distinguish queries to one zone from the queries to the other.

Omitting SNI entirely, or using ECH to hide the intended SNI, avoids this additional leakage. However, a series of queries that leak this information is still an improvement over the all-cleartext status quo at the time of this document.

## 8. Security Considerations

The guidance in this draft provides defense against passive network monitors for most queries. It does not defend against active attackers. It can also leak some queries and their responses due to "happy eyeballs" optimizations when the resolver's cache is cold.

Implementation of the guidance in this draft should increase deployment of opportunistic encrypted DNS transport between recursive resolvers and authoritative servers at little operational risk.

However, implementers should not rely on the guidance in this draft for robust defense against active attackers, but should treat it as a stepping stone en route to stronger defense.

In particular, a recursive resolver following this guidance can easily be forced by an active attacker to fall back to cleartext DNS queries. Or, an active attacker could position itself as a machine-

in-the-middle, which the recursive resolver would not defend against or detect due to lack of server authentication. Defending against these attacks without risking additional unexpected protocol failures would require signalling and coordination that are out of scope for this draft.

This guidance is only one part of operating a privacy-preserving DNS ecosystem. A privacy-preserving recursive resolver should adopt other practices as well, such as QNAME minimization, local root zone, etc, to reduce the overall leakage of query information that could infringe on the client's privacy.

## 9. Acknowledgements

Many people contributed to the development of this draft beyond the authors, including Brian Dickson, Christian Huitema, Eric Nygren, Jim Reid, Kris Shrishak, Paul Hoffman, Ralf Weber, Robert Evans, and the DPRIVE working group.

## 10. References

### 10.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.
- [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/info/rfc8174>>.

### 10.2. Informative References

- [RFC1035] Mockapetris, P., "Domain names - implementation and specification", STD 13, RFC 1035, DOI 10.17487/RFC1035, November 1987, <<https://www.rfc-editor.org/info/rfc1035>>.
- [I-D.ietf-dprive-dnssoquic] Huitema, C., Dickinson, S., and A. Mankin, "DNS over Dedicated QUIC Connections", Work in Progress, Internet-Draft, draft-ietf-dprive-dnssoquic-08, 11 January 2022, <<https://www.ietf.org/archive/id/draft-ietf-dprive-dnssoquic-08.txt>>.
- [I-D.ietf-tls-esni] Rescorla, E., Oku, K., Sullivan, N., and C. A. Wood, "TLS Encrypted Client Hello", Work in Progress, Internet-Draft,

draft-ietf-tls-esni-13, 12 August 2021,  
<<https://www.ietf.org/archive/id/draft-ietf-tls-esni-13.txt>>.

- [RFC7435] Dukhovni, V., "Opportunistic Security: Some Protection Most of the Time", RFC 7435, DOI 10.17487/RFC7435, December 2014, <<https://www.rfc-editor.org/info/rfc7435>>.
- [RFC7858] Hu, Z., Zhu, L., Heidemann, J., Mankin, A., Wessels, D., and P. Hoffman, "Specification for DNS over Transport Layer Security (TLS)", RFC 7858, DOI 10.17487/RFC7858, May 2016, <<https://www.rfc-editor.org/info/rfc7858>>.
- [RFC7871] Contavalli, C., van der Gaast, W., Lawrence, D., and W. Kumari, "Client Subnet in DNS Queries", RFC 7871, DOI 10.17487/RFC7871, May 2016, <<https://www.rfc-editor.org/info/rfc7871>>.
- [RFC8305] Schinazi, D. and T. Pauly, "Happy Eyeballs Version 2: Better Connectivity Using Concurrency", RFC 8305, DOI 10.17487/RFC8305, December 2017, <<https://www.rfc-editor.org/info/rfc8305>>.
- [RFC8467] Mayrhofer, A., "Padding Policies for Extension Mechanisms for DNS (EDNS(0))", RFC 8467, DOI 10.17487/RFC8467, October 2018, <<https://www.rfc-editor.org/info/rfc8467>>.
- [RFC8484] Hoffman, P. and P. McManus, "DNS Queries over HTTPS (DoH)", RFC 8484, DOI 10.17487/RFC8484, October 2018, <<https://www.rfc-editor.org/info/rfc8484>>.

#### Appendix A. Document Considerations

[ RFC Editor: please remove this section before publication ]

This document is currently edited as markdown. Minor editorial changes can be suggested via merge requests at <https://gitlab.com/dkg/dprive-unilateral-probing> or by e-mail to the editor. Please direct all significant commentary to the public IETF DPRIVE mailing list: [dprive@ietf.org](mailto:dprive@ietf.org)

The authors' latest draft can be read online in html (<https://dkg.gitlab.io/dprive-unilateral-probing/>) or pdf (<https://dkg.gitlab.io/dprive-unilateral-probing/unilateral-probing.pdf>) or text (<https://dkg.gitlab.io/dprive-unilateral-probing/unilateral-probing.txt>) formats.

## A.1. Document History

### A.1.1. Substantive changes from -01 to -02

- \* Clarify that deployment to a pool does not need to be strictly simultaneous
- \* Explain why authoritatives need to serve the same records regardless of SNI
- \* Defer to external, protocol-specific references for resource management
- \* Clarify that probed connections must not fail due to authentication failure

### A.1.2. Substantive changes from -00 to -01

- \* Fallback to cleartext when encrypted transport fails.
- \* Reduce default timeout to 4s
- \* Clarify SNI guidance: OK for selecting server credentials, not OK for changing answers
- \* Document ALPN and port numbers
- \* Justify sorting recursive resolver state by authoritative IP address

## Authors' Addresses

Daniel Kahn Gillmor  
American Civil Liberties Union  
125 Broad St.  
New York, NY, 10004  
United States of America  
  
Email: dkg@fifthhorseman.net

Joey Salazar  
ARTICLE 19  
108-114 Golden Lane  
London  
EC1Y 0TL  
United Kingdom

Email: [joeysal@gmail.com](mailto:joeysal@gmail.com)

Network Working Group  
Internet-Draft  
Intended status: Experimental  
Expires: 1 April 2022

P. Hoffman  
ICANN  
P. van Dijk  
PowerDNS  
28 September 2021

Recursive to Authoritative DNS with Unauthenticated Encryption  
draft-ietf-dprive-unauth-to-authoritative-04

Abstract

This document describes a use case and a method for a DNS recursive resolver to use unauthenticated encryption when communicating with authoritative servers. The motivating use case for this method is that more encryption on the Internet is better, and some resolver operators believe that unauthenticated encryption is better than no encryption at all. The method described here is optional for both the recursive resolver and the authoritative server.

Status of This Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <https://datatracker.ietf.org/drafts/current/>.

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

This Internet-Draft will expire on 1 April 2022.

Copyright Notice

Copyright (c) 2021 IETF Trust and the persons identified as the document authors. All rights reserved.

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

## Table of Contents

1. Introduction . . . . .	2
1.1. Use Case for Unauthenticated Encryption . . . . .	3
1.2. Summary of Protocol . . . . .	3
1.3. Definitions . . . . .	4
2. Discovery of Authoritative Server Encryption . . . . .	4
3. Processing Discovery Responses . . . . .	5
3.1. Resolver Process as Pseudocode . . . . .	6
3.2. Resolver Session Failures . . . . .	7
4. Serving with Encryption . . . . .	8
5. IANA Considerations . . . . .	8
6. Security Considerations . . . . .	8
7. Acknowledgements . . . . .	9
8. References . . . . .	9
8.1. Normative References . . . . .	9
8.2. Informative References . . . . .	10
Authors' Addresses . . . . .	11

## 1. Introduction

A recursive resolver using traditional DNS over port 53 may wish instead to use encrypted communication with authoritative servers in order to limit snooping of its DNS traffic by passive or on-path attackers. The recursive resolver can use unauthenticated encryption (defined in [OPPORTUN]) to achieve this goal.

This document describes the use case for unauthenticated encryption in recursive resolvers in Section 1.1. The encryption method with authoritative servers can be DNS-over-TLS [DNS-OVER-TLS] (DoT), DNS-over-HTTPS [DNS-OVER-HTTPS] (DoH), and/or DNS-over-QUIC [DNS-OVER-QUIC] (DoQ).

The document also describes a discovery method that shows if an authoritative server supports encryption in Section 2.

See [FULL-AUTH] for a description of the use case and a proposed mechanism for fully-authenticated encryption.

NOTE: The draft uses the SVCB record as a discovery mechanism for encryption by a particular authoritative server. Any record type that can show multiple types of encryption (currently DoT, DoH, and DoQ) can be used for discovery. Thus, this record type might change in the future, depending on the discussion in the DPRIVE WG.

### 1.1. Use Case for Unauthenticated Encryption

The use case in this document for unauthenticated encryption is recursive resolver operators who are happy to use encryption with authoritative servers if doing so doesn't significantly slow down getting answers, and authoritative server operators that are happy to use encryption with recursive resolvers if it doesn't cost much. In this use case, resolvers do not want to return an error for requests that were sent over an encrypted channel if they would have been able to give a correct answer using unencrypted transport. Ultimately, this effort has two goals: to protect queries from failing in case authenticated encryption is not available, and to enable recursive resolver operators to encrypt without server authentication.

Resolvers and authoritative servers understand that using encryption costs something, but are willing to absorb the costs for the benefit of more Internet traffic being encrypted. The extra costs (compared to using traditional DNS on port 53) include:

- \* Extra round trips to establish TCP for every session (but not necessarily for every query)
- \* Extra round trips for TLS establishment
- \* Greater CPU use for TLS establishment
- \* Greater CPU use for encryption after TLS establishment
- \* Greater memory use for holding TLS state

This use case is not expected to apply to all resolvers or authoritative servers. For example, according to [RSO\_STATEMENT], some root server operators do not want to be the early adopters for DNS with encryption. The protocol in this document explicitly allows authoritative servers to signal when they are ready to begin offering DNS with encryption.

### 1.2. Summary of Protocol

This summary gives an overview of how the parts of the protocol work together.



- \* The resolver discovers whether any authoritative server of interest supports DNS with encryption by querying for the SVCB records [SVCB]. As described in [DNS-SVCB], SVCB records can indicate that a server supports encrypted transport of DNS queries.

NOTE: In this document, the term "SVCB record" is used only for SVCB records that indicate encryption as described in [DNS-SVCB]. SVCB records that do not have these indicators in the RDATA are not included in the term "SVCB record" in this document.

- \* The resolver uses any authoritative server with a SVCB record that indicates encryption to perform unauthenticated encryption.
- \* The resolver does not fail to set up encryption if server authentication in the TLS session fails.

### 1.3. Definitions

The terms "recursive resolver", "authoritative server", and "classic DNS" are defined in [DNS-TERM].

"DNS with encryption" means transport of DNS over any of DoT, DoH, or DoQ. A server that supports DNS with encryption supports transport over one or more of DoT, DoH, or DoQ.

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 [MUST-SHOULD-1] [MUST-SHOULD-2] when, and only when, they appear in all capitals, as shown here.

## 2. Discovery of Authoritative Server Encryption

An authoritative server that supports DNS with encryption makes itself discoverable by publishing one or more DNS SVCB records that contain "alpn" parameter keys. SVCB records are defined in [SVCB], and the DNS extension to those records is defined in [DNS-SVCB].

A recursive resolver discovers whether an authoritative server supports DNS with encryption by looking for cached SVCB records for the name of the authoritative server with a positive answer. A cached DNS SVCB record with a negative answer indicates that the authoritative server does not support any encrypted transport.

A resolver MAY also use port probing, although the mechanism for that is not described here.

If the cache has no positive or negative answers for any SVCB record for any of a zone's authoritative servers, the resolver MAY send queries for the SVCB records (and for the A/AAAA records of names mentioned in those SVCB records) for some or all of the zone's authoritative servers and wait for a positive response so that the resolver can use DNS with encryption for the original query. In this situation, the resolver MAY instead just use classic DNS for the original query but simultaneously queue queries for the SVCB (and subsequent A/AAAA) records for some or all of the zone's authoritative servers so that future queries might be able to use DNS with encryption.

DNSSEC validation of SVCB RRsets used strictly for this discovery mechanism is not mandated.

### 3. Processing Discovery Responses

After a resolver has DNS SCVB records in its cache (possibly due to having just queried for them), it needs to use those records to try to find an authoritative server that uses DNS with encryption. This section describes how the resolver can make that selection.

A resolver MUST NOT attempt encryption for a server that has a negative response in its cache for the associated DNS SVCB record.

After sending out all requests for SVCB records for the authoritative servers in the NS RRset for a name, if all of the SVCB records for those authoritative servers in the cache are negative responses, the resolver MUST use classic (unencrypted) DNS instead of encryption. Similarly, if none of the DNS SVCB records for the authoritative servers in the cache have supported "alpn" parameters, the resolver MUST use classic (unencrypted) DNS instead of encryption.

If there are any DNS SVCB records in the cache for the authoritative servers for a zone with supported "alpn" parameters, the resolver MUST try each indicated authoritative server using DNS with encryption until it successfully sets up a connection. The resolver attempts to use the encrypted transports that are in the associated SVCB record for the authoritative server.

A resolver SHOULD keep a DNS with encryption session to a particular server open if it expects to send additional queries to that server in a short period of time. [DNS-OVER-TCP] says "both clients and servers SHOULD support connection reuse" for TCP connections, and that advice could apply as well for DNS with encryption, especially as DNS with encryption has far greater overhead for re-establishing a connection. If the server closes the DNS with encryption session, the resolver can possibly re-establish a DNS with encryption session

using encrypted session resumption. Configuration for the maximum timeout, minimum timeout, and duration of encrypted sessions should take into consideration the recommendations given in [TCP-TIMEOUT], [EDNS-TCP], and (for DoH) [HTTP-1.1].

For any DNS with encryption protocols, TLS version 1.3 [TLS-13] or later MUST be used.

A resolver following this protocol does not need to authenticate TLS servers. Thus, when setting up a TLS connection, if the server's authentication credentials do not match those expected by the resolver, the resolver continues with the TLS connection. Privacy-oriented resolvers (defined in [PRIVACY-REC]) following this protocol MUST NOT indicate that they are using encryption because this protocol is susceptible to on-path attacks.

If the resolver gets a TLS failure (such as those listed in Section 3.2, the resolver instead uses classic DNS on any of the authoritative servers.

### 3.1. Resolver Process as Pseudocode

This section is meant as an informal clarification of the protocol, and is not normative. The pseudocode here is designed to show the intent of the protocol, so it is not optimized for things like intersection of sets and other shortcuts.

In this code, `signal_rrset(this_name)` means an SVCB query for the `'_dns'` prefix of `this_name`. The Query over secure transport until successful section ignores differences in name server selection and retry behaviour in different resolvers.

```
# Inputs
ns_names = List of NS Rdatas from the NS RRset for the queried name
can_do_secure = List of secure transports supported by resolver
secure_names_and_transports = Empty list, filled in below

# Fill secure_names_and_transports with (name, transport) tuples
for this_name in ns_names:
    if signal_rrset(this_name) is in the resolver cache:
        if signal_rrset(this_name) positively does not exist:
            continue
        for this_transport in signal_rrset(this_name):
            if this_transport in can_do_secure:
                add (this_name, this_transport) to secure_names_and_transports
    else: # signal_rrset(this_name) is not in the resolver cache
        queue a query for signal_rrset(this_name) for later caching

# Query over secure transport until successful
for (this_name, this_transport) tuple in secure_names_and_transports:
    query using this_transport on this_name
    if successful:
        finished

# Got here if no this_name/this_transport query was successful
# or if secure_names_and_transports was empty
query using classic DNS; finished
```

### 3.2. Resolver Session Failures

The following are some of the reasons that a DNS with encryption session might fail to be set up:

- \* The resolver receives a TCP RST response
- \* The resolver does not receive replies to TCP or TLS setup (such as getting the TCP SYN message, the first TLS message, or completing TLS handshakes)
- \* The TLS handshake gets a definitive failure
- \* The encrypted session fails for reasons other than for authentication, such as incorrect algorithm choices or TLS record failures

#### 4. Serving with Encryption

An operator of an authoritative server following this protocol SHOULD publish SVCB records as described in Section 2. If they cannot publish such records, the security properties of their authoritative servers will not be found. If an operator wants to test serving using encryption, they can publish SVCB records with short TTLs and then stop serving with encryption after removing the SVCB records and waiting for the TTLs to expire.

It is acceptable for an operator of authoritative servers to only offer encryption on some of the named authoritative servers, such as when the operator is determining how far to roll out encrypted service.

A server MAY close an encrypted connection at any time. For example, it can close the session if it has not received a DNS query in a defined length of time. The server MAY close an encrypted session after it sends a DNS response; however, it might also want to keep the session open waiting for another DNS query from the resolver. [DNS-OVER-TCP] says "both clients and servers SHOULD support connection reuse" for TCP connections, and that advice could apply as well for DNS with encryption, especially as DNS with encryption has far greater overhead for re-establishing a connection. If the server closes the DNS with encryption session, the resolver can possibly re-establish a DNS with encryption session using encrypted session resumption.

For any DNS with encryption protocols, TLS version 1.3 [TLS-13] or later MUST be used.

#### 5. IANA Considerations

(( Update registration for TCP/853 to also include ADoT ))

(( Maybe other updates for DoH and DoQ ))

#### 6. Security Considerations

The method described in this document explicitly allows a resolver to perform DNS communications over traditional unencrypted, unauthenticated DNS on port 53, if it cannot find an authoritative server that advertises that it supports encryption. The method described in this document explicitly allows a resolver using encryption to choose to allow unauthenticated encryption. In either of these cases, the resulting communication will be susceptible to obvious and well-understood attacks from an attacker in the path of the communications.

[TLS-1.3] specifically warns against anonymous connections because such connections only provide protection against passive eavesdropping while failing to protect against active on-path attacks. Section C.5 of [TLS-1.3] explicitly states applications MUST NOT use TLS with unverifiable server authentication unless there is explicit configuration or a specific application profile to do so. This document is such an application profile.

Encrypting the traffic between resolvers and authoritative servers does not solve all the privacy issues for resolution. See [PRIVACY-REC] and [PRIVACY-CONS] for in-depth discussion of the associated privacy issues.

## 7. Acknowledgements

Puneet Sood contributed many ideas to early drafts of this document.

The DPRIVE Working Group has contributed many ideas that keep shifting the focus and content of this document.

## 8. References

### 8.1. Normative References

[DNS-SVCB] Schwartz, B., "Service Binding Mapping for DNS Servers", Work in Progress, Internet-Draft, draft-schwartz-svcb-dns-04, 26 July 2021, <<https://www.ietf.org/archive/id/draft-schwartz-svcb-dns-04.txt>>.

[DNS-TERM] Hoffman, P. and K. Fujiwara, "DNS Terminology", Work in Progress, Internet-Draft, draft-ietf-dnsop-rfc8499bis-03, 28 September 2021, <<https://www.ietf.org/archive/id/draft-ietf-dnsop-rfc8499bis-03.txt>>.

[FULL-AUTH] Pauly, T., Rescorla, E., Schinazi, D., and C. A. Wood, "Signaling Authoritative DNS Encryption", Work in Progress, Internet-Draft, draft-rescorla-dprive-adox-latest-00, 26 February 2021, <<https://www.ietf.org/archive/id/draft-rescorla-dprive-adox-latest-00.txt>>.

[MUST-SHOULD-1] 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/info/rfc2119>>.

## [MUST-SHOULD-2]

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/info/rfc8174>>.

[OPPORTUN] Dukhovni, V., "Opportunistic Security: Some Protection Most of the Time", RFC 7435, DOI 10.17487/RFC7435, December 2014, <<https://www.rfc-editor.org/info/rfc7435>>.

[SVCB] Schwartz, B., Bishop, M., and E. Nygren, "Service binding and parameter specification via the DNS (DNS SVCB and HTTPS RRs)", Work in Progress, Internet-Draft, draft-ietf-dnsop-svcb-https-07, 5 August 2021, <<https://www.ietf.org/archive/id/draft-ietf-dnsop-svcb-https-07.txt>>.

[TLS-13] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018, <<https://www.rfc-editor.org/info/rfc8446>>.

## 8.2. Informative References

## [DNS-OVER-HTTPS]

Hoffman, P. and P. McManus, "DNS Queries over HTTPS (DoH)", RFC 8484, DOI 10.17487/RFC8484, October 2018, <<https://www.rfc-editor.org/info/rfc8484>>.

## [DNS-OVER-QUIC]

Huitema, C., Dickinson, S., and A. Mankin, "Specification of DNS over Dedicated QUIC Connections", Work in Progress, Internet-Draft, draft-ietf-dprive-dnsquic-04, 3 September 2021, <<https://www.ietf.org/archive/id/draft-ietf-dprive-dnsquic-04.txt>>.

## [DNS-OVER-TCP]

Dickinson, J., Dickinson, S., Bellis, R., Mankin, A., and D. Wessels, "DNS Transport over TCP - Implementation Requirements", RFC 7766, DOI 10.17487/RFC7766, March 2016, <<https://www.rfc-editor.org/info/rfc7766>>.

## [DNS-OVER-TLS]

Hu, Z., Zhu, L., Heidemann, J., Mankin, A., Wessels, D., and P. Hoffman, "Specification for DNS over Transport Layer Security (TLS)", RFC 7858, DOI 10.17487/RFC7858, May 2016, <<https://www.rfc-editor.org/info/rfc7858>>.

- [EDNS-TCP] Wouters, P., Abley, J., Dickinson, S., and R. Bellis, "The edns-tcp-keepalive EDNS0 Option", RFC 7828, DOI 10.17487/RFC7828, April 2016, <<https://www.rfc-editor.org/info/rfc7828>>.
- [HTTP-1.1] Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing", RFC 7230, DOI 10.17487/RFC7230, June 2014, <<https://www.rfc-editor.org/info/rfc7230>>.
- [PRIVACY-CONS] Wicinski, T., Ed., "DNS Privacy Considerations", RFC 9076, DOI 10.17487/RFC9076, July 2021, <<https://www.rfc-editor.org/info/rfc9076>>.
- [PRIVACY-REC] Dickinson, S., Overeinder, B., van Rijswijk-Deij, R., and A. Mankin, "Recommendations for DNS Privacy Service Operators", BCP 232, RFC 8932, DOI 10.17487/RFC8932, October 2020, <<https://www.rfc-editor.org/info/rfc8932>>.
- [RSO\_STATEMENT] "Statement on DNS Encryption", 2021, <[https://root-servers.org/media/news/Statement\\_on\\_DNS\\_Encryption.pdf](https://root-servers.org/media/news/Statement_on_DNS_Encryption.pdf)>.
- [TCP-TIMEOUT] Kristoff, J. and D. Wessels, "DNS Transport over TCP - Operational Requirements", Work in Progress, Internet-Draft, draft-ietf-dnsop-dns-tcp-requirements-12, 18 August 2021, <<https://www.ietf.org/archive/id/draft-ietf-dnsop-dns-tcp-requirements-12.txt>>.
- [TLS-1.3] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018, <<https://www.rfc-editor.org/info/rfc8446>>.

#### Authors' Addresses

Paul Hoffman  
ICANN

Email: [paul.hoffman@icann.org](mailto:paul.hoffman@icann.org)

Peter van Dijk  
PowerDNS

Email: [peter.van.dijk@powerdns.com](mailto:peter.van.dijk@powerdns.com)



dprive  
Internet-Draft  
Intended status: Standards Track  
Expires: 20 February 2022

B. Schwartz  
Google LLC  
19 August 2021

Authenticated delegation information using DS records  
draft-schwartz-ds-glue-02

## Abstract

This draft describes a mechanism for conveying arbitrary authenticated DNS data from a parent nameserver to a recursive resolver as part of a delegation response.

## Discussion Venues

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

Discussion of this document takes place on the mailing list ([ds@ietf.org](mailto:ds@ietf.org)), which is archived at <https://mailarchive.ietf.org/arch/browse/ds/>.

Source for this draft and an issue tracker can be found at <https://github.com/bemasc/ds-glue>.

## Status of This Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <https://datatracker.ietf.org/drafts/current/>.

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

This Internet-Draft will expire on 20 February 2022.

## Copyright Notice

Copyright (c) 2021 IETF Trust and the persons identified as the document authors. All rights reserved.

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

## Table of Contents

1. Conventions and Definitions . . . . .	2
2. Background . . . . .	3
2.1. Obstacle 1: Authentication . . . . .	3
2.2. Obstacle 2: Flexibility . . . . .	3
3. Proposal . . . . .	3
3.1. Encoding . . . . .	4
3.2. Interpretation . . . . .	5
3.3. Allowed RR types . . . . .	6
4. Examples . . . . .	6
4.1. Out-of-bailiwick referral . . . . .	6
4.2. In-bailiwick referral . . . . .	7
4.3. In-bailiwick referral without IPv4 . . . . .	7
4.4. Delegation with authenticated encryption . . . . .	8
4.4.1. Disabling DANE . . . . .	8
5. Security Considerations . . . . .	8
6. Operational Considerations . . . . .	8
6.1. Compatibility with existing resolvers . . . . .	8
6.2. Publishing DSGLUE records . . . . .	9
6.3. Referral response size . . . . .	9
6.4. PKI and DANE for Authenticated Encryption . . . . .	9
7. IANA Considerations . . . . .	10
8. References . . . . .	10
8.1. Normative References . . . . .	11
8.2. Informative References . . . . .	11
Acknowledgments . . . . .	12
Author's Address . . . . .	12

## 1. Conventions and Definitions

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.

## 2. Background

The DPRIVE working group has been pursuing designs for authenticated encryption of recursive-to-authoritative communication. Recursive resolvers could enable authenticated encryption most easily and efficiently if they received authenticated information about the target nameserver's configuration during the in-bailiwick delegation that precedes the direct connection. However, there are several obstacles to this.

### 2.1. Obstacle 1: Authentication

Glue records in DNS referral responses are unauthenticated. Parents do not generally provide RRSIGs for these records in their responses, and resolvers do not expect such signatures to be present. An in-path attacker can modify or remove records in the delegation response without detection.

If the parent zone also implements authenticated encryption, this provides sufficient protection for the glue records, but many important parent zones seem unlikely to implement authenticated encryption in the near future.

### 2.2. Obstacle 2: Flexibility

Existing nameserver deployments assume that the delegation response includes only a fixed set of existing RR types (NS, A, AAAA, DS, RRSIG, etc.). These systems are slow to upgrade, and the working group would like to be able to begin deploying authenticated encryption without first requiring a significant change in these parents.

## 3. Proposal

This draft proposes a way to convey a glue RRSets inside a DS record, enabling authenticated delivery of arbitrary RR types as part of the delegation response.

There are three main records or RRSets involved in this process:

- \* A Source RRSets to be conveyed, which may be of any RR type and anywhere below the zone cut.
- \* A Virtual DNSKEY Record encapsulating the Source RRSets.
- \* The DSGLUE Record, a DS record derived from the Virtual DNSKEY Record and published in the parent.

### 3.1. Encoding

To encode a Source RRSset, a zone operator first transforms it into a Virtual DNSKEY Record as follows:

- \* Owner Name = The Owner Name of the Source RRSset relative to the child zone apex.
- \* Flags = 0x0001, i.e. only SEP (bit 15) is set.
- \* Protocol = 3
- \* Algorithm = DS Glue (see IANA registration in Section 7)
- \* Public Key = The following fields, concatenated
  - The RR type (uint16)
  - The RRSset TTL (uint32)
  - For each Source Record in canonical order ([RFC4034], Section 6.3),
    - o A length prefix (uint16)
    - o The canonicalized RDATA ([RFC4034], Section 6.2).

For example, this Source RRSset:

```
$ORIGIN example.com.
@ 3600 IN NS ns1
      IN NS ns2
      IN NS NS.OTHER.EXAMPLE.
```

would be represented as the following Virtual DNSKEY Record:

```
; Public Key =
; \000\002 ; Type = NS
; \000\000\014\016 ; TTL=3600
; \000\018 \002ns\005other\007example\000 ; Len=18, ns.other.example.
; \000\017 \003ns1\007example\003com\000 ; Len=17, ns1.example.com.
; \000\017 \003ns2\007example\003com\000 ; Len=17, ns2.example.com.

. 300 IN DNSKEY 1 3 $DSGLUE_NUM ( AAIAAA4QABICbnMFb3RoZXIHZXhhbXBsZ
  QAAEQNuczEHZXhhbXBsZQNjb20AABEDbnMyB2V4YW1wbGUDY29tAA== )
```

Note that:

- \* The NS Source Records are "real" records that appear in authoritative Answers and/or delegation glue, but the DNSKEY record is a "virtual record" because it does not appear in any zone or response (in this form).
- \* The Virtual DNSKEY Record's owner name is "." because the Source RRSset appears at the zone apex.
- \* The NS RDATA has been reordered and converted to lowercase as specified by the canonicalization algorithm.

Having constructed a Virtual DNSKEY Record, the DSGLUE Record is constructed as usual, but always using the VERBATIM digest type [I-D.draft-vandijk-dnsop-ds-digest-verbatim]. Thus, the DSGLUE Record's wire format RDATA forms the following concatenation:

```
Key Tag | Algorithm = DSGLUE | Digest Type = VERBATIM | Digest = (
  DNSKEY owner name = name prefix | DNSKEY RDATA = (
    Flags = 1 | Protocol = 3 | Algorithm = DSGLUE | Public Key = (
      RR Type | TTL | Len(1) | RDATA(1) | Len(2) | RDATA(2) | ...
    )
  )
)
```

The DSGLUE record is a real DS record that appears in the usual DS RRSset, whose owner name is the child apex.

QUESTION: Should we skip the virtual DNSKEY record, and construct the fake DS directly? This would save 4-6 bytes per RRSset, but would lose the ability to reuse DNSKEY->DS construction codepaths (unchanged except for a new digest type).

### 3.2. Interpretation

Upon receiving a delegation response, resolvers implementing this specification SHALL compute the Adjusted Delegation Response as follows:

1. Copy the delegation response.
2. Reverse the encoding process of any DSGLUE records to reconstruct the source RRSsets.
3. Add each of these reconstructed RRSsets to the Adjusted Delegation Response, replacing any RRSset with the same owner name and type.

Note that a Source RRSset MAY be empty, indicating that there are no records of the corresponding type at this name. After reconstructing an empty Source RRSset, recipients MUST remove any matching RRSets from the Adjusted Delegation Response and any glue cache, and MAY cache the negative result for the indicated TTL.

Resolution then proceeds as usual, using the Adjusted Delegation Response. When processing the DS RRSset, the recipient will verify the DS RRSIGs as usual, and abort the resolution as Bogus if DNSSEC validation fails.

Resolvers that do not implement this specification will ignore the DSGLUE records due to the unrecognized algorithm. Thus, these records are safe to use for both signed and unsigned child zones.

Source Records reconstructed from DSGLUE SHOULD be processed exactly like ordinary unauthenticated glue records. For example, they MAY be cached for use in future delegations but MUST NOT be returned in any responses (c.f. Section 5.4.1 of [RFC2181]).

### 3.3. Allowed RR types

DSGLUE records are capable of containing any record type. However, the meaning of certain record types (e.g. NSEC) is not yet clear in the DSGLUE context. To avoid ambiguity, child zones MUST only publish DSGLUE records containing RR types that have been registered for use with DSGLUE (Section 7), and recipients MUST ignore DSGLUE records indicating unexpected record types.

Recipients implementing this specification MUST accept the NS, A, and AAAA RR types in DSGLUE. Support for the other allowed RR types is OPTIONAL.

Recipients MUST ignore any unauthenticated TLSA records.

## 4. Examples

For these examples, the macro "\$DSGLUE(prefix, RR type, TTL, [RDATAs])" constructs a DSGLUE DS record as described in Section 3.1.

### 4.1. Out-of-bailiwick referral

An out-of-bailiwick referral contains only NS records, e.g.

```
$ORIGIN com.  
example 3600 IN NS ns1.example.net.  
           IN NS ns2.example.net.
```

These Source Records would be encoded in DSGLUE as:

```
$ORIGIN com.
example 3600 IN DS $DSGLUE(., NS, 3600,
    [ns1.example.net., ns2.example.net.] )
```

#### 4.2. In-bailiwick referral

An in-bailiwick referral contains NS records and at least one kind of address record.

```
$ORIGIN com.
example 3600 IN NS      ns1.example
                        IN NS      ns2.example
ns1.example 600 IN A      192.0.2.1
                        IN AAAA     2001:db8::1
ns2.example 600 IN A      192.0.2.2
                        IN AAAA     2001:db8::2
```

These records would be encoded in DSGLUE as:

```
$ORIGIN com.
example 600 IN DS $DSGLUE(., NS, 3600, [ns1.example.com.,
                                         ns2.example.com.])
                        IN DS $DSGLUE(ns1., A, 600, [192.0.2.1])
                        IN DS $DSGLUE(ns1., AAAA, 600, [2001:db8::1])
                        IN DS $DSGLUE(ns2., A, 600, [192.0.2.1])
                        IN DS $DSGLUE(ns2., AAAA, 600, [2001:db8::2])
```

#### 4.3. In-bailiwick referral without IPv4

Consider a delegation to a nameserver that is only reachable with IPv6:

```
$ORIGIN com.
example 3600 IN NS      ns1.example
ns1.example 600 IN AAAA  2001:db8::1
```

A zone in this configuration can optionally use an empty DSGLUE record to indicate that there is no IPv4 address:

```
$ORIGIN com.
example 600 IN DS $DSGLUE(., NS, 3600, [ns1.example.com.])
                        IN DS $DSGLUE(ns1., AAAA, 600, [2001:db8::1])
                        IN DS $DSGLUE(ns1., A, 7200, [])
```

This arrangement prevents an adversary from inserting forged A records for ns1.example.com into the delegation response.

Note that this negative answer is treated as glue that only applies during delegation, so A records for ns1.example.com can still be resolved if they exist.

#### 4.4. Delegation with authenticated encryption

Assuming a SVCB-based signaling mechanism similar to [I-D.draft-schwartz-svcb-dns], an in-bailiwick referral with support for authenticated encryption is indicated as follows:

```
$ORIGIN com.  
example 600 IN DS $DSGLUE(., NS, 3600, [ns1.example.com.])  
           IN DS $DSGLUE(ns1., A, 600, [192.0.2.1])  
           IN DS $DSGLUE(ns1., AAAA, 600, [2001:db8::1])  
           IN DS $DSGLUE(_dns.ns1., SVCB, 3600,  
                        [1 ns1.example.com. alpn=dot])
```

##### 4.4.1. Disabling DANE

Resolvers check whether a nameserver supports DANE by resolving a TLSA record during the delegation process (Section 6.4). However, this adds unnecessary latency to the delegation if the nameserver does not implement DANE. As an optimization, such nameservers can add an empty DSGLUE RRSset to indicate that there is no such TLSA record, e.g.:

```
IN DS $DSGLUE(_853._tcp.ns1., TLSA, 7200, [])
```

#### 5. Security Considerations

Resolvers that process DSGLUE MUST perform DNSSEC validation.

Source Records published as DSGLUE have owner names within the child zone, but are signed only by the parent. This makes them fully authenticated, but provides different cryptographic guarantees than a direct signature by the child. For example, these records might not appear in any key use logs maintained by the child.

#### 6. Operational Considerations

##### 6.1. Compatibility with existing resolvers

Resolver support for DSGLUE is OPTIONAL, so child zones MUST continue to place ordinary NS, A, and AAAA records in the parent zone as needed for non-DSGLUE resolution.



## 6.2. Publishing DSGLUE records

In order for the child to publish DSGLUE records, the parent must allow the child to publish arbitrary DS records or have specific support for this specification.

If the parent supports CDS [RFC8078], child zones MAY use CDS to push DSGLUE records into the parent. Note that CDNSKEY records cannot be used, because (1) the child cannot publish CDNSKEY records with the required owner name and (2) the child cannot guarantee that the parent will use the VERBATIM digest to produce the DS record.

Child zones SHOULD publish all Source Records as ordinary records of the specified type at the indicated owner name, in order to enable revalidation [I-D.draft-ietf-dnsop-ns-revalidation] and simplify debugging.

## 6.3. Referral response size

When records are present in both ordinary glue and DSGLUE, the response size is approximately doubled. This could cause performance issues due to response truncation when the initial query is over UDP.

## 6.4. PKI and DANE for Authenticated Encryption

TODO: Move some of this text into a different draft.

Nameservers supporting authenticated encryption MAY indicate any DANE mode, or none at all.

As an optimization, nameservers using DANE MAY place a TLSA record in the DSGLUE to avoid the latency of a TLSA lookup during delegation. However, child zones should be aware that this adds complexity and delay to the process of TLSA key rotation.

QUESTION: Should we recommend for or against including nonempty TLSA in DSGLUE? If CDS-like update mechanisms work well, and ADoT-DANE is widely deployed, this could warrant a positive recommendation. Conversely, if rotation is error-prone, and ADoT-DANE is rare, a negative recommendation might be better.

Nameservers that support PKI-based authentication but not DANE SHOULD deny the TLSA RSet in the DSGLUE, as shown in Section 4.4.1, to avoid an unnecessary delay.

Resolvers that support authenticated encryption MAY implement support for PKI-based authentication, DANE, or both. PKI-only resolvers MUST nonetheless resolve TLSA records, and MUST NOT require authentication

if the DANE mode is DANE-TA(2) or DANE-EE(3) [RFC7671]. DANE-only resolvers MUST NOT require authentication if the TLSA record does not exist.

## 7. IANA Considerations

IANA is requested to add a new entry to the DNS Security Algorithm Numbers registry:

Number	Description	Mnemonic	Zone Signing	Trans. Sec.	Reference
\$DSGLUE_NUM	Authenticated Glue	DSGLUE	N	?	(This document)

Table 1

IANA is requested to open a new registry named "Authenticated Glue Allowed Record Types", with a policy of "Standards Action" and the following fields:

- \* Record Type: The name of a registered DNS record type
- \* Interpretation Reference: The standards document defining how to interpret this RR type in the Authenticated Glue context.

The initial contents are as follows:

Record Type	Interpretation Reference
NS	(This document)
A	(This document)
AAAA	(This document)
SVCB	(This document)
TLSA	(This document)

Table 2

## 8. References

## 8.1. Normative References

- [I-D.draft-vandijk-dnsop-ds-digest-verbatim]  
Dijk, P. V., "The VERBATIM Digest Algorithm for DS records", Work in Progress, Internet-Draft, draft-vandijk-dnsop-ds-digest-verbatim-01, 10 August 2021, <<https://datatracker.ietf.org/doc/html/draft-vandijk-dnsop-ds-digest-verbatim-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>>.
- [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>>.
- [RFC7671] Dukhovni, V. and W. Hardaker, "The DNS-Based Authentication of Named Entities (DANE) Protocol: Updates and Operational Guidance", RFC 7671, DOI 10.17487/RFC7671, October 2015, <<https://www.rfc-editor.org/rfc/rfc7671>>.
- [RFC8078] Gudmundsson, O. and P. Wouters, "Managing DS Records from the Parent via CDS/CDNSKEY", RFC 8078, DOI 10.17487/RFC8078, March 2017, <<https://www.rfc-editor.org/rfc/rfc8078>>.
- [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>>.

## 8.2. Informative References

- [I-D.draft-ietf-dnsop-ns-revalidation]  
Huque, S., Vixie, P., and R. Dolmans, "Delegation Revalidation by DNS Resolvers", Work in Progress, Internet-Draft, draft-ietf-dnsop-ns-revalidation-01, 12 July 2021, <<https://datatracker.ietf.org/doc/html/draft-ietf-dnsop-ns-revalidation-01>>.
- [I-D.draft-schwartz-svcb-dns]  
Schwartz, B., "Service Binding Mapping for DNS Servers", Work in Progress, Internet-Draft, draft-schwartz-svcb-dns-04, 26 July 2021, <<https://datatracker.ietf.org/doc/html/draft-schwartz-svcb-dns-04>>.

[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>>.

#### Acknowledgments

Thanks to Paul Hoffman, Ilari Liusvaara, Puneet Sood, and Alexandar Mayrhofer for detailed comments.

#### Author's Address

Benjamin Schwartz  
Google LLC

Email: [bemasc@google.com](mailto:bemasc@google.com)