

Network Working Group  
Internet-Draft  
Intended status: Standards Track  
Expires: 22 October 2022

C. Huitema  
Private Octopus Inc.  
S. Dickinson  
Sinodun IT  
A. Mankin  
Salesforce  
20 April 2022

DNS over Dedicated QUIC Connections  
draft-ietf-dprive-dnsquic-12

Abstract

This document describes the use of QUIC to provide transport confidentiality for DNS. The encryption provided by QUIC has similar properties to those provided by TLS, while QUIC transport eliminates the head-of-line blocking issues inherent with TCP and provides more efficient packet loss recovery than UDP. DNS over QUIC (DoQ) has privacy properties similar to DNS over TLS (DoT) specified in RFC7858, and latency characteristics similar to classic DNS over UDP. This specification describes the use of DNS over QUIC as a general-purpose transport for DNS and includes the use of DNS over QUIC for stub to recursive, recursive to authoritative, and zone transfer scenarios.

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 22 October 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
2. Key Words . . . . .	4
3. Document work via GitHub . . . . .	5
4. Design Considerations . . . . .	5
4.1. Provide DNS Privacy . . . . .	5
4.2. Design for Minimum Latency . . . . .	5
4.3. Middlebox Considerations . . . . .	6
4.4. No Server-Initiated Transactions . . . . .	6
5. Specifications . . . . .	6
5.1. Connection Establishment . . . . .	7
5.1.1. Draft Version Identification . . . . .	7
5.1.2. Port Selection . . . . .	7
5.2. Stream Mapping and Usage . . . . .	7
5.2.1. DNS Message IDs . . . . .	9
5.3. DoQ Error Codes . . . . .	9
5.3.1. Transaction Cancellation . . . . .	10
5.3.2. Transaction Errors . . . . .	11
5.3.3. Protocol Errors . . . . .	11
5.3.4. Alternative error codes . . . . .	12
5.4. Connection Management . . . . .	12
5.5. Session Resumption and 0-RTT . . . . .	13
5.6. Message Sizes . . . . .	14
6. Implementation Requirements . . . . .	14
6.1. Authentication . . . . .	14
6.2. Fallback to Other Protocols on Connection Failure . . . . .	15
6.3. Address Validation . . . . .	15
6.4. Padding . . . . .	16
6.5. Connection Handling . . . . .	16
6.5.1. Connection Reuse . . . . .	17
6.5.2. Resource Management . . . . .	17
6.5.3. Using 0-RTT and Session Resumption . . . . .	18
6.5.4. Controlling Connection Migration For Privacy . . . . .	18
6.6. Processing Queries in Parallel . . . . .	19
6.7. Zone transfer . . . . .	19
6.8. Flow Control Mechanisms . . . . .	19
7. Implementation Status . . . . .	20
7.1. Performance Measurements . . . . .	21

8. Security Considerations . . . . .	21
9. Privacy Considerations . . . . .	22
9.1. Privacy Issues With 0-RTT data . . . . .	22
9.2. Privacy Issues With Session Resumption . . . . .	23
9.3. Privacy Issues With Address Validation Tokens . . . . .	24
9.4. Privacy Issues With Long Duration Sessions . . . . .	24
9.5. Traffic Analysis . . . . .	25
10. IANA Considerations . . . . .	25
10.1. Registration of DoQ Identification String . . . . .	25
10.2. Reservation of Dedicated Port . . . . .	25
10.3. Reservation of Extended DNS Error Code Too Early . . . . .	26
10.4. DNS over QUIC Error Codes Registry . . . . .	27
11. Acknowledgements . . . . .	28
12. References . . . . .	29
12.1. Normative References . . . . .	29
12.2. Informative References . . . . .	31
Appendix A. The NOTIFY Service . . . . .	33
Appendix B. Notable Changes From Previous Versions . . . . .	33
B.1. Stream Mapping Incompatibility With Draft-02 . . . . .	33
Authors' Addresses . . . . .	34

## 1. Introduction

Domain Name System (DNS) concepts are specified in "Domain names - concepts and facilities" [RFC1034]. The transmission of DNS queries and responses over UDP and TCP is specified in "Domain names - implementation and specification" [RFC1035].

This document presents a mapping of the DNS protocol over the QUIC transport [RFC9000] [RFC9001]. DNS over QUIC is referred to here as DoQ, in line with "DNS Terminology" [I-D.ietf-dnsop-rfc8499bis].

The goals of the DoQ mapping are:

1. Provide the same DNS privacy protection as DNS over TLS (DoT) [RFC7858]. This includes an option for the client to authenticate the server by means of an authentication domain name as specified in "Usage Profiles for DNS over TLS and DNS over DTLS" [RFC8310].
2. Provide an improved level of source address validation for DNS servers compared to classic DNS over UDP.
3. Provide a transport that does not impose path MTU limitations on the size of DNS responses it can send.

In order to achieve these goals, and to support ongoing work on encryption of DNS, the scope of this document includes

- \* the "stub to recursive resolver" scenario
- \* the "recursive resolver to authoritative nameserver" scenario and
- \* the "nameserver to nameserver" scenario (mainly used for zone transfers (XFR) [RFC1995], [RFC5936]).

In other words, this document specifies QUIC as a general-purpose transport for DNS.

The specific non-goals of this document are:

1. No attempt is made to evade potential blocking of DNS over QUIC traffic by middleboxes.
2. No attempt to support server-initiated transactions, which are used only in DNS Stateful Operations (DSO) [RFC8490].

Specifying the transmission of an application over QUIC requires specifying how the application's messages are mapped to QUIC streams, and generally how the application will use QUIC. This is done for HTTP in "Hypertext Transfer Protocol Version 3 (HTTP/3)" [I-D.ietf-quic-http]. The purpose of this document is to define the way DNS messages can be transmitted over QUIC.

DNS over HTTP [RFC8484] can be used with HTTP/3 to get some of the benefits of QUIC. However, a lightweight direct mapping for DNS over QUIC can be regarded as a more natural fit for both the recursive to authoritative and zone transfer scenarios which rarely involve intermediaries. In these scenarios, the additional overhead of HTTP is not offset by, e.g., benefits of HTTP proxying and caching behavior.

In this document, Section 4 presents the reasoning that guided the proposed design. Section 5 specifies the actual mapping of DoQ. Section 6 presents guidelines on the implementation, usage and deployment of DoQ.

## 2. Key Words

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. Document work via GitHub

(RFC EDITOR NOTE: THIS SECTION TO BE REMOVED BEFORE PUBLICATION) The GitHub repository for this document is at <https://github.com/huitema/dnssoquic>. Proposed text and editorial changes are very much welcomed there, but any functional changes should always first be discussed on the IETF DPRIVE WG (dns-privacy) mailing list.

### 4. Design Considerations

This section and its subsections present the design guidelines that were used for DoQ. Whilst all other sections in this document are normative, this section is informative in nature.

#### 4.1. Provide DNS Privacy

DoT [RFC7858] defines how to mitigate some of the issues described in "DNS Privacy Considerations" [RFC9076] by specifying how to transmit DNS messages over TLS. The "Usage Profiles for DNS over TLS and DNS over DTLS" [RFC8310] specify Strict and Opportunistic Usage Profiles for DoT including how stub resolvers can authenticate recursive resolvers.

QUIC connection setup includes the negotiation of security parameters using TLS, as specified in "Using TLS to Secure QUIC" [RFC9001], enabling encryption of the QUIC transport. Transmitting DNS messages over QUIC will provide essentially the same privacy protections as DoT [RFC7858] including Strict and Opportunistic Usage Profiles [RFC8310]. Further discussion on this is provided in Section 9.

#### 4.2. Design for Minimum Latency

QUIC is specifically designed to reduce protocol-induced delays, with features such as:

1. Support for 0-RTT data during session resumption.
2. Support for advanced packet loss recovery procedures as specified in "QUIC Loss Detection and Congestion Control" [RFC9002].
3. Mitigation of head-of-line blocking by allowing parallel delivery of data on multiple streams.

This mapping of DNS to QUIC will take advantage of these features in three ways:

1. Optional support for sending 0-RTT data during session resumption (the security and privacy implications of this are discussed in later sections).
2. Long-lived QUIC connections over which multiple DNS transactions are performed, generating the sustained traffic required to benefit from advanced recovery features.
3. Mapping of each DNS Query/Response transaction to a separate stream, to mitigate head-of-line blocking. This enables servers to respond to queries "out of order". It also enables clients to process responses as soon as they arrive, without having to wait for in order delivery of responses previously posted by the server.

These considerations are reflected in the mapping of DNS traffic to QUIC streams in Section 5.2.

#### 4.3. Middlebox Considerations

Using QUIC might allow a protocol to disguise its purpose from devices on the network path using encryption and traffic analysis resistance techniques like padding, traffic pacing, and traffic shaping. This specification does not include any measures that are designed to avoid such classification -- the padding mechanisms defined in Section 6.4 are intended to obfuscate the specific records contained in DNS queries and responses, but not the fact that this is DNS traffic. Consequently, firewalls and other middleboxes might be able to distinguish DoQ from other protocols that use QUIC, like HTTP, and apply different treatment.

The lack of measures in this specification to avoid protocol classification is not an endorsement of such practices.

#### 4.4. No Server-Initiated Transactions

As stated in Section 1, this document does not specify support for server-initiated transactions within established DoQ connections. That is, only the initiator of the DoQ connection may send queries over the connection.

DSO does support server-initiated transactions within existing connections. However, DoQ as defined here does not meet the criteria for an applicable transport for DSO because it does not guarantee in-order delivery of messages, see Section 4.2 of [RFC8490].

### 5. Specifications

### 5.1. Connection Establishment

DoQ connections are established as described in the QUIC transport specification [RFC9000]. During connection establishment, DoQ support is indicated by selecting the Application-Layer Protocol Negotiation (ALPN) token "doq" in the crypto handshake.

#### 5.1.1. Draft Version Identification

(RFC EDITOR NOTE: THIS SECTION TO BE REMOVED BEFORE PUBLICATION) Only implementations of the final, published RFC can identify themselves as "doq". Until such an RFC exists, implementations MUST NOT identify themselves using this string.

Implementations of draft versions of the protocol MUST add the string "-" and the corresponding draft number to the identifier. For example, draft-ietf-dprive-dnsquic-00 is identified using the string "doq-i00".

#### 5.1.2. Port Selection

By default, a DNS server that supports DoQ MUST listen for and accept QUIC connections on the dedicated UDP port TBD (number to be defined in Section 10), unless there is a mutual agreement to use another port.

By default, a DNS client desiring to use DoQ with a particular server MUST establish a QUIC connection to UDP port TBD on the server, unless there is a mutual agreement to use another port.

DoQ connections MUST NOT use UDP port 53. This recommendation against use of port 53 for DoQ is to avoid confusion between DoQ and the use of DNS over UDP [RFC1035]. The risk of confusion exists even if two parties agreed on port 53, as other parties without knowledge of that agreement might still try to use that port.

In the stub to recursive scenario, the use of port 443 as a mutually agreed alternative port can be operationally beneficial, since port 443 is used by many services using QUIC and HTTP-3 and thus less likely to be blocked than other ports. Several mechanisms for stubs to discover recursives offering encrypted transports, including the use of custom ports, are the subject of ongoing work.

### 5.2. Stream Mapping and Usage

The mapping of DNS traffic over QUIC streams takes advantage of the QUIC stream features detailed in Section 2 of [RFC9000], the QUIC transport specification.

DNS query/response traffic [RFC1034], [RFC1035] follows a simple pattern in which the client sends a query, and the server provides one or more responses (multiple responses can occur in zone transfers).

The mapping specified here requires that the client selects a separate QUIC stream for each query. The server then uses the same stream to provide all the response messages for that query. In order that multiple responses can be parsed, a 2-octet length field is used in exactly the same way as the 2-octet length field defined for DNS over TCP [RFC1035]. The practical result of this is that the content of each QUIC stream is exactly the same as the content of a TCP connection that would manage exactly one query.

All DNS messages (queries and responses) sent over DoQ connections MUST be encoded as a 2-octet length field followed by the message content as specified in [RFC1035].

The client MUST select the next available client-initiated bidirectional stream for each subsequent query on a QUIC connection, in conformance with the QUIC transport specification [RFC9000]. Packet losses and other network events might cause queries to arrive in a different order. Servers SHOULD process queries as they arrive, as not doing so would cause unnecessary delays.

The client MUST send the DNS query over the selected stream, and MUST indicate through the STREAM FIN mechanism that no further data will be sent on that stream.

The server MUST send the response(s) on the same stream and MUST indicate, after the last response, through the STREAM FIN mechanism that no further data will be sent on that stream.

Therefore, a single DNS transaction consumes a single bidirectional client-initiated stream. This means that the client's first query occurs on QUIC stream 0, the second on 4, and so on (see Section 2.1 of [RFC9000]).

Servers MAY defer processing of a query until the STREAM FIN has been indicated on the stream selected by the client.

Servers and clients MAY monitor the number of "dangling" streams. These are open streams where the following events have not occurred after implementation defined timeouts:

- \* the expected queries or responses have not been received or,



- \* the expected queries or responses have been received but not the STREAM FIN

Implementations MAY impose a limit on the number of such dangling streams. If limits are encountered, implementations MAY close the connection.

#### 5.2.1. DNS Message IDs

When sending queries over a QUIC connection, the DNS Message ID MUST be set to zero. The stream mapping for DoQ allows for unambiguous correlation of queries and responses and so the Message ID field is not required.

This has implications for proxying DoQ message to and from other transports. For example, proxies may have to manage the fact that DoQ can support a larger number of outstanding queries on a single connection than e.g., DNS over TCP because DoQ is not limited by the Message ID space. This issue already exists for DoH, where a Message ID of 0 is recommended.

When forwarding a DNS message from DoQ over another transport, a DNS Message ID MUST be generated according to the rules of the protocol that is in use. When forwarding a DNS message from another transport over DoQ, the Message ID MUST be set to zero.

#### 5.3. DoQ Error Codes

The following error codes are defined for use when abruptly terminating streams, and used as application protocol error codes when aborting reading of streams, or immediately closing connections:

DOQ\_NO\_ERROR (0x0): No error. This is used when the connection or stream needs to be closed, but there is no error to signal.

DOQ\_INTERNAL\_ERROR (0x1): The DoQ implementation encountered an internal error and is incapable of pursuing the transaction or the connection.

DOQ\_PROTOCOL\_ERROR (0x2): The DoQ implementation encountered a protocol error and is forcibly aborting the connection.

DOQ\_REQUEST\_CANCELLED (0x3): A DoQ client uses this to signal that it wants to cancel an outstanding transaction.

DOQ\_EXCESSIVE\_LOAD (0x4): A DoQ implementation uses this to signal when closing a connection due to excessive load.

DOQ\_UNSPECIFIED\_ERROR (0x5): A DoQ implementation uses this in the absence of a more specific error code.

DOQ\_ERROR\_RESERVED (0xd098ea5e): Alternative error code used for tests.

See Section 10.4 for details on registering new error codes.

#### 5.3.1. Transaction Cancellation

In QUIC, sending STOP\_SENDING requests that a peer cease transmission on a stream. If a DoQ client wishes to cancel an outstanding request, it MUST issue a QUIC STOP\_SENDING, and it SHOULD use the error code DOQ\_REQUEST\_CANCELLED. It MAY use a more specific error code registered according to Section 10.4. The STOP\_SENDING request may be sent at any time but will have no effect if the server response has already been sent, in which case the client will simply discard the incoming response. The corresponding DNS transaction MUST be abandoned.

Servers that receive STOP\_SENDING act in accordance with Section 3.5 of [RFC9000]. Servers SHOULD NOT continue processing a DNS transaction if they receive a STOP\_SENDING.

Servers MAY impose implementation limits on the total number or rate of request cancellations. If limits are encountered, servers MAY close the connection. In this case, servers wanting to help client debugging MAY use the error code DOQ\_EXCESSIVE\_LOAD. There is always a trade-off between helping good faith clients debug issues and allowing denial-of-service attackers to test server defenses, so depending on circumstances servers might very well choose to send different error codes.

Note that this mechanism provides a way for secondaries to cancel a single zone transfer occurring on a given stream without having to close the QUIC connection.

Servers MUST NOT continue processing a DNS transaction if they receive a RESET\_STREAM request from the client before the client indicates the STREAM FIN. The server MUST issue a RESET\_STREAM to indicate that the transaction is abandoned unless

- \* it has already done so for another reason or
- \* it has already both sent the response and indicated the STREAM FIN.

### 5.3.2. Transaction Errors

Servers normally complete transactions by sending a DNS response (or responses) on the transaction's stream, including cases where the DNS response indicates a DNS error. For example, a Server Failure (SERVFAIL, [RFC1035]) SHOULD be notified to the client by sending back a response with the Response Code set to SERVFAIL.

If a server is incapable of sending a DNS response due to an internal error, it SHOULD issue a QUIC RESET\_STREAM frame. The error code SHOULD be set to DOQ\_INTERNAL\_ERROR. The corresponding DNS transaction MUST be abandoned. Clients MAY limit the number of unsolicited QUIC Stream Resets received on a connection before choosing to close the connection.

Note that this mechanism provides a way for primaries to abort a single zone transfer occurring on a given stream without having to close the QUIC connection.

### 5.3.3. Protocol Errors

Other error scenarios can occur due to malformed, incomplete or unexpected messages during a transaction. These include (but are not limited to)

- \* a client or server receives a message with a non-zero Message ID
- \* a client or server receives a STREAM FIN before receiving all the bytes for a message indicated in the 2-octet length field
- \* a client receives a STREAM FIN before receiving all the expected responses
- \* a server receives more than one query on a stream
- \* a client receives a different number of responses on a stream than expected (e.g., multiple responses to a query for an A record)
- \* a client receives a STOP\_SENDING request
- \* the client or server does not indicate the expected STREAM FIN after sending requests or responses (see Section 5.2).
- \* an implementation receives a message containing the edns-tcp-keepalive EDNS(0) Option [RFC7828] (see Section 6.5.2)
- \* a client or a server attempts to open a unidirectional QUIC stream

- \* a server attempts to open a server-initiated bidirectional QUIC stream
- \* receiving a "replayable" transaction in 0-RTT data (for servers not willing to handle this case - see section Section 5.5)

If a peer encounters such an error condition it is considered a fatal error. It SHOULD forcibly abort the connection using QUIC's CONNECTION\_CLOSE mechanism, and SHOULD use the DoQ error code DOQ\_PROTOCOL\_ERROR. In some cases, it MAY instead silently abandon the connection, which uses fewer of the local resources but makes debugging at the offending node more difficult.

It is noted that the restrictions on use of the above EDNS(0) options has implications for proxying message from TCP/DoT/DoH over DoQ.

#### 5.3.4. Alternative error codes

This specification suggests specific error codes in Section 5.3.1, Section 5.3.2, and Section 5.3.3. These error codes are meant to facilitate investigation of failures and other incidents. New error codes may be defined in future versions of DoQ, or registered as specified in Section 10.4.

Because new error codes can be defined without negotiation, use of an error code in an unexpected context or receipt of an unknown error code MUST be treated as equivalent to DOQ\_UNSPECIFIED\_ERROR.

Implementations MAY wish to test the support for the error code extension mechanism by using error codes not listed in this document, or they MAY use DOQ\_ERROR\_RESERVED.

#### 5.4. Connection Management

Section 10 of [RFC9000], the QUIC transport specification, specifies that connections can be closed in three ways:

- \* idle timeout
- \* immediate close
- \* stateless reset

Clients and servers implementing DoQ SHOULD negotiate use of the idle timeout. Closing on idle timeout is done without any packet exchange, which minimizes protocol overhead. Per Section 10.1 of [RFC9000], the QUIC transport specification, the effective value of the idle timeout is computed as the minimum of the values advertised by the two endpoints. Practical considerations on setting the idle timeout are discussed in Section 6.5.2.

Clients SHOULD monitor the idle time incurred on their connection to the server, defined by the time spent since the last packet from the server has been received. When a client prepares to send a new DNS query to the server, it SHOULD check whether the idle time is sufficiently lower than the idle timer. If it is, the client SHOULD send the DNS query over the existing connection. If not, the client SHOULD establish a new connection and send the query over that connection.

Clients MAY discard their connections to the server before the idle timeout expires. A client that has outstanding queries SHOULD close the connection explicitly using QUIC's CONNECTION\_CLOSE mechanism and the DoQ error code DOQ\_NO\_ERROR.

Clients and servers MAY close the connection for a variety of other reasons, indicated using QUIC's CONNECTION\_CLOSE. Client and servers that send packets over a connection discarded by their peer might receive a stateless reset indication. If a connection fails, all the in progress transaction on that connection MUST be abandoned.

#### 5.5. Session Resumption and 0-RTT

A client MAY take advantage of the session resumption and 0-RTT mechanisms supported by QUIC transport [RFC9000] and QUIC TLS [RFC9001], if the server supports them. Clients SHOULD consider potential privacy issues associated with session resumption before deciding to use this mechanism and specifically evaluate the trade-offs presented in the various sections of this document. The privacy issues are detailed in Section 9.1 and Section 9.2, and the implementation considerations are discussed in Section 6.5.3.

The 0-RTT mechanism MUST NOT be used to send DNS requests that are not "replayable" transactions. In this specification, only transactions that have an OPCODE of QUERY or NOTIFY are considered replayable and therefore other OPCODES MUST NOT be sent in 0-RTT data. See Appendix A for a detailed discussion of why NOTIFY is included here.

Servers MAY support session resumption, and MAY do that with or without supporting 0-RTT, using the mechanisms described in Section 4.6.1 of [RFC9001]. Servers supporting 0-RTT MUST NOT immediately process non-replayable transactions received in 0-RTT data, but instead MUST adopt one of the following behaviours:

- \* Queue the offending transaction and only process it after the QUIC handshake has been completed, as defined in Section 4.1.1 of [RFC9001].
- \* Reply to the offending transaction with a response code REFUSED and an Extended DNS Error Code (EDE) "Too Early", using the extended RCODE mechanisms defined in [RFC6891] and the extended DNS errors defined in [RFC8914]; see Section 10.3.
- \* Close the connection with the error code DOQ\_PROTOCOL\_ERROR.

## 5.6. Message Sizes

DoQ Queries and Responses are sent on QUIC streams, which in theory can carry up to  $2^{62}$  bytes. However, DNS messages are restricted in practice to a maximum size of 65535 bytes. This maximum size is enforced by the use of a two-octet message length field in DNS over TCP [RFC1035] and DNS over TLS [RFC7858], and by the definition of the "application/dns-message" for DNS over HTTP [RFC8484]. DoQ enforces the same restriction.

The Extension Mechanisms for DNS (EDNS) [RFC6891] allow peers to specify the UDP message size. This parameter is ignored by DoQ. DoQ implementations always assume that the maximum message size is 65535 bytes.

## 6. Implementation Requirements

### 6.1. Authentication

For the stub to recursive resolver scenario, the authentication requirements are the same as described in DoT [RFC7858] and "Usage Profiles for DNS over TLS and DNS over DTLS" [RFC8310]. [RFC8932] states that DNS privacy services SHOULD provide credentials that clients can use to authenticate the server. Given this, and to align with the authentication model for DoH, DoQ stubs SHOULD use a Strict authentication profile. Client authentication for the encrypted stub to recursive scenario is not described in any DNS RFC.

For zone transfer, the authentication requirements are the same as described in [RFC9103].

For the recursive resolver to authoritative nameserver scenario, authentication requirements are unspecified at the time of writing and are the subject on ongoing work in the DPRIVE WG.

## 6.2. Fallback to Other Protocols on Connection Failure

If the establishment of the DoQ connection fails, clients MAY attempt to fall back to DoT and then potentially clear text, as specified in DoT [RFC7858] and "Usage Profiles for DNS over TLS and DNS over DTLS" [RFC8310], depending on their privacy profile.

DNS clients SHOULD remember server IP addresses that don't support DoQ. Mobile clients might also remember the lack of DoQ support by given IP addresses on a per-context basis (e.g. per network or provisioning domain).

Timeouts, connection refusals, and QUIC handshake failures are indicators that a server does not support DoQ. Clients SHOULD NOT attempt DoQ queries to a server that does not support DoQ for a reasonable period (such as one hour per server). DNS clients following an out-of-band key-pinned privacy profile ([RFC7858]) MAY be more aggressive about retrying after DoQ connection failures.

## 6.3. Address Validation

Section 8 of [RFC9000], the QUIC transport specification, defines Address Validation procedures to avoid servers being used in address amplification attacks. DoQ implementations MUST conform to this specification, which limits the worst case amplification to a factor 3.

DoQ implementations SHOULD consider configuring servers to use the Address Validation using Retry Packets procedure defined in Section 8.1.2 of [RFC9000], the QUIC transport specification. This procedure imposes a 1-RTT delay for verifying the return routability of the source address of a client, similar to the DNS Cookies mechanism [RFC7873].

DoQ implementations that configure Address Validation using Retry Packets SHOULD implement the Address Validation for Future Connections procedure defined in Section 8.1.3 of [RFC9000], the QUIC transport specification. This defines how servers can send NEW\_TOKEN frames to clients after the client address is validated, in order to avoid the 1-RTT penalty during subsequent connections by the client from the same address.

#### 6.4. Padding

Implementations MUST protect against the traffic analysis attacks described in Section 9.5 by the judicious injection of padding. This could be done either by padding individual DNS messages using the EDNS(0) Padding Option [RFC7830] or by padding QUIC packets (see Section 19.1 of [RFC9000]).

In theory, padding at the QUIC packet level could result in better performance for the equivalent protection, because the amount of padding can take into account non-DNS frames such as acknowledgements or flow control updates, and also because QUIC packets can carry multiple DNS messages. However, applications can only control the amount of padding in QUIC packets if the implementation of QUIC exposes adequate APIs. This leads to the following recommendation:

- \* if the implementation of QUIC exposes APIs to set a padding policy, DNS over QUIC SHOULD use that API to align the packet length to a small set of fixed sizes.
- \* if padding at the QUIC packet level is not available or not used, DNS over QUIC MUST ensure that all DNS queries and responses are padded to a small set of fixed sizes, using the EDNS(0) padding extension as specified in [RFC7830].

Implementation might choose not to use a QUIC API for padding if it is significantly simpler to re-use existing DNS message padding logic which is applied to other encrypted transports.

In the absence of a standard policy for padding sizes, implementations SHOULD follow the recommendations of the Experimental status "Padding Policies for Extension Mechanisms for DNS (EDNS(0))" [RFC8467]. While Experimental, these recommendations are referenced because they are implemented and deployed for DoT, and provide a way for implementations to be fully compliant with this specification.

#### 6.5. Connection Handling

"DNS Transport over TCP - Implementation Requirements" [RFC7766] provides updated guidance on DNS over TCP, some of which is applicable to DoQ. This section provides similar advice on connection handling for DoQ.



#### 6.5.1. Connection Reuse

Historic implementations of DNS clients are known to open and close TCP connections for each DNS query. To amortize connection setup costs, both clients and servers SHOULD support connection reuse by sending multiple queries and responses over a single persistent QUIC connection.

In order to achieve performance on par with UDP, DNS clients SHOULD send their queries concurrently over the QUIC streams on a QUIC connection. That is, when a DNS client sends multiple queries to a server over a QUIC connection, it SHOULD NOT wait for an outstanding reply before sending the next query.

#### 6.5.2. Resource Management

Proper management of established and idle connections is important to the healthy operation of a DNS server.

An implementation of DoQ SHOULD follow best practices similar to those specified for DNS over TCP [RFC7766], in particular with regard to:

- \* Concurrent Connections (Section 6.2.2 of [RFC7766], updated by Section 6.4 of [RFC9103])
- \* Security Considerations (Section 10 of [RFC7766])

Failure to do so may lead to resource exhaustion and denial of service.

Clients that want to maintain long duration DoQ connections SHOULD use the idle timeout mechanisms defined in Section 10.1 of [RFC9000], the QUIC transport specification. Clients and servers MUST NOT send the edns-tcp-keepalive EDNS(0) Option [RFC7828] in any messages sent on a DoQ connection (because it is specific to the use of TCP/TLS as a transport).

This document does not make specific recommendations for timeout values on idle connections. Clients and servers should reuse and/or close connections depending on the level of available resources. Timeouts may be longer during periods of low activity and shorter during periods of high activity.

### 6.5.3. Using 0-RTT and Session Resumption

Using 0-RTT for DNS over QUIC has many compelling advantages. Clients can establish connections and send queries without incurring a connection delay. Servers can thus negotiate low values of the connection timers, which reduces the total number of connections that they need to manage. They can do that because the clients that use 0-RTT will not incur latency penalties if new connections are required for a query.

Session resumption and 0-RTT data transmission create privacy risks detailed in Section 9.2 and Section 9.1. The following recommendations are meant to reduce the privacy risks while enjoying the performance benefits of 0-RTT data, subject to the restrictions specified in Section 5.5.

Clients SHOULD use resumption tickets only once, as specified in Appendix C.4 to [RFC8446]. By default, clients SHOULD NOT use session resumption if the client's connectivity has changed.

Clients could receive address validation tokens from the server using the NEW\_TOKEN mechanism; see Section 8 of [RFC9000]. The associated tracking risks are mentioned in Section 9.3. Clients SHOULD only use the address validation tokens when they are also using session resumption, thus avoiding additional tracking risks.

Servers SHOULD issue session resumption tickets with a sufficiently long lifetime (e.g., 6 hours), so that clients are not tempted to either keep connection alive or frequently poll the server to renew session resumption tickets. Servers SHOULD implement the anti-replay mechanisms specified in Section 8 of [RFC8446].

### 6.5.4. Controlling Connection Migration For Privacy

DoQ implementations might consider using the connection migration features defined in Section 9 of [RFC9000]. These features enable connections to continue operating as the client's connectivity changes. As detailed in Section 9.4, these features trade off privacy for latency. By default, clients SHOULD be configured to prioritize privacy and start new sessions if their connectivity changes.

## 6.6. Processing Queries in Parallel

As specified in Section 7 of [RFC7766] "DNS Transport over TCP - Implementation Requirements", resolvers are RECOMMENDED to support the preparing of responses in parallel and sending them out of order. In DoQ, they do that by sending responses on their specific stream as soon as possible, without waiting for availability of responses for previously opened streams.

## 6.7. Zone transfer

[RFC9103] specifies zone transfer over TLS (XoT) and includes updates to [RFC1995] (IXFR), [RFC5936] (AXFR) and [RFC7766]. Considerations relating to the re-use of XoT connections described there apply analogously to zone transfers performed using DoQ connections. One reason for re-iterating such specific guidance is the lack of effective connection re-use in existing TCP/TLS zone transfer implementations today. The following recommendations apply:

- \* DoQ servers MUST be able to handle multiple concurrent IXFR requests on a single QUIC connection
- \* DoQ servers MUST be able to handle multiple concurrent AXFR requests on a single QUIC connection
- \* DoQ implementations SHOULD
  - use the same QUIC connection for both AXFR and IXFR requests to the same primary
  - send those requests in parallel as soon as they are queued i.e. do not wait for a response before sending the next query on the connection (this is analogous to pipelining requests on a TCP/TLS connection)
  - send the response(s) for each request as soon as they are available i.e. response streams MAY be sent intermingled

## 6.8. Flow Control Mechanisms

Servers and Clients manage flow control using the mechanisms defined in Section 4 of [RFC9000]. These mechanisms allow clients and servers to specify how many streams can be created, how much data can be sent on a stream, and how much data can be sent on the union of all streams. For DNS over QUIC, controlling how many streams are created allows servers to control how many new requests the client can send on a given connection.

Flow control exists to protect endpoint resources. For servers, global and per-stream flow control limits control how much data can be sent by clients. The same mechanisms allow clients to control how much data can be sent by servers. Values that are too small will unnecessarily limit performance. Values that are too large might expose endpoints to overload or memory exhaustion. Implementations or deployments will need to adjust flow control limits to balance these concerns. In particular, zone transfer implementations will need to control these limits carefully to ensure both large and concurrent zone transfers are well managed.

Initial values of parameters control how many requests and how much data can be sent by clients and servers at the beginning of the connection. These values are specified in transport parameters exchanged during the connection handshake. The parameter values received in the initial connection also control how many requests and how much data can be sent by clients using 0-RTT data in a resumed connection. Using too small values of these initial parameters would restrict the usefulness of allowing 0-RTT data.

## 7. Implementation Status

(RFC EDITOR NOTE: THIS SECTION TO BE REMOVED BEFORE PUBLICATION) This section records the status of known implementations of the protocol defined by this specification at the time of posting of this Internet-Draft, and is based on a proposal described in [RFC7942].

1. AdGuard launched a DoQ recursive resolver service in December 2020. They have released a suite of open source tools that support DoQ:
  1. AdGuard C++ DNS libraries (<https://github.com/AdguardTeam/DnsLibs>) A DNS proxy library that supports all existing DNS protocols including DNS-over-TLS, DNS-over-HTTPS, DNSCrypt and DNS-over-QUIC (experimental).
  2. DNS Proxy (<https://github.com/AdguardTeam/dnsproxy>) A simple DNS proxy server that supports all existing DNS protocols including DNS-over-TLS, DNS-over-HTTPS, DNSCrypt, and DNS-over-QUIC. Moreover, it can work as a DNS-over-HTTPS, DNS-over-TLS or DNS-over-QUIC server.
  3. CoreDNS fork for AdGuard DNS (<https://github.com/AdguardTeam/coredns>) Includes DNS-over-QUIC server-side support.
  4. dnslookup (<https://github.com/ameshkov/dnslookup>) Simple command line utility to make DNS lookups. Supports all known DNS protocols: plain DNS, DoH, DoT, DoQ, DNSCrypt.

2. Quicdoq (<https://github.com/private-octopus/quicdoq>) Quicdoq is a simple open source implementation of DoQ. It is written in C, based on Picoquic (<https://github.com/private-octopus/picoquic>).
3. Flamethrower (<https://github.com/DNS-OARC/flamethrower/tree/dns-over-quic>) is an open source DNS performance and functional testing utility written in C++ that has an experimental implementation of DoQ.
4. aioquic (<https://github.com/aiortc/aioquic>) is an implementation of QUIC in Python. It includes example client and server for DoQ.

#### 7.1. Performance Measurements

To the authors' knowledge, no benchmarking studies comparing DoT, DoH and DoQ are published yet. However, anecdotal evidence from the AdGuard DoQ recursive resolver deployment (<https://adguard.com/en/blog/dns-over-quic.html>) indicates that it performs similarly (and possibly better) compared to the other encrypted protocols, particularly in mobile environments. Reasons given for this include that DoQ

- \* Uses less bandwidth due to a more efficient handshake (and due to less per message overhead when compared to DoH).
- \* Performs better in mobile environments due to the increased resilience to packet loss
- \* Can maintain connections as users move between mobile networks via its connection management

#### 8. Security Considerations

A Threat Analysis of the Domain Name System is found in [RFC3833]. This analysis was written before the development of DoT, DoH and DoQ, and probably needs to be updated.

The security considerations of DoQ should be comparable to those of DoT [RFC7858]. DoT as specified in [RFC7858] only addresses the stub to recursive resolver scenario, but the considerations about person-in-the-middle attacks, middleboxes and caching of data from clear text connections also apply for DoQ to the resolver to authoritative server scenario. As stated in Section 6.1 the authentication requirements for securing zone transfer using DoQ are the same as those for zone transfer over DoT, therefore the general security considerations are entirely analogous to those described in [RFC9103].

DoQ relies on QUIC, which itself relies on TLS 1.3 and thus supports by default the protections against downgrade attacks described in [BCP195]. QUIC specific issues and their mitigations are described in Section 21 of [RFC9000].

## 9. Privacy Considerations

The general considerations of encrypted transports provided in "DNS Privacy Considerations" [RFC9076] apply to DoQ. The specific considerations provided there do not differ between DoT and DoQ, and are not discussed further here. Similarly, "Recommendations for DNS Privacy Service Operators" [RFC8932] (which covers operational, policy, and security considerations for DNS privacy services) is also applicable to DoQ services.

QUIC incorporates the mechanisms of TLS 1.3 [RFC8446] and this enables QUIC transmission of "0-RTT" data. This can provide interesting latency gains, but it raises two concerns:

1. Adversaries could replay the 0-RTT data and infer its content from the behavior of the receiving server.
2. The 0-RTT mechanism relies on TLS session resumption, which can provide linkability between successive client sessions.

These issues are developed in Section 9.1 and Section 9.2.

### 9.1. Privacy Issues With 0-RTT data

The 0-RTT data can be replayed by adversaries. That data may trigger queries by a recursive resolver to authoritative resolvers. Adversaries may be able to pick a time at which the recursive resolver outgoing traffic is observable, and thus find out what name was queried for in the 0-RTT data.

This risk is in fact a subset of the general problem of observing the behavior of the recursive resolver discussed in "DNS Privacy Considerations" [RFC9076]. The attack is partially mitigated by reducing the observability of this traffic. The mandatory replay protection mechanisms in TLS 1.3 [RFC8446] limit but do not eliminate the risk of replay. 0-RTT packets can only be replayed within a narrow window, which is only wide enough to account for variations in clock skew and network transmission.

The recommendation for TLS 1.3 [RFC8446] is that the capability to use 0-RTT data should be turned off by default, and only enabled if the user clearly understands the associated risks. In the case of DoQ, allowing 0-RTT data provides significant performance gains, and

there is a concern that a recommendation to not use it would simply be ignored. Instead, a set of practical recommendations is provided in Section 5.5 and Section 6.5.3.

The specifications in Section 5.5 block the most obvious risks of replay attacks, as they only allow for transactions that will not change the long-term state of the server.

The attacks described above apply to the stub resolver to recursive resolver scenario, but similar attacks might be envisaged in the recursive resolver to authoritative resolver scenario, and the same mitigations apply.

## 9.2. Privacy Issues With Session Resumption

The QUIC session resumption mechanism reduces the cost of re-establishing sessions and enables 0-RTT data. There is a linkability issue associated with session resumption, if the same resumption token is used several times. Attackers on path between client and server could observe repeated usage of the token and use that to track the client over time or over multiple locations.

The session resumption mechanism allows servers to correlate the resumed sessions with the initial sessions, and thus to track the client. This creates a virtual long duration session. The series of queries in that session can be used by the server to identify the client. Servers can most probably do that already if the client address remains constant, but session resumption tickets also enable tracking after changes of the client's address.

The recommendations in Section 6.5.3 are designed to mitigate these risks. Using session tickets only once mitigates the risk of tracking by third parties. Refusing to resume a session if addresses change mitigates the incremental risk of tracking by the server (but the risk of tracking by IP address remains).

The privacy trade-offs here may be context specific. Stub resolvers will have a strong motivation to prefer privacy over latency since they often change location. However, recursive resolvers that use a small set of static IP addresses are more likely to prefer the reduced latency provided by session resumption and may consider this a valid reason to use resumption tickets even if the IP address changed between sessions.

Encrypted zone transfer (RFC9103) explicitly does not attempt to hide the identity of the parties involved in the transfer, but at the same time such transfers are not particularly latency sensitive. This means that applications supporting zone transfers may decide to apply the same protections as stub to recursive applications.

### 9.3. Privacy Issues With Address Validation Tokens

QUIC specifies address validation mechanisms in Section 8 of [RFC9000]. Use of an address validation token allows QUIC servers to avoid an extra RTT for new connections. Address validation tokens are typically tied to an IP address. QUIC clients normally only use these tokens when setting up a new connection from a previously used address. However, clients are not always aware that they are using a new address. This could be due to NAT, or because the client does not have an API available to check if the IP address has changed (which can be quite often for IPv6). There is a linkability risk if clients mistakenly use address validation tokens after unknowingly moving to a new location.

The recommendations in Section 6.5.3 mitigates this risk by tying the usage of the NEW\_TOKEN to that of session resumption, though this recommendation does not cover the case where the client is unaware of the address change.

### 9.4. Privacy Issues With Long Duration Sessions

A potential alternative to session resumption is the use of long duration sessions: if a session remains open for a long time, new queries can be sent without incurring connection establishment delays. It is worth pointing out that the two solutions have similar privacy characteristics. Session resumption may allow servers to keep track of the IP addresses of clients, but long duration sessions have the same effect.

In particular, a DoQ implementation might take advantage of the connection migration features of QUIC to maintain a session even if the client's connectivity changes, for example if the client migrates from a Wi-Fi connection to a cellular network connection, and then to another Wi-Fi connection. The server would be able to track the client location by monitoring the succession of IP addresses used by the long duration connection.

The recommendation in Section 6.5.4 mitigates the privacy concerns related to long duration sessions using multiple client addresses.



### 9.5. Traffic Analysis

Even though QUIC packets are encrypted, adversaries can gain information from observing packet lengths, in both queries and responses, as well as packet timing. Many DNS requests are emitted by web browsers. Loading a specific web page may require resolving dozens of DNS names. If an application adopts a simple mapping of one query or response per packet, or "one QUIC STREAM frame per packet", then the succession of packet lengths may provide enough information to identify the requested site.

Implementations SHOULD use the mechanisms defined in Section 6.4 to mitigate this attack.

## 10. IANA Considerations

### 10.1. Registration of DoQ Identification String

This document creates a new registration for the identification of DoQ in the "Application Layer Protocol Negotiation (ALPN) Protocol IDs" registry [RFC7301].

The "doq" string identifies DoQ:

Protocol: DoQ

Identification Sequence: 0x64 0x6F 0x71 ("doq")

Specification: This document

### 10.2. Reservation of Dedicated Port

For both TCP and UDP port 853 is currently reserved for 'DNS query-response protocol run over TLS/DTLS' [RFC7858].

However, the specification for DNS over DTLS (DoD) [RFC8094] is experimental, limited to stub to resolver, and no implementations or deployments currently exist to the authors' knowledge (even though several years have passed since the specification was published).

This specification proposes to additionally reserve the use of UDP port 853 for DoQ. QUIC version 1 was designed to be able to co-exist with other protocols on the same port, including DTLS, see Section 17.2 of [RFC9000]. This means that deployments that serve DNS over DTLS and DNS over QUIC (QUIC version 1) on the same port will be able to demultiplex the two due to the second most significant bit in each UDP payload. Such deployments ought to check the signatures of future versions or extensions (e.g., [I-D.ietf-quic-bit-grease]) of QUIC and DTLS before deploying them to serve DNS on the same port.

IANA is requested to update the following value in the "Service Name and Transport Protocol Port Number Registry" in the System Range. The registry for that range requires IETF Review or IESG Approval [RFC6335].

Service Name: domain-s

Port Number: 853

Transport Protocol(s): UDP

Assignee: IESG

Contact: IETF Chair

Description: DNS query-response protocol run over DTLS or QUIC

Reference: [RFC7858][RFC8094] This document

Additionally, IANA is requested to update the Description field for the corresponding TCP port 853 allocation to be 'DNS query-response protocol run over TLS' and to remove [RFC8094] from the TCP allocation's Reference field for consistency and clarity.

(UPDATE ON IANA REQUEST: THIS SENTENCE TO BE REMOVED BEFORE PUBLICATION) Review by the port experts on 13th December 2021 determined that the proposed updates to the existing port allocation were acceptable and will be made when this document is approved for publication.

### 10.3. Reservation of Extended DNS Error Code Too Early

IANA is requested to add the following value to the Extended DNS Error Codes registry [RFC8914]:

INFO-CODE: TBD

Purpose: Too Early

Reference: This document

#### 10.4. DNS over QUIC Error Codes Registry

IANA [SHALL add/has added] a registry for "DNS over QUIC Error Codes" on the "Domain Name System (DNS) Parameters" web page.

The "DNS over QUIC Error Codes" registry governs a 62-bit space. This space is split into three regions that are governed by different policies:

- \* Permanent registrations for values between 0x00 and 0x3f (in hexadecimal; inclusive), which are assigned using Standards Action or IESG Approval as defined in Section 4.9 and Section 4.10 of [RFC8126]
- \* Permanent registrations for values larger than 0x3f, which are assigned using the Specification Required policy ([RFC8126])
- \* Provisional registrations for values larger than 0x3f, which require Expert Review, as defined in Section 4.5 of [RFC8126].

Provisional reservations share the range of values larger than 0x3f with some permanent registrations. This is by design, to enable conversion of provisional registrations into permanent registrations without requiring changes in deployed systems. (This design is aligned with the principles set in Section 22 of [RFC9000].)

Registrations in this registry MUST include the following fields:

Value: The assigned codepoint.

Status: "Permanent" or "Provisional".

Contact: Contact details for the registrant.

In addition, permanent registrations MUST include:

Error: A short mnemonic for the parameter.

Specification: A reference to a publicly available specification for the value (optional for provisional registrations).

Description: A brief description of the error code semantics, which MAY be a summary if a specification reference is provided.

Provisional registrations of codepoints are intended to allow for private use and experimentation with extensions to DNS over QUIC. However, provisional registrations could be reclaimed and reassigned for another purpose. In addition to the parameters listed above, provisional registrations MUST include:

**Date:** The date of last update to the registration.

A request to update the date on any provisional registration can be made without review from the designated expert(s).

The initial contents of this registry are shown in Table 1 and all entries share the following fields:

**Status:** Permanent

**Contact:** DPRIVE WG

**Specification:** Section 5.3

Value	Error	Description
0x0	DOQ_NO_ERROR	No error
0x1	DOQ_INTERNAL_ERROR	Implementation error
0x2	DOQ_PROTOCOL_ERROR	Generic protocol violation
0x3	DOQ_REQUEST_CANCELLED	Request cancelled by client
0x4	DOQ_EXCESSIVE_LOAD	Closing a connection for excessive load
0x5	DOQ_UNSPECIFIED_ERROR	No error reason specified
0xd098ea5e	DOQ_ERROR_RESERVED	Alternative error code used for tests

Table 1: Initial DNS over QUIC Error Codes Entries

## 11. Acknowledgements

This document liberally borrows text from the HTTP-3 specification [I-D.ietf-quic-http] edited by Mike Bishop, and from the DoT specification [RFC7858] authored by Zi Hu, Liang Zhu, John Heidemann, Allison Mankin, Duane Wessels, and Paul Hoffman.

The privacy issue with 0-RTT data and session resumption were analyzed by Daniel Kahn Gillmor (DKG) in a message to the IETF "DPRIVE" working group [DNS0RTT].

Thanks to Tony Finch for an extensive review of the initial version of this draft, and to Robert Evans for the discussion of 0-RTT privacy issues. Early reviews by Paul Hoffman and Martin Thomson and interoperability tests conducted by Stephane Bortzmeyer helped improve the definition of the protocol.

Thanks also to Martin Thomson and Martin Duke for their later reviews focussing on the low level QUIC details which helped clarify several aspects of DoQ. Thanks to Andrey Meshkov, Loganaden Velvindron, Lucas Pardue, Matt Joras, Mirja Kuelewind, Brian Trammell and Phillip Hallam-Baker for their reviews and contributions.

## 12. References

### 12.1. Normative References

- [RFC1034] Mockapetris, P., "Domain names - concepts and facilities", STD 13, RFC 1034, DOI 10.17487/RFC1034, November 1987, <<https://www.rfc-editor.org/info/rfc1034>>.
- [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>>.
- [RFC1995] Ohta, M., "Incremental Zone Transfer in DNS", RFC 1995, DOI 10.17487/RFC1995, August 1996, <<https://www.rfc-editor.org/info/rfc1995>>.
- [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>>.
- [RFC5936] Lewis, E. and A. Hoenes, Ed., "DNS Zone Transfer Protocol (AXFR)", RFC 5936, DOI 10.17487/RFC5936, June 2010, <<https://www.rfc-editor.org/info/rfc5936>>.
- [RFC6891] Damas, J., Graff, M., and P. Vixie, "Extension Mechanisms for DNS (EDNS(0))", STD 75, RFC 6891, DOI 10.17487/RFC6891, April 2013, <<https://www.rfc-editor.org/info/rfc6891>>.

- [RFC7301] Friedl, S., Popov, A., Langley, A., and E. Stephan, "Transport Layer Security (TLS) Application-Layer Protocol Negotiation Extension", RFC 7301, DOI 10.17487/RFC7301, July 2014, <<https://www.rfc-editor.org/info/rfc7301>>.
- [RFC7766] 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>>.
- [RFC7830] Mayrhofer, A., "The EDNS(0) Padding Option", RFC 7830, DOI 10.17487/RFC7830, May 2016, <<https://www.rfc-editor.org/info/rfc7830>>.
- [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>>.
- [RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 8126, DOI 10.17487/RFC8126, June 2017, <<https://www.rfc-editor.org/info/rfc8126>>.
- [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>>.
- [RFC8310] Dickinson, S., Gillmor, D., and T. Reddy, "Usage Profiles for DNS over TLS and DNS over DTLS", RFC 8310, DOI 10.17487/RFC8310, March 2018, <<https://www.rfc-editor.org/info/rfc8310>>.
- [RFC8446] 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>>.
- [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>>.
- [RFC8914] Kumari, W., Hunt, E., Arends, R., Hardaker, W., and D. Lawrence, "Extended DNS Errors", RFC 8914, DOI 10.17487/RFC8914, October 2020, <<https://www.rfc-editor.org/info/rfc8914>>.

- [RFC9000] Iyengar, J., Ed. and M. Thomson, Ed., "QUIC: A UDP-Based Multiplexed and Secure Transport", RFC 9000, DOI 10.17487/RFC9000, May 2021, <<https://www.rfc-editor.org/info/rfc9000>>.
- [RFC9001] Thomson, M., Ed. and S. Turner, Ed., "Using TLS to Secure QUIC", RFC 9001, DOI 10.17487/RFC9001, May 2021, <<https://www.rfc-editor.org/info/rfc9001>>.
- [RFC9103] Toorop, W., Dickinson, S., Sahib, S., Aras, P., and A. Mankin, "DNS Zone Transfer over TLS", RFC 9103, DOI 10.17487/RFC9103, August 2021, <<https://www.rfc-editor.org/info/rfc9103>>.

## 12.2. Informative References

- [BCP195] Sheffer, Y., Holz, R., and P. Saint-Andre, "Recommendations for Secure Use of Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS)", BCP 195, RFC 7525, May 2015.
- Moriarty, K. and S. Farrell, "Deprecating TLS 1.0 and TLS 1.1", BCP 195, RFC 8996, March 2021.
- <<https://www.rfc-editor.org/info/bcp195>>
- [DNS0RTT] Kahn Gillmor, D., "DNS + 0-RTT", Message to DNS-Privacy WG mailing list, 6 April 2016, <<https://www.ietf.org/mail-archive/web/dns-privacy/current/msg01276.html>>.
- [I-D.ietf-dnsop-rfc8499bis] 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>>.
- [I-D.ietf-quic-bit-grease] Thomson, M., "Greasing the QUIC Bit", Work in Progress, Internet-Draft, draft-ietf-quic-bit-grease-02, 10 November 2021, <<https://www.ietf.org/archive/id/draft-ietf-quic-bit-grease-02.txt>>.
- [I-D.ietf-quic-http] Bishop, M., "Hypertext Transfer Protocol Version 3 (HTTP/3)", Work in Progress, Internet-Draft, draft-ietf-quic-http-34, 2 February 2021, <<https://www.ietf.org/archive/id/draft-ietf-quic-http-34.txt>>.

- [RFC1996] Vixie, P., "A Mechanism for Prompt Notification of Zone Changes (DNS NOTIFY)", RFC 1996, DOI 10.17487/RFC1996, August 1996, <<https://www.rfc-editor.org/info/rfc1996>>.
- [RFC3833] Atkins, D. and R. Austein, "Threat Analysis of the Domain Name System (DNS)", RFC 3833, DOI 10.17487/RFC3833, August 2004, <<https://www.rfc-editor.org/info/rfc3833>>.
- [RFC6335] Cotton, M., Eggert, L., Touch, J., Westerlund, M., and S. Cheshire, "Internet Assigned Numbers Authority (IANA) Procedures for the Management of the Service Name and Transport Protocol Port Number Registry", BCP 165, RFC 6335, DOI 10.17487/RFC6335, August 2011, <<https://www.rfc-editor.org/info/rfc6335>>.
- [RFC7828] 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>>.
- [RFC7873] Eastlake 3rd, D. and M. Andrews, "Domain Name System (DNS) Cookies", RFC 7873, DOI 10.17487/RFC7873, May 2016, <<https://www.rfc-editor.org/info/rfc7873>>.
- [RFC7942] Sheffer, Y. and A. Farrel, "Improving Awareness of Running Code: The Implementation Status Section", BCP 205, RFC 7942, DOI 10.17487/RFC7942, July 2016, <<https://www.rfc-editor.org/info/rfc7942>>.
- [RFC8094] Reddy, T., Wing, D., and P. Patil, "DNS over Datagram Transport Layer Security (DTLS)", RFC 8094, DOI 10.17487/RFC8094, February 2017, <<https://www.rfc-editor.org/info/rfc8094>>.
- [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>>.
- [RFC8490] Bellis, R., Cheshire, S., Dickinson, J., Dickinson, S., Lemon, T., and T. Pusateri, "DNS Stateful Operations", RFC 8490, DOI 10.17487/RFC8490, March 2019, <<https://www.rfc-editor.org/info/rfc8490>>.
- [RFC8932] 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>>.



[RFC9002] Iyengar, J., Ed. and I. Swett, Ed., "QUIC Loss Detection and Congestion Control", RFC 9002, DOI 10.17487/RFC9002, May 2021, <<https://www.rfc-editor.org/info/rfc9002>>.

[RFC9076] Wicinski, T., Ed., "DNS Privacy Considerations", RFC 9076, DOI 10.17487/RFC9076, July 2021, <<https://www.rfc-editor.org/info/rfc9076>>.

## Appendix A. The NOTIFY Service

This appendix discusses why it is considered acceptable to send NOTIFY (see [RFC1996]) in 0-RTT data.

Section 5.5 says "The 0-RTT mechanism SHOULD NOT be used to send DNS requests that are not "replayable" transactions". This specification supports sending a NOTIFY in 0-RTT data because although a NOTIFY technically changes the state of the receiving server, the effect of replaying NOTIFYS has negligible impact in practice.

NOTIFY messages prompt a secondary to either send an SOA query or an XFR request to the primary on the basis that a newer version of the zone is available. It has long been recognized that NOTIFYS can be forged and, in theory, used to cause a secondary to send repeated unnecessary requests to the primary. For this reason, most implementations have some form of throttling of the SOA/XFR queries triggered by the receipt of one or more NOTIFYS.

[RFC9103] describes the privacy risks associated with both NOTIFY and SOA queries and does not include addressing those risks within the scope of encrypting zone transfers. Given this, the privacy benefit of using DoQ for NOTIFY is not clear - but for the same reason, sending NOTIFY as 0-RTT data has no privacy risk above that of sending it using cleartext DNS.

## Appendix B. Notable Changes From Previous Versions

(RFC EDITOR NOTE: THIS SECTION TO BE REMOVED BEFORE PUBLICATION)

### B.1. Stream Mapping Incompatibility With Draft-02

Versions prior to -02 of this specification proposed a simpler mapping scheme of queries and responses to QUIC stream, which omitted the 2 byte length field and supported only a single response on a given stream. The more complex mapping in Section 5.2 was adopted to specifically cater for XFR support, however, it breaks compatibility with earlier versions.

Authors' Addresses

Christian Huitema  
Private Octopus Inc.  
427 Golfcourse Rd  
Friday Harbor, WA 98250  
United States of America  
Email: huitema@huitema.net

Sara Dickinson  
Sinodun IT  
Oxford Science Park  
Oxford  
OX4 4GA  
United Kingdom  
Email: sara@sinodun.com

Allison Mankin  
Salesforce  
Email: allison.mankin@gmail.com

DPRIVE  
Internet-Draft  
Intended status: Informational  
Expires: May 6, 2021

J. Livingood  
Comcast  
A. Mayrhofer  
nic.at GmbH  
B. Overeinder  
NLnet Labs  
November 02, 2020

DNS Privacy Requirements for Exchanges between Recursive Resolvers and  
Authoritative Servers  
draft-ietf-dprive-phase2-requirements-02

Abstract

This document describes requirements and considerations for adding confidentiality to DNS exchanges between recursive resolvers and authoritative servers. The intent of this document is to guide Internet Drafts in the DNS Private Exchange (DPRIVE) Working Group pertaining to recursive to authorized name servers, with the stated requirements and considerations.

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 May 6, 2021.

Copyright Notice

Copyright (c) 2020 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 & Scope . . . . .	2
2. Document Work Via GitHub . . . . .	3
3. Terminology . . . . .	3
4. Threat Model and Problem Statement . . . . .	3
5. Features to Provide Confidentiality . . . . .	4
5.1. Requirements . . . . .	4
5.2. Optional Features . . . . .	5
6. Security Considerations . . . . .	5
7. IANA Considerations . . . . .	6
8. Changelog . . . . .	6
9. APPENDIX: Perspectives and Use Cases . . . . .	6
9.1. The User Perspective and Use Cases . . . . .	6
9.2. The Operator Perspective and Use Cases . . . . .	7
9.3. The Implementor / Software Vendor Perspective and Use Cases . . . . .	9
10. References . . . . .	9
10.1. Normative References . . . . .	9
10.2. Informative References . . . . .	9
10.3. URIs . . . . .	10
Acknowledgments . . . . .	10
Authors' Addresses . . . . .	10

## 1. Introduction & Scope

The 2018 approved charter of the IETF DPRIVE Working Group [1] contains milestones related to confidentiality aspects of DNS transactions between the recursive resolver and authoritative name servers.

This is also reflected in the DPRIVE milestones [2], which (as of October 2019) contains two relevant milestones:

Develop requirements for adding confidentiality to DNS exchanges between recursive resolvers and authoritative servers (unpublished document).

Investigate potential solutions for adding confidentiality to DNS exchanges involving authoritative servers (Experimental).

This document intends to cover the first milestone for defining requirements for adding confidentiality to DNS exchanges between recursive resolvers and authoritative servers. This may in turn lead to progress in investigating, developing and standardizing potential experimental methods of meeting those requirements.

The motivation for this work is to extend the confidentiality methods used between a user's stub resolver and a recursive resolver to the recursive queries sent by recursive resolvers in response to a DNS lookup (when a cache miss occurs and the server must perform recursion to obtain a response to the query). A recursive resolver will send queries to root servers, to Top Level Domain (TLD) servers, to authoritative second level domain servers and potentially to other authoritative DNS servers and each of these query/response transactions presents an opportunity to extend the confidentiality of user DNS queries.

## 2. Document Work Via GitHub

The authors are working on this document via GitHub at <https://github.com/alex-nicat/ietf-dprive-phase2-requirements>. Feedback via pull requests and issues are invited there.

## 3. Terminology

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

This document also makes use of DNS Terminology defined in [RFC8499]

## 4. Threat Model and Problem Statement

Currently, protocols such as DoT provide encryption between the user's stub resolver and a recursive resolver. This potentially provides (1) protection from observation of end user DNS queries and responses, (2) protection from on-the-wire modification DNS queries or responses (including potentially forcing a downgrade to an unencrypted communication). Of course, observation and modification are still possible when performed by the recursive resolver, which decrypts queries, serves a response from cache or performs recursion to obtain a response (or synthesizes a response), and then encrypts the response and sends it back to the user's stub resolver.

But observation and modification threats still exist when a recursive resolver must perform DNS recursion, from the root to TLD to

authoritative servers. This document specifies requirements for filling those gaps.

## 5. Features to Provide Confidentiality

Confidentiality can be provided using a combination of techniques. This section describes the protocol implementation requirements and optional features that can be used to provide confidentiality.

### 5.1. Requirements

1. Each implementing party MUST be able to independently take incremental steps to meet requirements without the need for close coordination (e.g. loosely coupled)
2. A recursive resolver that supports recursive-to-authoritative DNS encryption MUST be able to determine whether or not a given authoritative name server to which it intends to connect also supports recursive-to-authoritative DNS encryption.
3. An authoritative name server that supports recursive-to-authoritative DNS encryption MUST be able to indicate that it supports recursive-to-authoritative DNS encryption in a way that facilitates (2).
4. An authoritative name server that does not support recursive-to-authoritative MUST NOT have to make any changes to facilitate (2).
5. The secure transport MUST only be established when referential integrity can be verified, MUST NOT have circular dependencies, and MUST be easily analyzed for diagnostic purposes.
6. Each implementing party MUST be able to negotiate use of a secure transport protocol or other DNS privacy protections in a manner that enables operators to perform appropriate performance and security monitoring, conduct relevant research, etc.
7. The authoritative domain owner or their administrator MUST have the option to specify their secure transport preferences (e.g. what specific protocols are supported). This SHALL include a method to publish a list of secure transport protocols (e.g. DoH, DoT and other future protocols not yet developed). In addition this SHALL include whether a secure transport protocol MUST always be used (non-downgradable) or whether a secure transport protocol MAY be used on an opportunistic (not strict) basis in recognition that some servers for a domain might use a secure transport protocol and others might not.

8. The authoritative domain owner or their administrator **MUST** have the option to vary their preferences on an authoritative nameserver to nameserver basis, due to the fact that administration of a particular DNS zone may be delegated to multiple parties (such as several CDNs), each of which may have different technical capabilities. This includes that some servers for a domain may use secure transport and others may not, as it is common for a given name server to be authoritative for multiple zones.
9. A given name server may be authoritative for multiple zones. As such, a name server **MAY** support use of a secure transport protocol for one zone, but not for another.
10. The specification of secure transport preferences **MUST** be performed using the DNS and **MUST NOT** depend on non-DNS protocols.
11. For secure transports using TLS, TLS 1.3 (or later versions) **MUST** be supported and downgrades from TLS 1.3 to prior versions **MUST** not occur.

## 5.2. Optional Features

1. QNAME minimisation **SHOULD** be implemented in all steps of recursion
2. DNSSEC validation **SHOULD** be performed
3. If an authoritative domain owner or their administrator indicates that (1) multiple secure transport protocols are available, or that (2) a secure transport and insecure transport are available, or that (3) no secure transport is available, then a recursive server **SHOULD** negotiate selection of an available transport protocol.

## 6. Security Considerations

Authoritative name servers will need to perform additional processing steps, such as completing key exchanges and maintaining persistent connections, when responding to queries from a recursive resolver that requests use of a secure transport protocol. These additional processing steps can have an impact on server availability if they are abused. As such, negotiation and use of a secure transport protocol should be done in a manner that does not increase the risk of an authoritative name server outage or lead a recursive server to fail to communicate with an authoritative name server.

## 7. IANA Considerations

This document has no actions for IANA.

## 8. Changelog

Version 00: Updated prior individual draft following IETF-106 feedback  
Version 01: Small editorial changes  
Version 02: Incorporate feedback and suggestions from Scott Hollenbeck, Duane Wessels and email discussions.

## 9. APPENDIX: Perspectives and Use Cases

The DNS resolving process involves several entities. These entities have different interests/requirements, and hence it does make sense to examine the interests of those entities separately - though in many cases their interests are aligned. Four different entities can be identified, and their interests are described in the following sections:

- o Users
- o Operators
- o Implementors / Software Developers
- o Researchers

### 9.1. The User Perspective and Use Cases

The privacy and confidentiality of Users (that is, users as in clients of recursive resolvers, which in turn forward/resolve the user's DNS requests by contacting authoritative servers) can be improved in several ways. We call this "minimisation of exposure", and there are currently three ways to reduce that exposure:

- o Qname minimisation [RFC7816], reducing the amount of information to what is absolutely necessary to resolve a query
- o Aggressive NSEC/local auth cache [RFC8198], reducing the amount of outgoing queries in the first place
- o Encryption, removing exposure of information while in transit

As recursors typically forwards queries received from the user to authoritative servers. This creates a transitive trust between the user and the recursor, as well as the authoritative server, since information created by the user is exposed to the authoritative



server. However, the user never has a chance to identify what data was exposed to which authoritative party (via which path).

Also, Users would want to be informed about the status of the connections which were made on their behalf, which adds a fourth point

Encryption/privacy status signaling

\*TODO\*: Actual requirements - what do users "want"? Start below:

## 9.2. The Operator Perspective and Use Cases

Operators of authoritative services have to provide stable and fast DNS services, and interact with a wide range of clients, not all of them authoritative servers. The operator side actually consists of two sides:

- o The "upstream" facing side of recursive resolvers
- o The "downstream" side of authoritative servers

Those two sides are typically operated by different entities, but many entities operate "both sides". Even though that is discouraged (\*TODO\* source), the two sides might even be operated on the same nameserver.

- o Maybe different technical perspectives for operators
  - \* Intelligence (sharing information)
  - \* SLD popularity for marketing
- o Focus initially on Second Level Domains (SLDs) initially
  - \* Is there a difference for TLDs vs. SLDs from a "protocol" perspective?
- o Monitoring and aggregated data analysis
- o Signaling provisioning information
  - \* New record type for finding authoritative server key and authentication? Use SRV? (Being able to use different servers for serving up DNS-over-{TCP,UDP} vs DNS-over-TLS responses may be valuable.

- \* Signal secure transport details (DNS-over-TLS, DNS-over-QUIC, EncryptedSNI, connectionless, etc.), perhaps in an extensible manner? Minimize RTTs and reduce need for trials.
  - \* Large provider use cases where the NS names are out of bailiwick for the zone (e.g. small number of distinct NS records serving 100k+ zones)
  - o EDNS client subnet (JL: Not sure ECS crosses the cost/benefit threshold to be included as a requirement and many CDNs that run auth servers will likely say ECS is quite operationally important)
  - o Decide between TLS and connectionless (such as COSE-based messages)
  - o Costs of TLS connection vs. connectionless
  - \* Technical solution, e.g. encryption of the DNS query, shouldn't enable an attack vector for DDoS or resource exhaustion. For example, only if the client uses DNS-over-TLS, the upstream query to the authoritative will be over DNS-over-TLS also. If the client uses UDP, the resolver won't invest resources in DNS-over-TLS to prevent a potential resource exhaustion attack.
  - \* Reuse connection state (if any) and examine resumption considerations
  - \* Minimize server-side state (eg, with session tickets)
  - \* Need empirical studies on capacity, traffic, attack vectors
  - \* Evaluate impact on architecture and footprint expansion
  - \* Analyze optimal persistent connection time/time-out
  - \* Analyze optimal number of persistent connections recursive resolvers should maintain
  - \* Consider operational concerns with respect to capabilities signaling
  - \* Develop a profile that has operational advantages for operators
- \*TODO\*: Actual requirements - what do operators "want"?

### 9.3. The Implementor / Software Vendor Perspective and Use Cases

Implementer requirements follows requirements from user and operator perspectives:

- o Non-functional requirements, e.g. diversity of implementations
- o Horizontal vs. vertical scaling, for example similar to http servers
- o Use of DANE [RFC6698] for authentication: strict vs. opportunistic
- o Incremental deployment
- o Cache reuse vs. downgrade? Does the cache need to be partitioned? When can an in-cache answer retrieved via cleartext be served encrypted to a recursive query?
- o (Use of TCP fast open) - but this might be a requirement for the actual encryption protocol

\*TODO\*: Actual requirements of implementors - essentially, they follow what Operators need?

## 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>>.
- [RFC8499] Hoffman, P., Sullivan, A., and K. Fujiwara, "DNS Terminology", BCP 219, RFC 8499, DOI 10.17487/RFC8499, January 2019, <<https://www.rfc-editor.org/info/rfc8499>>.

### 10.2. Informative 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>>.

- [RFC7816] Bortzmeyer, S., "DNS Query Name Minimisation to Improve Privacy", RFC 7816, DOI 10.17487/RFC7816, March 2016, <<https://www.rfc-editor.org/info/rfc7816>>.
- [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>>.

### 10.3. URIs

- [1] <https://datatracker.ietf.org/doc/charter-ietf-dprime/>
- [2] <https://datatracker.ietf.org/wg/dprime/about/>

### Acknowledgments

The authors would like to thank Scott Hollenbeck for his early feedback and providing text for the Internet Draft. We would also like to thank Duane Wessels for the feedback on the mailing list, and Peter van Dijk for his comments in personal conversations.

### Authors' Addresses

Jason Livingood  
Comcast

Email: [Jason\\_Livingood@comcast.com](mailto:Jason_Livingood@comcast.com)

Alexander Mayrhofer  
nic.at GmbH

Email: [alex.mayrhofer.ietf@gmail.com](mailto:alex.mayrhofer.ietf@gmail.com)

Benno Overeinder  
NLnet Labs

Email: [benno@NLnetLabs.nl](mailto:benno@NLnetLabs.nl)

dprive  
Internet-Draft  
Updates: 1995, 5936, 7766 (if approved)  
Intended status: Standards Track  
Expires: 27 November 2021

W. Toorop  
NLnet Labs  
S. Dickinson  
Sinodun IT  
S. Sahib  
Brave Software  
P. Aras  
A. Mankin  
Salesforce  
26 May 2021

DNS Zone Transfer-over-TLS  
draft-ietf-dprive-xfr-over-tls-12

Abstract

DNS zone transfers are transmitted in clear text, which gives attackers the opportunity to collect the content of a zone by eavesdropping on network connections. The DNS Transaction Signature (TSIG) mechanism is specified to restrict direct zone transfer to authorized clients only, but it does not add confidentiality. This document specifies the use of TLS, rather than clear text, to prevent zone content collection via passive monitoring of zone transfers: XFR-over-TLS (XoT). Additionally, this specification updates RFC1995 and RFC5936 with respect to efficient use of TCP connections, and RFC7766 with respect to the recommended number of connections between a client and server for each transport.

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 November 2021.

## 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. Document work via GitHub . . . . .	6
3. Terminology . . . . .	6
4. Threat Model . . . . .	6
5. Design Considerations for XoT . . . . .	7
6. Connection and Data Flows in Existing XFR Mechanisms . . . . .	8
6.1. AXFR Mechanism . . . . .	8
6.2. IXFR Mechanism . . . . .	10
6.3. Data Leakage of NOTIFY and SOA Message Exchanges . . . . .	11
6.3.1. NOTIFY . . . . .	11
6.3.2. SOA . . . . .	12
7. Updates to existing specifications . . . . .	12
7.1. Update to RFC1995 for IXFR-over-TCP . . . . .	13
7.2. Update to RFC5936 for AXFR-over-TCP . . . . .	14
7.3. Updates to RFC1995 and RFC5936 for XFR-over-TCP . . . . .	14
7.3.1. Connection reuse . . . . .	14
7.3.2. AXFRs and IXFRs on the same connection . . . . .	15
7.3.3. XFR limits . . . . .	15
7.3.4. The edns-tcp-keepalive EDNS0 Option . . . . .	15
7.3.5. Backwards compatibility . . . . .	16
7.4. Update to RFC7766 . . . . .	16
8. XoT specification . . . . .	17
8.1. Connection establishment . . . . .	17
8.2. TLS versions . . . . .	17
8.3. Port selection . . . . .	18
8.4. High level XoT descriptions . . . . .	18
8.5. XoT transfers . . . . .	20
8.6. XoT connections . . . . .	21
8.7. XoT vs ADoT . . . . .	21
8.8. Response RCODES . . . . .	22
8.9. AXoT specifics . . . . .	22
8.9.1. Padding AXoT responses . . . . .	22

8.10. IXoT specifics . . . . .	23
8.10.1. Condensation of responses . . . . .	23
8.10.2. Fallback to AXFR . . . . .	24
8.10.3. Padding of IXoT responses . . . . .	24
8.11. Name compression and maximum payload sizes . . . . .	24
9. Multi-primary Configurations . . . . .	25
10. Authentication mechanisms . . . . .	26
10.1. TSIG . . . . .	26
10.2. SIG(0) . . . . .	26
10.3. TLS . . . . .	26
10.3.1. Opportunistic TLS . . . . .	27
10.3.2. Strict TLS . . . . .	27
10.3.3. Mutual TLS . . . . .	27
10.4. IP Based ACL on the Primary . . . . .	28
10.5. ZONEMD . . . . .	28
11. XoT authentication . . . . .	28
12. Policies for Both AXoT and IXoT . . . . .	29
13. Implementation Considerations . . . . .	30
14. Operational Considerations . . . . .	30
15. IANA Considerations . . . . .	31
16. Implementation Status . . . . .	31
17. Security Considerations . . . . .	31
18. Acknowledgements . . . . .	32
19. Contributors . . . . .	32
20. Changelog . . . . .	32
21. Normative References . . . . .	35
22. Informative References . . . . .	37
Appendix A. XoT server connection handling . . . . .	39
A.1. Only listen on TLS on a specific IP address . . . . .	39
A.2. Client specific TLS acceptance . . . . .	39
A.3. SNI based TLS acceptance . . . . .	40
A.4. Transport specific response policies . . . . .	40
A.4.1. SNI based response policies . . . . .	41
Authors' Addresses . . . . .	41

## 1. Introduction

DNS has a number of privacy vulnerabilities, as discussed in detail in [I-D.ietf-dprive-rfc7626-bis]. Stub client to recursive resolver query privacy has received the most attention to date, with standards track documents for both DNS-over-TLS (DoT) [RFC7858] and DNS-over-HTTPS (DoH) [RFC8484], and a proposal for DNS-over-QUIC [I-D.ietf-dprive-dnsquic]. There is ongoing work on DNS privacy requirements for exchanges between recursive resolvers and authoritative servers [I-D.ietf-dprive-phase2-requirements] and some suggestions for how signaling of DoT support by authoritative nameservers might work. However, there is currently no RFC that specifically defines recursive to authoritative DNS-over-TLS (ADoT).

[I-D.ietf-dprive-rfc7626-bis] established that stub client DNS query transactions are not public and needed protection, but on zone transfer [RFC1995] [RFC5936] it says only:

"Privacy risks for the holder of a zone (the risk that someone gets the data) are discussed in [RFC5936] and [RFC5155]."

In what way is exposing the full contents of a zone a privacy risk? The contents of the zone could include information such as names of persons used in names of hosts. Best practice is not to use personal information for domain names, but many such domain names exist. The contents of the zone could also include references to locations that allow inference about location information of the individuals associated with the zone's organization. It could also include references to other organizations. Examples of this could be:

- \* Person-laptop.example.org
- \* MX-for-Location.example.org
- \* Service-tenant-from-another-org.example.org

Additionally, the full zone contents expose all the IP addresses of endpoints held in the DNS records which can make reconnaissance and attack targeting easier, particularly for IPv6 addresses or private networks. There may also be regulatory, policy or other reasons why the zone contents in full must be treated as private.

Neither of the RFCs mentioned in [I-D.ietf-dprive-rfc7626-bis] contemplates the risk that someone gets the data through eavesdropping on network connections, only via enumeration or unauthorized transfer as described in the following paragraphs.

Zone enumeration is trivially possible for DNSSEC zones which use NSEC; i.e. queries for the authenticated denial of existence records allow a client to walk through the entire zone contents. [RFC5155] specifies NSEC3, a mechanism to provide measures against zone enumeration for DNSSEC signed zones (a goal was to make it as hard to enumerate a DNSSEC signed zone as an unsigned zone). Whilst this is widely used, it has been demonstrated that zone walking is possible for precomputed NSEC3 using attacks such as those described in [NSEC3-attacks]. This prompted further work on an alternative mechanism for DNSSEC authenticated denial of existence - NSEC5 [I-D.vcelak-nsec5] - however questions remain over the practicality of this mechanism.



[RFC5155] does not address data obtained outside zone enumeration (nor does [I-D.vcelak-nsec5]). Preventing eavesdropping of zone transfers (this document) is orthogonal to preventing zone enumeration, though they aim to protect the same information.

[RFC5936] specifies using TSIG [RFC8945] for authorization of the clients of a zone transfer and for data integrity, but does not express any need for confidentiality, and TSIG does not offer encryption.

Section 8 of the NIST guide on 'Secure Domain Name System (DNS) Deployment' [nist-guide] discusses restricting access for zone transfers using ACLs and TSIG in more detail. It also discusses the possibility that specific deployments might choose to use a lower level network layer to protect zone transfers, e.g., IPSec.

It is noted that in all the common open source implementations such ACLs are applied on a per query basis (at the time of writing). Since requests typically occur on TCP connections authoritatives must therefore accept any TCP connection and then handling the authentication of each zone transfer (XFR) request individually.

Because both AXFR (authoritative transfer) and IXFR (incremental transfer) are typically carried out over TCP from authoritative DNS protocol implementations, encrypting zone transfers using TLS [RFC8499], based closely on DoT [RFC7858], seems like a simple step forward. This document specifies how to use TLS (1.3 or later) as a transport to prevent zone collection from zone transfers.

This document also updates the previous specifications for zone transfers to clarify and extend them, mainly with respect to TCP usage:

- \* RFC1995 (IXFR) and RFC5936 (AXFR) are both updated to add further specification on efficient use of TCP connections
- \* Section 6.2.2 of RFC7766 (DNS Transport over TCP - Implementation Requirements) is updated with a new recommendation about the number of connections between a client and server for each transport.

## 2. Document work via GitHub

[THIS SECTION TO BE REMOVED BEFORE PUBLICATION] The Github repository for this document is at <https://github.com/hanzhang0116/hzpa-dprive-xfr-over-tls> (<https://github.com/hanzhang0116/hzpa-dprive-xfr-over-tls>). Proposed text and editorial changes are very much welcomed there, but any functional changes should always first be discussed on the IETF DPRIVE WG (dns-privacy) mailing list.

## 3. Terminology

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

Privacy terminology is as described in Section 3 of [RFC6973].

DNS terminology is as described in [RFC8499]. Note that as in [RFC8499], the terms 'primary' and 'secondary' are used for two servers engaged in zone transfers.

DoT: DNS-over-TLS as specified in [RFC7858]

XFR-over-TCP: Used to mean both IXFR-over-TCP [RFC1995] and AXFR-over-TCP [RFC5936].

XoT: XFR-over-TLS mechanisms as specified in this document which apply to both AXFR-over-TLS and IXFR-over-TLS

AXoT: AXFR-over-TLS

IXoT: IXFR over-TLS

## 4. Threat Model

The threat model considered here is one where the current contents and size of the zone are considered sensitive and should be protected during transfer.

The threat model does not, however, consider the existence of a zone, the act of zone transfer between two entities, nor the identities of the nameservers hosting a zone (including both those acting as hidden primaries/secondaries or directly serving the zone) as sensitive information. The proposed mechanism does not attempt to obscure such information. The reasons for this include:

- \* much of this information can be obtained by various methods, including active scanning of the DNS
- \* an attacker who can monitor network traffic can relatively easily infer relations between nameservers simply from traffic patterns, even when some or all of the traffic is encrypted (in terms of current deployments)

The model does not consider attacks on the mechanisms that trigger a zone transfer, e.g., NOTIFY messages.

It is noted that simply using XoT will indicate a desire by the zone owner that the contents of the zone remain confidential and so could be subject to blocking (e.g., via blocking of port 853) if an attacker had such capabilities. However this threat is likely true of any such mechanism that attempts to encrypt data passed between nameservers, e.g., IPsec.

## 5. Design Considerations for XoT

The following principles were considered in the design for XoT:

- \* Confidentiality. Clearly using an encrypted transport for zone transfers will defeat zone content leakage that can occur via passive surveillance.
- \* Authentication. Use of single or mutual TLS (mTLS) authentication (in combination with access control lists (ACLs)) can complement and potentially be an alternative to TSIG.
- \* Performance.
  - Existing AXFR and IXFR mechanisms have the burden of backwards compatibility with older implementations based on the original specifications in [RFC1034] and [RFC1035]. For example, some older AXFR servers don't support using a TCP connection for multiple AXFR sessions or XFRs of different zones because they have not been updated to follow the guidance in [RFC5936]. Any implementation of XoT would obviously be required to implement optimized and interoperable transfers as described in [RFC5936], e.g., transfer of multiple zones over one connection.
  - Current usage of TCP for IXFR is sub-optimal in some cases i.e. connections are frequently closed after a single IXFR.

## 6. Connection and Data Flows in Existing XFR Mechanisms

The original specification for zone transfers in [RFC1034] and [RFC1035] was based on a polling mechanism: a secondary performed a periodic query for the SOA (start of zone authority) record (based on the refresh timer) to determine if an AXFR was required.

[RFC1995] and [RFC1996] introduced the concepts of IXFR and NOTIFY respectively, to provide for prompt propagation of zone updates. This has largely replaced AXFR where possible, particularly for dynamically updated zones.

[RFC5936] subsequently redefined the specification of AXFR to improve performance and interoperability.

In this document we use the term "XFR mechanism" to describe the entire set of message exchanges between a secondary and a primary that concludes in a successful AXFR or IXFR request/response. This set may or may not include

- \* NOTIFY messages
- \* SOA queries
- \* Fallback from IXFR to AXFR
- \* Fallback from IXFR-over-UDP to IXFR-over-TCP

The term is used to encompass the range of permutations that are possible and is useful to distinguish the 'XFR mechanism' from a single XFR request/response exchange.

### 6.1. AXFR Mechanism

The figure below provides an outline of an AXFR mechanism including NOTIFYS.

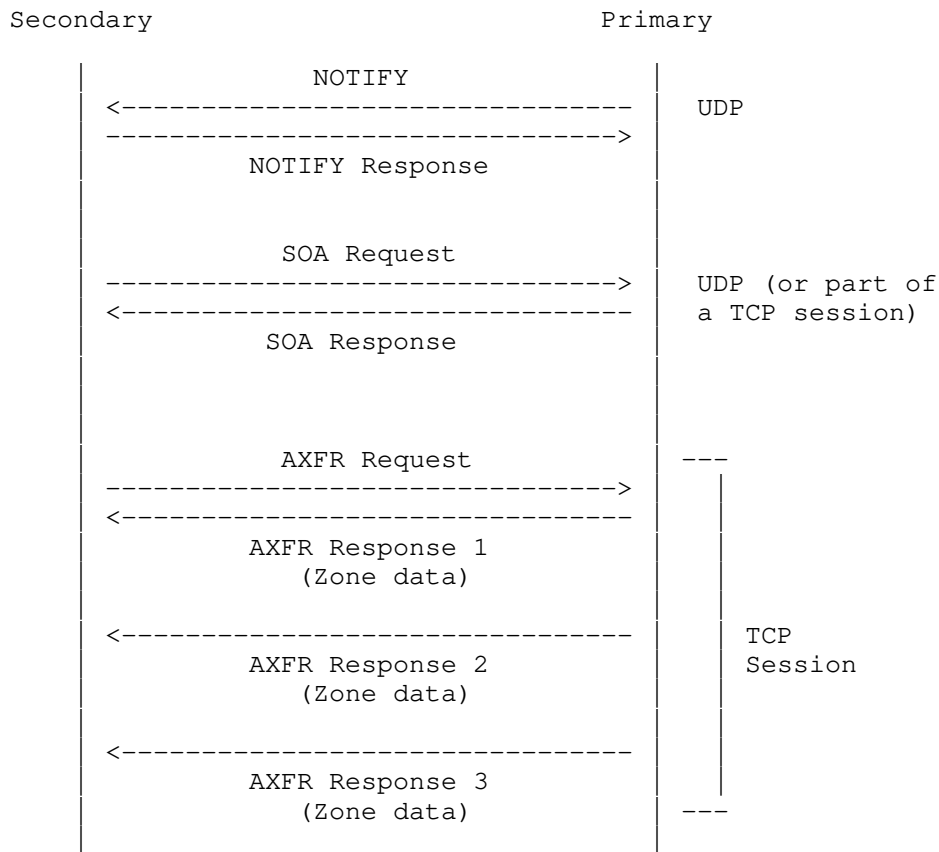


Figure 1. AXFR Mechanism

1. An AXFR is often (but not always) preceded by a NOTIFY (over UDP) from the primary to the secondary. A secondary may also initiate an AXFR based on a refresh timer or scheduled/triggered zone maintenance.
2. The secondary will normally (but not always) make a SOA query to the primary to obtain the serial number of the zone held by the primary.
3. If the primary serial is higher than the secondary's serial (using Serial Number Arithmetic [RFC1982]), the secondary makes an AXFR request (over TCP) to the primary after which the AXFR data flows in one or more AXFR responses on the TCP connection. [RFC5936] defines this specific step as an 'AXFR session' i.e. as an AXFR query message and the sequence of AXFR response messages returned for it.

[RFC5936] re-specified AXFR providing additional guidance beyond that provided in [RFC1034] and [RFC1035] and importantly specified that AXFR must use TCP as the transport protocol.

Additionally, sections 4.1, 4.1.1 and 4.1.2 of [RFC5936] provide improved guidance for AXFR clients and servers with regard to re-use of TCP connections for multiple AXFRs and AXFRs of different zones. However [RFC5936] was constrained by having to be backwards compatible with some very early basic implementations of AXFR. For example, it outlines that the SOA query can also happen on this connection. However, this can cause interoperability problems with older implementations that support only the trivial case of one AXFR per connection.

## 6.2. IXFR Mechanism

The figure below provides an outline of the IXFR mechanism including NOTIFYs.

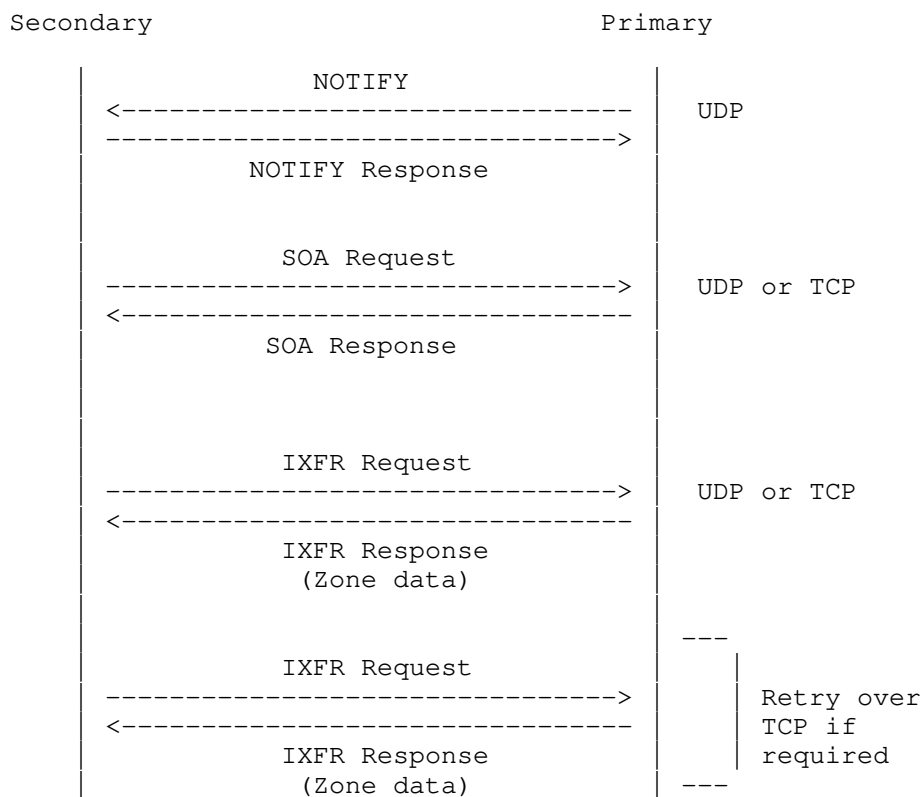


Figure 2. IXFR Mechanism

1. An IXFR is normally (but not always) preceded by a NOTIFY (over UDP) from the primary to the secondary. A secondary may also initiate an IXFR based on a refresh timer or scheduled/triggered zone maintenance.
2. The secondary will normally (but not always) make a SOA query to the primary to obtain the serial number of the zone held by the primary.
3. If the primary serial is higher than the secondaries serial (using Serial Number Arithmetic [RFC1982]), the secondary makes an IXFR request to the primary after which the primary sends an IXFR response.

[RFC1995] specifies that Incremental Transfer may use UDP if the entire IXFR response can be contained in a single DNS packet, otherwise, TCP is used. In fact it says:

"Thus, a client should first make an IXFR query using UDP."

So there may be a fourth step above where the client falls back to IXFR-over-TCP. There may also be a additional step where the secondary must fall back to AXFR because, e.g., the primary does not support IXFR.

However it is noted that most widely used open source authoritative nameserver implementations (including both [BIND] and [NSD]) do IXFR using TCP by default in their latest releases. For BIND, TCP connections are sometimes used for SOA queries but in general they are not used persistently and close after an IXFR is completed.

### 6.3. Data Leakage of NOTIFY and SOA Message Exchanges

This section presents a rationale for considering encrypting the other messages in the XFR mechanism.

Since the SOA of the published zone can be trivially discovered by simply querying the publicly available authoritative servers, leakage of this resource record (RR) via such a direct query is not discussed in the following sections.

#### 6.3.1. NOTIFY

Unencrypted NOTIFY messages identify configured secondaries on the primary.

[RFC1996] also states:

"If ANCOUNT>0, then the answer section represents an unsecure hint at the new RRset for this (QNAME,QCLASS,QTYPE).

But since the only QTYPE for NOTIFY defined at the time of this writing is SOA, this does not pose a potential leak.

#### 6.3.2. SOA

For hidden XFR servers (either primaries or secondaries), an SOA response directly from that server only additionally leaks the degree of SOA serial number lag of any downstream secondary of that server.

### 7. Updates to existing specifications

For convenience, the term 'XFR-over-TCP' is used in this document to mean both IXFR-over-TCP and AXFR-over-TCP and therefore statements that use that term update both [RFC1995] and [RFC5936], and implicitly also apply to XoT. Differences in behavior specific to XoT are discussed in Section 8.

Both [RFC1995] and [RFC5936] were published sometime before TCP was considered a first class transport for DNS. [RFC1995], in fact, says nothing with respect to optimizing IXFRs over TCP or re-using already open TCP connections to perform IXFRs or other queries. Therefore, there arguably is an implicit assumption that a TCP connection is used for one and only one IXFR request. Indeed, many major open source implementations take this approach (at the time of this writing). And whilst [RFC5936] gives guidance on connection re-use for AXFR, it pre-dates more recent specifications describing persistent TCP connections (e.g., [RFC7766], [RFC7828]), and AXFR implementations again often make less than optimal use of open connections.

Given this, new implementations of XoT will clearly benefit from specific guidance on TCP/TLS connection usage for XFR, because this will:

- \* result in more consistent XoT implementations with better interoperability
- \* remove any need for XoT implementations to support legacy behavior for XoT connections that XFR-over-TCP implementations have historically often supported

Therefore this document updates both the previous specifications for XFR-over-TCP ([RFC1995] and [RFC5936]) to clarify that



- \* Implementations MUST use [RFC7766] (DNS Transport over TCP - Implementation Requirements) to optimize the use of TCP connections.
- \* Whilst RFC7766 states that 'DNS clients SHOULD pipeline their queries' on TCP connections, it did not distinguish between XFRs and other queries for this behavior. It is now recognized that XFRs are not as latency sensitive as other queries, and can be significantly more complex for clients to handle, both because of the large amount of state that must be kept and because there may be multiple messages in the responses. For these reasons, it is clarified here that a valid reason for not pipelining queries is when they are all XFR queries i.e. clients sending multiple XFRs MAY choose not to pipeline those queries. Clients that do not pipeline XFR queries, therefore, have no additional requirements to handle out-of-order or intermingled responses (as described later), since they will never receive them.
- \* Implementations SHOULD use [RFC7828] (The edns-tcp-keepalive EDNS0 Option) to manage persistent connections (which is more flexible than using just fixed timeouts).

The following sections include detailed clarifications on the updates to XFR behavior implied in [RFC7766] and how the use of [RFC7828] applies specifically to XFR exchanges. They also discuss how IXFR and AXFR can reuse the same TCP connection.

For completeness, we also mention here the recent specification of extended DNS error (EDE) codes [RFC8914]. For zone transfers, when returning REFUSED to a zone transfer request from an 'unauthorized' client (e.g., where the client is not listed in an ACL for zone transfers or does not sign the request with a valid TSIG key), the extended DNS error code 18 (Prohibited) can also be sent.

#### 7.1. Update to RFC1995 for IXFR-over-TCP

For clarity - an IXFR-over-TCP server compliant with this specification MUST be able to handle multiple concurrent IXoT requests on a single TCP connection (for the same and different zones) and SHOULD send the responses as soon as they are available, which might be out-of-order compared to the requests.

## 7.2. Update to RFC5936 for AXFR-over-TCP

For clarity - an AXFR-over-TCP server compliant with this specification **MUST** be able to handle multiple concurrent AXoT sessions on a single TCP connection (for the same and different zones). The response streams for concurrent AXFRs **MAY** be intermingled and AXFR-over-TCP clients compliant with this specification which pipeline AXFR requests **MUST** be able to handle this.

## 7.3. Updates to RFC1995 and RFC5936 for XFR-over-TCP

### 7.3.1. Connection reuse

As specified, XFR-over-TCP clients **SHOULD** re-use any existing open TCP connection when starting any new XFR request to the same primary, and for issuing SOA queries, instead of opening a new connection. The number of TCP connections between a secondary and primary **SHOULD** be minimized (also see Section 7.4).

Valid reasons for not re-using existing connections might include:

- \* as already noted in [RFC7766], separate connections for different zones might be preferred for operational reasons. In this case, the number of concurrent connections for zone transfers **SHOULD** be limited to the total number of zones transferred between the client and server.
- \* reaching a configured limit for the number of outstanding queries or XFR requests allowed on a single TCP connection
- \* the message ID pool has already been exhausted on an open connection
- \* a large number of timeouts or slow responses have occurred on an open connection
- \* an edns-tcp-keepalive EDNS0 option with a timeout of 0 has been received from the server and the client is in the process of closing the connection (see Section 7.3.4)

If no TCP connections are currently open, XFR clients **MAY** send SOA queries over UDP or a new TCP connection.

### 7.3.2. AXFRs and IXFRs on the same connection

Neither [RFC1995] nor [RFC5936] explicitly discuss the use of a single TCP connection for both IXFR and AXFR requests. [RFC5936] does make the general statement:

"Non-AXFR session traffic can also use an open TCP connection."

We clarify here that implementations capable of both AXFR and IXFR and compliant with this specification SHOULD

- \* use the same TCP connection for both AXFR and IXFR requests to the same primary
- \* pipeline such requests (if they pipeline XFR requests in general) and MAY intermingle them
- \* send the response(s) for each request as soon as they are available i.e. responses MAY be sent intermingled

For some current implementations adding all the above functionality would introduce significant code complexity. In such a case, there will need to be an assessment of the trade-off between that and the performance benefits of the above for XFR.

### 7.3.3. XFR limits

The server MAY limit the number of concurrent IXFRs, AXFRs or total XFR transfers in progress, or from a given secondary, to protect server resources. Servers SHOULD return SERVFAIL if this limit is hit, since it is a transient error and a retry at a later time might succeed (there is no previous specification for this behavior).

### 7.3.4. The edns-tcp-keepalive EDNS0 Option

XFR clients that send the edns-tcp-keepalive EDNS0 option on every XFR request provide the server with maximum opportunity to update the edns-tcp-keepalive timeout. The XFR server may use the frequency of recent XFRs to calculate an average update rate as input to the decision of what edns-tcp-keepalive timeout to use. If the server does not support edns-tcp-keepalive, the client MAY keep the connection open for a few seconds ([RFC7766] recommends that servers use timeouts of at least a few seconds).

Whilst the specification for EDNS0 [RFC6891] does not specifically mention AXFRs, it does say

"If an OPT record is present in a received request, compliant responders MUST include an OPT record in their respective responses."

We clarify here that if an OPT record is present in a received AXFR request, compliant responders MUST include an OPT record in each of the subsequent AXFR responses. Note that this requirement, combined with the use of edns-tcp-keepalive, enables AXFR servers to signal the desire to close a connection (when existing transactions have competed) due to low resources by sending an edns-tcp-keepalive EDNS0 option with a timeout of 0 on any AXFR response. This does not signal that the AXFR is aborted, just that the server wishes to close the connection as soon as possible.

#### 7.3.5. Backwards compatibility

Certain legacy behaviors were noted in [RFC5936], with provisions that implementations may want to offer options to fallback to legacy behavior when interoperating with servers known to not support [RFC5936]. For purposes of interoperability, IXFR and AXFR implementations may want to continue offering such configuration options, as well as supporting some behaviors that were underspecified prior to this work (e.g., performing IXFR and AXFRs on separate connections). However, XoT connections should have no need to do so.

#### 7.4. Update to RFC7766

[RFC7766] made general implementation recommendations with regard to TCP/TLS connection handling:

"To mitigate the risk of unintentional server overload, DNS clients MUST take care to minimize the number of concurrent TCP connections made to any individual server. It is RECOMMENDED that for any given client/server interaction there SHOULD be no more than one connection for regular queries, one for zone transfers, and one for each protocol that is being used on top of TCP (for example, if the resolver was using TLS). However, it is noted that certain primary/ secondary configurations with many busy zones might need to use more than one TCP connection for zone transfers for operational reasons (for example, to support concurrent transfers of multiple zones)."

Whilst this recommends a particular behavior for the clients using TCP, it does not relax the requirement for servers to handle 'mixed' traffic (regular queries and zone transfers) on any open TCP/TLS connection. It also overlooks the potential that other transports might want to take the same approach with regard to using separate connections for different purposes.

This specification updates the above general guidance in [RFC7766] to provide the same separation of connection purpose (regular queries and zone transfers) for all transports being used on top of TCP.

Therefore, it is RECOMMENDED that for each protocol used on top of TCP in any given client/server interaction, there SHOULD be no more than one connection for regular queries and one for zone transfers.

As an illustration, it could be imagined that in future such an interaction could hypothetically include one or all of the following:

- \* one TCP connection for regular queries
- \* one TCP connection for zone transfers
- \* one TLS connection for regular queries
- \* one TLS connection for zone transfers
- \* one DoH connection for regular queries
- \* one DoH connection for zone transfers

Section 7.3.1 has provided specific details of reasons where more than one connection for a given transport might be required for zone transfers from a particular client.

## 8. XoT specification

### 8.1. Connection establishment

During connection establishment the Application-Layer Protocol Negotiation (ALPN) token "dot" [DoT-ALPN] MUST be selected in the TLS handshake.

### 8.2. TLS versions

All implementations of this specification MUST use only TLS 1.3 [RFC8446] or later.

### 8.3. Port selection

The connection for XoT SHOULD be established using port 853, as specified in [RFC7858], unless there is mutual agreement between the secondary and primary to use a port other than port 853 for XoT. There MAY be agreement to use different ports for AXoT and IXoT, or for different zones.

### 8.4. High level XoT descriptions

It is useful to note that in XoT, it is the secondary that initiates the TLS connection to the primary for a XFR request, so that in terms of connectivity, the secondary is the TLS client and the primary the TLS server.

The figure below provides an outline of the AXoT mechanism including NOTIFYs.

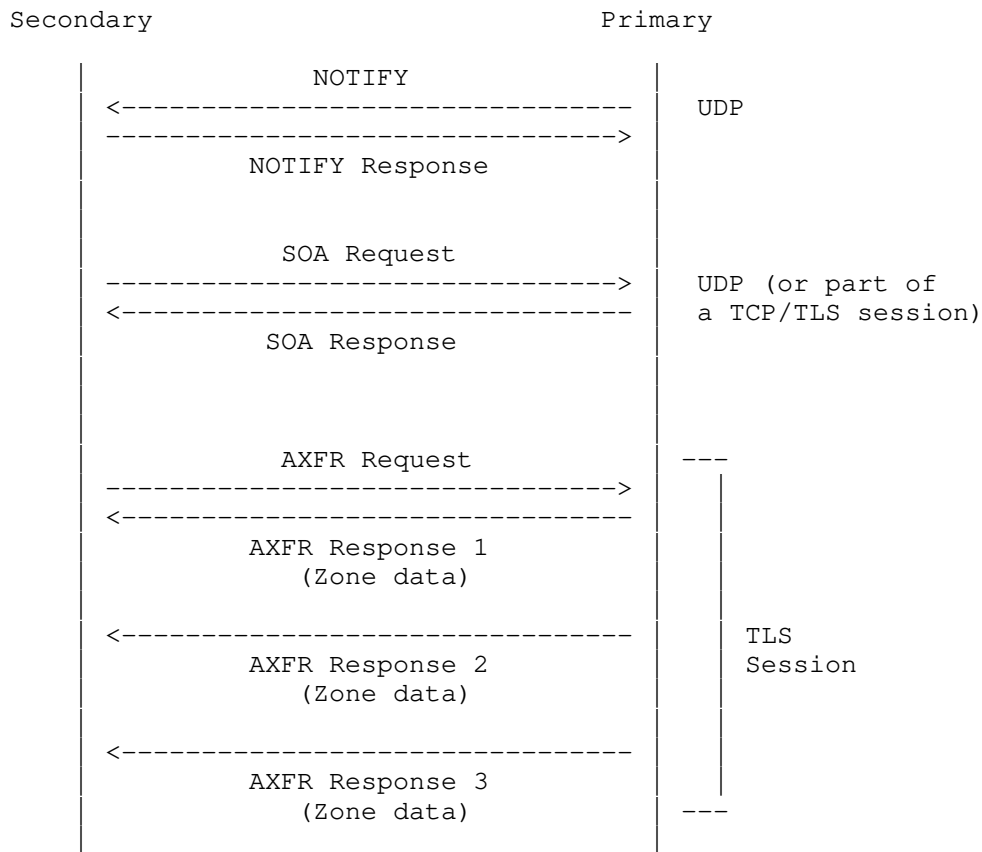


Figure 3. AXoT Mechanism

The figure below provides an outline of the IXoT mechanism including NOTIFYs.

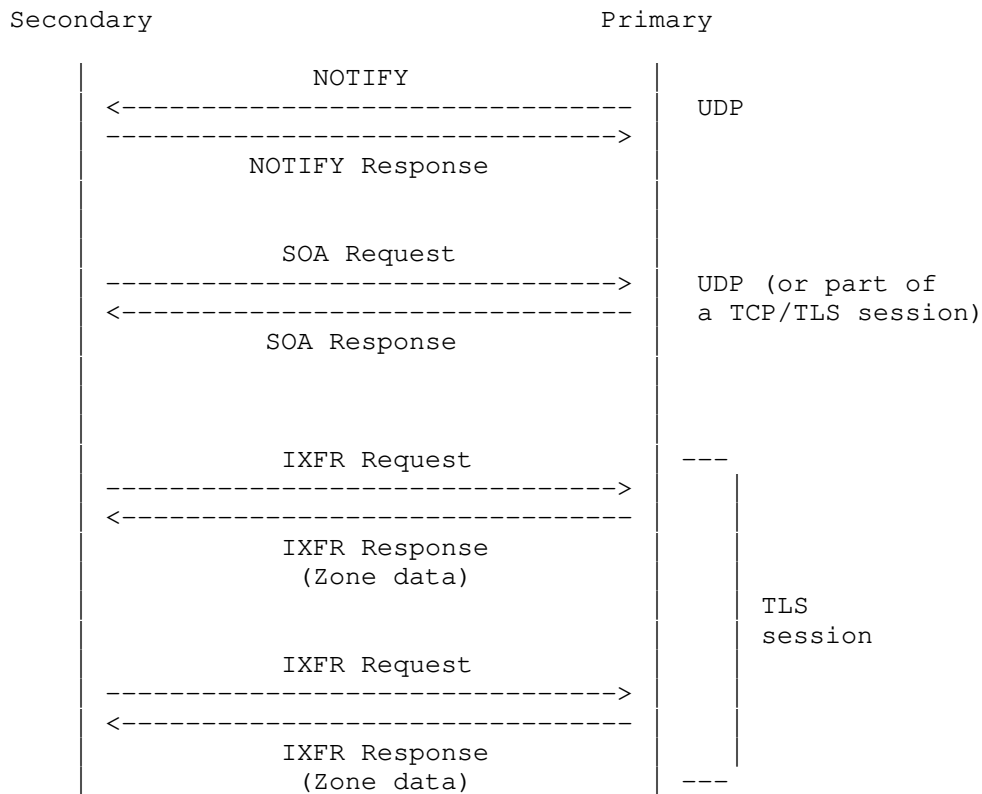


Figure 4. IXoT Mechanism

### 8.5. XoT transfers

For a zone transfer between two end points to be considered protected with XoT all XFR requests and response for that zone **MUST** be sent over TLS connections where at a minimum:

- \* the client **MUST** authenticate the server by use of an authentication domain name using a Strict Privacy Profile, as described in [RFC8310]
- \* the server **MUST** validate the client is authorized to request or proxy a zone transfer by using one or both of the following methods:
  - Mutual TLS (mTLS)



- an IP based ACL (which can be either per-message or per-connection) combined with a valid TSIG/SIG(0) signature on the XFR request

If only one method is selected then mTLS is preferred because it provides strong cryptographic protection at both endpoints.

Authentication mechanisms are discussed in full in Section 10 and the rationale for the above requirement in Section 11. Transfer group policies are discussed in Section 12.

#### 8.6. XoT connections

The details in Section 7 about, e.g., persistent connections and XFR message handling are fully applicable to XoT connections as well. However, any behavior specified here takes precedence for XoT.

If no TLS connections are currently open, XoT clients MAY send SOA queries over UDP or TCP, or TLS.

#### 8.7. XoT vs ADoT

As noted earlier, there is currently no specification for encryption of connections from recursive resolvers to authoritative servers. Some authoritatives are experimenting with ADoT and opportunistic encryption has also been raised as a possibility; it is therefore highly likely that use of encryption by authoritative servers will evolve in the coming years.

This raises questions in the short term with regard to TLS connection and message handling for authoritative servers. In particular, there is likely to be a class of authoritatives that wish to use XoT in the near future with a small number of configured secondaries, but that do not wish to support DoT for regular queries from recursives in that same time frame. These servers have to potentially cope with probing and direct queries from recursives and from test servers, and also potential attacks that might wish to make use of TLS to overload the server.

[RFC5936] clearly states that non-AXFR session traffic can use an open TCP connection, however, this requirement needs to be re-evaluated when considering applying the same model to XoT. Proposing that a server should also start responding to all queries received over TLS just because it has enabled XoT would be equivalent to defining a form of authoritative DoT. This specification does not propose that, but it also does not prohibit servers from answering queries unrelated to XFR exchanges over TLS. Rather, this specification simply outlines in later sections:

- \* how XoT implementations should utilize EDE codes in response to queries on TLS connections they are not willing to answer (see Section 8.8)
- \* the operational and policy options that a XoT server operator has with regard to managing TLS connections and messages (see Appendix A)

## 8.8. Response RCODES

XoT clients and servers MUST implement EDE codes. If a XoT server receives non-XoT traffic it is not willing to answer on a TLS connection, it SHOULD respond with REFUSED and the extended DNS error code 21 - Not Supported [RFC8914]. XoT clients should not send any further queries of this type to the server for a reasonable period of time (for example, one hour) i.e., long enough that the server configuration or policy might be updated.

Historically, servers have used the REFUSED RCODE for many situations, and so clients often had no detailed information on which to base an error or fallback path when queries were refused. As a result, the client behavior could vary significantly. XoT servers that refuse queries must cater for the fact that client behavior might vary from continually retrying queries regardless of receiving REFUSED to every query, or at the other extreme clients may decide to stop using the server over any transport. This might be because those clients are either non-XoT clients or do not implement EDE codes.

## 8.9. AXoT specifics

### 8.9.1. Padding AXoT responses

The goal of padding AXoT responses is two fold:

- \* to obfuscate the actual size of the transferred zone to minimize information leakage about the entire contents of the zone.
- \* to obfuscate the incremental changes to the zone between SOA updates to minimize information leakage about zone update activity and growth.

Note that the re-use of XoT connections for transfers of multiple different zones slightly complicates any attempt to analyze the traffic size and timing to extract information. Also, effective padding may require state to be kept as zones may grow and/or shrink over time.

It is noted here that, depending on the padding policies eventually developed for XoT, the requirement to obfuscate the total zone size might require a server to create 'empty' AXoT responses. That is, AXoT responses that contain no RR's apart from an OPT RR containing the EDNS(0) option for padding. For example, without this capability the maximum size that a tiny zone could be padded to would theoretically be limited if there had to be a minimum of 1 RR per packet.

However, as with existing AXFR, the last AXoT response message sent MUST contain the same SOA that was in the first message of the AXoT response series in order to signal the conclusion of the zone transfer.

[RFC5936] says:

"Each AXFR response message SHOULD contain a sufficient number of RRs to reasonably amortize the per-message overhead, up to the largest number that will fit within a DNS message (taking the required content of the other sections into account, as described below)."

'Empty' AXoT responses generated in order to meet a padding requirement will be exceptions to the above statement. For flexibility, future proofing and in order to guarantee support for future padding policies, we state here that secondary implementations MUST be resilient to receiving padded AXoT responses, including 'empty' AXoT responses that contain only an OPT RR containing the EDNS(0) option for padding.

Recommendation of specific policies for padding AXoT responses are out of scope for this specification. Detailed considerations of such policies and the trade-offs involved are expected to be the subject of future work.

## 8.10. IXoT specifics

### 8.10.1. Condensation of responses

[RFC1995] says condensation of responses is optional and MAY be done. Whilst it does add complexity to generating responses, it can significantly reduce the size of responses. However any such reduction might be offset by increased message size due to padding. This specification does not update the optionality of condensation for XoT responses.

#### 8.10.2. Fallback to AXFR

Fallback to AXFR can happen, for example, if the server is not able to provide an IXFR for the requested SOA. Implementations differ in how long they store zone deltas and how many may be stored at any one time.

Just as with IXFR-over-TCP, after a failed IXFR a IXoT client SHOULD request the AXFR on the already open XoT connection.

#### 8.10.3. Padding of IXoT responses

The goal of padding IXoT responses is to obfuscate the incremental changes to the zone between SOA updates to minimize information leakage about zone update activity and growth. Both the size and timing of the IXoT responses could reveal information.

IXFR responses can vary greatly in size from the order of 100 bytes for one or two record updates, to tens of thousands of bytes for large dynamic DNSSEC signed zones. The frequency of IXFR responses can also depend greatly on if and how the zone is DNSSEC signed.

In order to guarantee support for future padding policies, we state here that secondary implementations MUST be resilient to receiving padded IXoT responses.

Recommendation of specific policies for padding IXoT responses are out of scope for this specification. Detailed considerations of such padding policies, the use of traffic obfuscation techniques (such as 'dummy' traffic), and the trade-offs involved are expected to be the subject of future work.

#### 8.11. Name compression and maximum payload sizes

It is noted here that name compression [RFC1035] can be used in XFR responses to reduce the size of the payload, however, the maximum value of the offset that can be used in the name compression pointer structure is 16384. For some DNS implementations this limits the size of an individual XFR response used in practice to something around the order of 16kB. In principle, larger payload sizes can be supported for some responses with more sophisticated approaches (e.g., by pre-calculating the maximum offset required).

Implementations may wish to offer options to disable name compression for XoT responses to enable larger payloads. This might be particularly helpful when padding is used since minimizing the payload size is not necessarily a useful optimization in this case and disabling name compression will reduce the resources required to construct the payload.

## 9. Multi-primary Configurations

This model can provide flexibility and redundancy particularly for IXFR. A secondary will receive one or more NOTIFY messages and can send an SOA to all of the configured primaries. It can then choose to send an XFR request to the primary with the highest SOA (or based on other criteria, e.g., RTT).

When using persistent connections, the secondary may have a XoT connection already open to one or more primaries. Should a secondary preferentially request an XFR from a primary to which it already has an open XoT connection or the one with the highest SOA (assuming it doesn't have a connection open to it already)?

Two extremes can be envisaged here. The first one can be considered a 'preferred primary connection' model. In this case, the secondary continues to use one persistent connection to a single primary until it has reason not to. Reasons not to might include the primary repeatedly closing the connection, long query/response RTTs on transfers or the SOA of the primary being an unacceptable lag behind the SOA of an alternative primary.

The other extreme can be considered a 'parallel primary connection' model. Here, a secondary could keep multiple persistent connections open to all available primaries and only request XFRs from the primary with the highest serial number. Since normally the number of secondaries and primaries in direct contact in a transfer group is reasonably low this might be feasible if latency is the most significant concern.

Recommendation of a particular scheme is out of scope of this document, but implementations are encouraged to provide configuration options that allow operators to make choices about this behavior.

## 10. Authentication mechanisms

To provide context to the requirements in Section 8.5, this section provides a brief summary of some of the existing authentication and validation mechanisms (both transport independent and TLS specific) that are available when performing zone transfers. Section 11 then discusses in more details specifically how a combination of TLS authentication, TSIG and IP based ACLs interact for XoT.

We classify the mechanisms based on the following properties:

- \* 'Data Origin Authentication' (DO): Authentication that the DNS message originated from the party with whom credentials were shared, and of the data integrity of the message contents (the originating party may or may not be party operating the far end of a TCP/TLS connection in a 'proxy' scenario).
- \* 'Channel Confidentiality' (CC): Confidentiality of the communication channel between the client and server (i.e. the two end points of a TCP/TLS connection) from passive surveillance.
- \* 'Channel Authentication' (CA): Authentication of the identity of party to whom a TCP/TLS connection is made (this might not be a direct connection between the primary and secondary in a proxy scenario).

### 10.1. TSIG

TSIG [RFC8945] provides a mechanism for two or more parties to use shared secret keys which can then be used to create a message digest to protect individual DNS messages. This allows each party to authenticate that a request or response (and the data in it) came from the other party, even if it was transmitted over an unsecured channel or via a proxy.

Properties: Data origin authentication

### 10.2. SIG(0)

SIG(0) [RFC2931] similarly provides a mechanism to digitally sign a DNS message but uses public key authentication, where the public keys are stored in DNS as KEY RRs and a private key is stored at the signer.

Properties: Data origin authentication

### 10.3. TLS

### 10.3.1. Opportunistic TLS

Opportunistic TLS for DoT is defined in [RFC8310] and can provide a defense against passive surveillance, providing on-the-wire confidentiality. Essentially

- \* clients that know authentication information for a server SHOULD try to authenticate the server
- \* however they MAY fallback to using TLS without authentication and
- \* they MAY fallback to using cleartext if TLS is not available.

As such, it does not offer a defense against active attacks (e.g., an on path active attacker on the connection from client to server), and is not considered as useful for XoT.

Properties: None guaranteed.

### 10.3.2. Strict TLS

Strict TLS for DoT [RFC8310] requires that a client is configured with an authentication domain name (and/or SPKI pinset) that MUST be used to authenticate the TLS handshake with the server. If authentication of the server fails, the client will not proceed with the connection. This provides a defense for the client against active surveillance, providing client-to-server authentication and end-to-end channel confidentiality.

Properties: Channel confidentiality and channel authentication (of the server).

### 10.3.3. Mutual TLS

This is an extension to Strict TLS [RFC8310] which requires that a client is configured with an authentication domain name (and/or SPKI pinset) and a client certificate. The client offers the certificate for authentication by the server and the client can authenticate the server the same way as in Strict TLS. This provides a defense for both parties against active surveillance, providing bi-directional authentication and end-to-end channel confidentiality.

Properties: Channel confidentiality and mutual channel authentication.

#### 10.4. IP Based ACL on the Primary

Most DNS server implementations offer an option to configure an IP based Access Control List (ACL), which is often used in combination with TSIG based ACLs to restrict access to zone transfers on primary servers on a per query basis.

This is also possible with XoT, but it must be noted that, as with TCP, the implementation of such an ACL cannot be enforced on the primary until an XFR request is received on an established connection.

As discussed in Appendix A, an IP based per connection ACL could also be implemented where only TLS connections from recognized secondaries are accepted.

Properties: Channel authentication of the client.

#### 10.5. ZONEMD

For completeness, we also describe Message Digest for DNS Zones (ZONEMD) [RFC8976] here. The message digest is a mechanism that can be used to verify the content of a standalone zone. It is designed to be independent of the transmission channel or mechanism, allowing a general consumer of a zone to do origin authentication of the entire zone contents. Note that the current version of [RFC8976] states:

"As specified herein, ZONEMD is impractical for large, dynamic zones due to the time and resources required for digest calculation. However, The ZONEMD record is extensible so that new digest schemes may be added in the future to support large, dynamic zones."

It is complementary but orthogonal the above mechanisms; and can be used in conjunction with XoT, but is not considered further here.

#### 11. XoT authentication

It is noted that zone transfer scenarios can vary from a simple single primary/secondary relationship where both servers are under the control of a single operator to a complex hierarchical structure which includes proxies and multiple operators. Each deployment scenario will require specific analysis to determine which combination of authentication methods are best suited to the deployment model in question.



The XoT authentication requirement specified in Section 8.5 addresses the issue of ensuring that the transfers are encrypted between the two endpoints directly involved in the current transfers. The following table summarizes the properties of a selection of the mechanisms discussed in Section 10. The two letter acronyms for the properties are used below and (S) indicates the secondary and (P) indicates the primary.

Method	DO (S)	CC (S)	CA (S)	DO (P)	CC (P)	CA (P)
Strict TLS		Y	Y		Y	
Mutual TLS		Y	Y		Y	Y
ACL on primary						Y
TSIG	Y			Y		

Table 1

Table 1: Properties of Authentication methods for XoT

Based on this analysis it can be seen that:

- \* Using just mutual TLS can be considered a standalone solution since both end points are cryptographically authenticated
- \* Using secondary side Strict TLS with a primary side IP ACL and TSIG/SIG(0) combination provides sufficient protection to be acceptable.

Using just an IP ACL could be susceptible to attacks that can spoof TCP IP addresses, using TSIG/SIG(0) alone could be susceptible to attacks that were able to capture such messages should they be accidentally sent in clear text by any server with the key.

## 12. Policies for Both AXoT and IXoT

Whilst the protection of the zone contents in a transfer between two end points can be provided by the XoT protocol, the protection of all the transfers of a given zone requires operational administration and policy management.

We call the entire group of servers involved in XFR for a particular set of zones (all the primaries and all the secondaries) the 'transfer group'.

In order to assure the confidentiality of the zone information, the entire transfer group MUST have a consistent policy of using XoT. If any do not, this is a weak link for attackers to exploit. For clarification, this means that within any transfer group both AXFRs and IXFRs for a zone MUST all use XoT.

An individual zone transfer is not considered protected by XoT unless both the client and server are configured to use only XoT and the overall zone transfer is not considered protected until all members of the transfer group are configured to use only XoT with all other transfers servers (see Section 13).

A XoT policy MUST specify if

- \* mutual TLS is used and/or
- \* a IP ACL and TSIG/SIG(0) combination is used

Since this may require configuration of a number of servers who may be under the control of different operators the desired consistency could be hard to enforce and audit in practice.

Certain aspects of the Policies can be relatively easily tested independently, e.g., by requesting zone transfers without TSIG, from unauthorized IP addresses or over cleartext DNS. Other aspects such as if a secondary will accept data without a TSIG digest or if secondaries are using Strict as opposed to Opportunistic TLS are more challenging.

The mechanics of co-ordinating or enforcing such policies are out of the scope of this document but may be the subject of future operational guidance.

### 13. Implementation Considerations

Server implementations may want to also offer options that allow ACLs on a zone to specify that a specific client can use either XoT or TCP. This would allow for flexibility while clients are migrating to XoT.

Client implementations may similarly want to offer options to cater for the multi-primary case where the primaries are migrating to XoT.

### 14. Operational Considerations

If the options described in Section 13 are available, such configuration options MUST only be used in a 'migration mode', and therefore should be used with great care.

It is noted that use of a TLS proxy in front of the primary server is a simple deployment solution that can enable server side XoT.

## 15. IANA Considerations

None.

## 16. Implementation Status

[THIS SECTION TO BE REMOVED BEFORE PUBLICATION] This section records the status of known implementations of the protocol defined by this specification at the time of posting of this Internet-Draft, and is based on a proposal described in [RFC7942].

A summary of current behavior and implementation status can be found here: XoT implementation status (<https://dnsprivacy.org/wiki/display/DP/DNS+Privacy+Implementation+Status#DNSPrivacyImplementationStatus-XFR/XoTImplementationstatus>)

Specific recent activity includes:

1. The 1.9.2 version of Unbound (<https://github.com/NLnetLabs/unbound/blob/release-1.9.2/doc/Changelog>) includes an option to perform AXoT (instead of AXFR-over-TCP).
2. There are currently open pull requests against NSD to implement
  1. Connection re-use by default during XFR-over-TCP (<https://github.com/NLnetLabs/nsd/pull/145>)
  2. Client side XoT (<https://github.com/NLnetLabs/nsd/pull/149>)
3. Version 9.17.7 of BIND contained an initial implementation of DoT, implementation of XoT is planned for early 2021 (<https://gitlab.isc.org/isc-projects/bind9/-/issues/1784>)

Both items 1. and 2.2. listed above require the client (secondary) to authenticate the server (primary) using a configured authentication domain name if XoT is used.

## 17. Security Considerations

This document specifies a security measure against a DNS risk: the risk that an attacker collects entire DNS zones through eavesdropping on clear text DNS zone transfers.

This does not mitigate:

- \* the risk that some level of zone activity might be inferred by observing zone transfer sizes and timing on encrypted connections (even with padding applied), in combination with obtaining SOA records by directly querying authoritative servers.
- \* the risk that hidden primaries might be inferred or identified via observation of encrypted connections.
- \* the risk of zone contents being obtained via zone enumeration techniques.

Security concerns of DoT are outlined in [RFC7858] and [RFC8310].

## 18. Acknowledgements

The authors thank Tony Finch, Benno Overeinder, Shumon Huque and Tim Wicinski and many other members of DPRIVE for review and discussions.

The authors particularly thank Peter van Dijk, Ondrej Sury, Brian Dickson and several other open source DNS implementers for valuable discussion and clarification on the issue associated with pipelining XFR queries and handling out-of-order/intermingled responses.

## 19. Contributors

Significant contributions to the document were made by:

Han Zhang  
Salesforce  
San Francisco, CA  
United States

Email: hzhang@salesforce.com

## 20. Changelog

[THIS SECTION TO BE REMOVED BEFORE PUBLICATION]

draft-ietf-dprive-xfr-over-tls-12

- \* Changes from IESG review
- \* Add section 8.1 on the requirement to use the DoT ALPN
- \* Modify the one of the options for validation of a client from just an IP ACL to a combination of IP ACL and TSIG/SIG(0)

- Update Abstract and Introduction with clear descriptions of how earlier specifications are updated
- Add reference on NSEC3 attacks
- Justify use of SHOULD in sections 7.3.2 and 7.3.3.
- Clarify the Appendix is non-normative
- Numerous typos and editorial improvements.
- Use xml2rfc v3 (some format changes occur as a result)

draft-ietf-dprive-xfr-over-tls-11

- \* Fix definition update missed in -10

draft-ietf-dprive-xfr-over-tls-10

- \* Address issued raised from IETF Last Call

draft-ietf-dprive-xfr-over-tls-09

- \* Address issued raised in the AD review

draft-ietf-dprive-xfr-over-tls-08

- \* RFC2845 -> (obsoleted by) RFC8945
- \* I-D.ietf-dnsop-dns-zone-digest -> RFC8976
- \* Minor editorial changes + email update

draft-ietf-dprive-xfr-over-tls-07

- \* Reference RFC7942 in the implementation status section
- \* Convert the URIs that will remain on publication to references
- \* Correct typos in acknowledgments

draft-ietf-dprive-xfr-over-tls-06

- \* Update text relating to pipelining and connection reuse after WGLC comments.
- \* Add link to implementation status matrix

- \* Various typos

draft-ietf-dprive-xfr-over-tls-05

- \* Remove the open questions that received no comments.

- \* Add more detail to the implementation section

draft-ietf-dprive-xfr-over-tls-04

- \* Add Github repository

- \* Fix typos and references and improve layout.

draft-ietf-dprive-xfr-over-tls-03

- \* Remove propose to use ALPN

- \* Clarify updates to both RFC1995 and RFC5936 by adding specific sections on this

- \* Add a section on the threat model

- \* Convert all SVG diagrams to ASCII art

- \* Add discussions on concurrency limits

- \* Add discussions on Extended DNS error codes

- \* Re-work authentication requirements and discussion

- \* Add appendix discussion TLS connection management

draft-ietf-dprive-xfr-over-tls-02

- \* Significantly update descriptions for both AXoT and IXoT for message and connection handling taking into account previous specifications in more detail

- \* Add use of APLN and limitations on traffic on XoT connections.

- \* Add new discussions of padding for both AXoT and IXoT

- \* Add text on SIG(0)

- \* Update security considerations

- \* Move multi-primary considerations to earlier as they are related to connection handling

draft-ietf-dprive-xfr-over-tls-01

- \* Minor editorial updates
- \* Add requirement for TLS 1.3. or later

draft-ietf-dprive-xfr-over-tls-00

- \* Rename after adoption and reference update.
- \* Add placeholder for SIG(0) discussion
- \* Update section on ZONEMD

draft-hzpa-dprive-xfr-over-tls-02

- \* Substantial re-work of the document.

draft-hzpa-dprive-xfr-over-tls-01

- \* Editorial changes, updates to references.

draft-hzpa-dprive-xfr-over-tls-00

- \* Initial commit

[-@?I-D.ietf-tls-esni]

## 21. Normative References

- [DoT-ALPN] IANA, "TLS Application-Layer Protocol Negotiation (ALPN) Protocol IDs", 2021, <<https://www.iana.org/assignments/tls-extensiontype-values/tls-extensiontype-values.xhtml#alpn-protocol-ids>>.
- [RFC1034] Mockapetris, P., "Domain names - concepts and facilities", STD 13, RFC 1034, DOI 10.17487/RFC1034, November 1987, <<https://www.rfc-editor.org/info/rfc1034>>.
- [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>>.

- [RFC1995] Ohta, M., "Incremental Zone Transfer in DNS", RFC 1995, DOI 10.17487/RFC1995, August 1996, <<https://www.rfc-editor.org/info/rfc1995>>.
- [RFC1996] Vixie, P., "A Mechanism for Prompt Notification of Zone Changes (DNS NOTIFY)", RFC 1996, DOI 10.17487/RFC1996, August 1996, <<https://www.rfc-editor.org/info/rfc1996>>.
- [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>>.
- [RFC2931] Eastlake 3rd, D., "DNS Request and Transaction Signatures ( SIG(0)s )", RFC 2931, DOI 10.17487/RFC2931, September 2000, <<https://www.rfc-editor.org/info/rfc2931>>.
- [RFC5936] Lewis, E. and A. Hoenes, Ed., "DNS Zone Transfer Protocol (AXFR)", RFC 5936, DOI 10.17487/RFC5936, June 2010, <<https://www.rfc-editor.org/info/rfc5936>>.
- [RFC6973] Cooper, A., Tschofenig, H., Aboba, B., Peterson, J., Morris, J., Hansen, M., and R. Smith, "Privacy Considerations for Internet Protocols", RFC 6973, DOI 10.17487/RFC6973, July 2013, <<https://www.rfc-editor.org/info/rfc6973>>.
- [RFC7766] 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>>.
- [RFC7828] 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>>.
- [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>>.



- [RFC8310] Dickinson, S., Gillmor, D., and T. Reddy, "Usage Profiles for DNS over TLS and DNS over DTLS", RFC 8310, DOI 10.17487/RFC8310, March 2018, <<https://www.rfc-editor.org/info/rfc8310>>.
- [RFC8446] 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>>.
- [RFC8499] Hoffman, P., Sullivan, A., and K. Fujiwara, "DNS Terminology", BCP 219, RFC 8499, DOI 10.17487/RFC8499, January 2019, <<https://www.rfc-editor.org/info/rfc8499>>.
- [RFC8914] Kumari, W., Hunt, E., Arends, R., Hardaker, W., and D. Lawrence, "Extended DNS Errors", RFC 8914, DOI 10.17487/RFC8914, October 2020, <<https://www.rfc-editor.org/info/rfc8914>>.
- [RFC8945] Dupont, F., Morris, S., Vixie, P., Eastlake 3rd, D., Gudmundsson, O., and B. Wellington, "Secret Key Transaction Authentication for DNS (TSIG)", STD 93, RFC 8945, DOI 10.17487/RFC8945, November 2020, <<https://www.rfc-editor.org/info/rfc8945>>.

## 22. Informative References

- [BIND] ISC, "BIND 9.16.16", 2021, <<https://www.isc.org/bind/>>.
- [I-D.ietf-dprive-dnsoquic]  
Huitema, C., Mankin, A., and S. Dickinson, "Specification of DNS over Dedicated QUIC Connections", Work in Progress, Internet-Draft, draft-ietf-dprive-dnsoquic-02, 22 February 2021, <<https://tools.ietf.org/html/draft-ietf-dprive-dnsoquic-02>>.
- [I-D.ietf-dprive-phase2-requirements]  
Livingood, J., Mayrhofer, A., and B. Overeinder, "DNS Privacy Requirements for Exchanges between Recursive Resolvers and Authoritative Servers", Work in Progress, Internet-Draft, draft-ietf-dprive-phase2-requirements-02, 2 November 2020, <<https://tools.ietf.org/html/draft-ietf-dprive-phase2-requirements-02>>.
- [I-D.ietf-dprive-rfc7626-bis]  
Wicinski, T., "DNS Privacy Considerations", Work in Progress, Internet-Draft, draft-ietf-dprive-rfc7626-bis-09, 9 March 2021, <<https://tools.ietf.org/html/draft-ietf-dprive-rfc7626-bis-09>>.

- [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-10, 8 March 2021, <<https://tools.ietf.org/html/draft-ietf-tls-esni-10>>.
- [I-D.vcelak-nsec5]  
Vcelak, J., Goldberg, S., Papadopoulos, D., Huque, S., and D. Lawrence, "NSEC5, DNSSEC Authenticated Denial of Existence", Work in Progress, Internet-Draft, draft-vcelak-nsec5-08, 29 December 2018, <<https://tools.ietf.org/html/draft-vcelak-nsec5-08>>.
- [NSD] NLnet Labs, "NSD 4.3.6", 2021, <<https://www.nlnetlabs.nl/projects/nsd/about/>>.
- [NSEC3-attacks]  
Goldberf, S., Naor, N., Papadopoulos, D., Reyzin, L., Vasant, S., and A. Ziv, "Stretching NSEC3 to the Limit: Efficient Zone Enumeration Attacks on NSEC3 Variants", 2015, <<https://www.cs.bu.edu/~goldbe/papers/nsec3attacks.pdf>>.
- [RFC1982] Elz, R. and R. Bush, "Serial Number Arithmetic", RFC 1982, DOI 10.17487/RFC1982, August 1996, <<https://www.rfc-editor.org/info/rfc1982>>.
- [RFC5155] Laurie, B., Sisson, G., Arends, R., and D. Blacka, "DNS Security (DNSSEC) Hashed Authenticated Denial of Existence", RFC 5155, DOI 10.17487/RFC5155, March 2008, <<https://www.rfc-editor.org/info/rfc5155>>.
- [RFC6891] Damas, J., Graff, M., and P. Vixie, "Extension Mechanisms for DNS (EDNS(0))", STD 75, RFC 6891, DOI 10.17487/RFC6891, April 2013, <<https://www.rfc-editor.org/info/rfc6891>>.
- [RFC7942] Sheffer, Y. and A. Farrel, "Improving Awareness of Running Code: The Implementation Status Section", BCP 205, RFC 7942, DOI 10.17487/RFC7942, July 2016, <<https://www.rfc-editor.org/info/rfc7942>>.
- [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>>.

[RFC8976] Wessels, D., Barber, P., Weinberg, M., Kumari, W., and W. Hardaker, "Message Digest for DNS Zones", RFC 8976, DOI 10.17487/RFC8976, February 2021, <<https://www.rfc-editor.org/info/rfc8976>>.

[nist-guide] Chandramouli, R. and S. Rose, "Secure Domain Name System (DNS) Deployment Guide", 2013, <<https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-81-2.pdf>>.

## Appendix A. XoT server connection handling

This appendix provides a non-normative outline of the pros and cons of XoT server connection handling options.

For completeness, it is noted that an earlier version of the specification suggested using a XoT specific ALPN to negotiate TLS connections that supported only a limited set of queries (SOA, XRFs), however, this did not gain support. Reasons given included additional code complexity and the fact that XoT and ADoT are both DNS wire format and so should share the "dot" ALPN.

### A.1. Only listen on TLS on a specific IP address

Obviously a nameserver which hosts a zone and services queries for the zone on an IP address published in an NS record may wish to use a separate IP address for listening on TLS for XoT, only publishing that address to its secondaries.

Pros: Probing of the public IP address will show no support for TLS. ACLs will prevent zone transfer on all transports on a per query basis.

Cons: Attackers passively observing traffic will still be able to observe TLS connections to the separate address.

### A.2. Client specific TLS acceptance

Primaries that include IP based ACLs and/or mutual TLS in their authentication models have the option of only accepting TLS connections from authorized clients. This could be implemented using a proxy or directly in DNS implementation.

Pros: Connection management happens at setup time. The maximum number of TLS connections a server will have to support can be easily assessed. Once the connection is accepted the server might well be willing to answer any query on that connection since it is coming from a configured secondary and a specific response policy on the connection may not be needed (see below).

Cons: Currently, none of the major open source DNS authoritative implementations support such an option.

#### A.3. SNI based TLS acceptance

Primaries could also choose to only accept TLS connections based on an SNI that was published only to their secondaries.

Pros: Reduces the number of accepted connections.

Cons: As above. Also, this is not a recommended use of SNI. For SNIs sent in the clear, this would still allow attackers passively observing traffic to potentially abuse this mechanism. The use of Encrypted Client Hello [I-D.ietf-tls-esni] may be of use here.

#### A.4. Transport specific response policies

Some primaries might rely on TSIG/SIG(0) combined with per-query IP based ACLs to authenticate secondaries. In this case the primary must accept all incoming TLS/TCP connections and then apply a transport-specific response policy on a per query basis.

As an aside, whilst [RFC7766] makes a general purpose distinction in the advice to clients about their usage of connections (between regular queries and zone transfers) this is not strict and nothing in the DNS protocol prevents using the same connection for both types of traffic. Hence a server cannot know the intention of any client that connects to it, it can only inspect the messages it receives on such a connection and make per-query decisions about whether or not to answer those queries.

Example policies a XoT server might implement are:

- \* strict: REFUSE all queries on TLS connections except SOA and authorized XFR requests
- \* moderate: REFUSE all queries on TLS connections until one is received that is signed by a recognized TSIG/SIG(0) key, then answer all queries on the connection after that

- \* **complex:** apply a heuristic to determine which queries on a TLS connections to REFUSE
- \* **relaxed:** answer all non-XoT queries on all TLS connections with the same policy applied to TCP queries

**Pros:** Allows for flexible behavior by the server that could be changed over time.

**Cons:** The server must handle the burden of accepting all TLS connections just to perform XFRs with a small number of secondaries. Client behavior to REFUSED response is not clearly defined (see Section 8.8). Currently, none of the major open source DNS authoritative implementations offer an option for different response policies in different transports (but such functionality could potentially be implemented using a proxy).

#### A.4.1. SNI based response policies

In a similar fashion, XoT servers might use the presence of an SNI in the client hello to determine which response policy to initially apply to the TLS connections.

**Pros:** This has to potential to allow a clean distinction between a XoT service and any future DoT based service for answering recursive queries.

**Cons:** As above.

#### Authors' Addresses

Willem Toorop  
NLnet Labs  
Science Park 400  
Amsterdam

Email: [willem@nlnetlabs.nl](mailto:willem@nlnetlabs.nl)

Sara Dickinson  
Sinodun IT  
Magdalen Centre  
Oxford Science Park  
Oxford  
OX4 4GA  
United Kingdom

Email: [sara@sinodun.com](mailto:sara@sinodun.com)

Shivan Sahib  
Brave Software  
Vancouver, BC  
Canada

Email: shivankaulsahib@gmail.com

Pallavi Aras  
Salesforce  
Herndon, VA,  
United States

Email: paras@salesforce.com

Allison Mankin  
Salesforce  
Herndon, VA,  
United States

Email: allison.mankin@gmail.com