Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: April 23, 2021

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

October 20, 2020

Specification of DNS over Dedicated QUIC Connections
draft-ietf-dprive-dnsoquic-01

Abstract

This document describes the use of QUIC to provide transport privacy for DNS. The encryption provided by QUIC has similar properties to that provided by TLS, while QUIC transport eliminates the head-of-line blocking issues inherent with TCP and provides more efficient error corrections 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.

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 http://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 April 23, 2021.

Table of Contents

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 [I-D.ietf-quic-transport] [I-D.ietf-quic-tls]. DNS over QUIC is referred here as DoQ, in line with the "Terminology for DNS Transports and Location" [I-D.ietf-dnsop-terminology-ter]. 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 is not constrained by path MTU limitations on the size of DNS responses it can send.

4. Explore the characteristics of using QUIC as a DNS transport, versus other solutions like DNS over UDP [RFC1035], DoT [RFC7858], or DNS over HTTPS (DoH) [RFC8484].

In order to achieve these goals, the focus of this document is limited to the "stub to recursive resolver" scenario also addressed by DoT [RFC7858]. That is, the protocol described here works for queries and responses between stub clients and recursive servers. The specific non-goals of this document are:

1. No attempt is made to support AXFR "DNS Zone Transfer Protocol (AXFR)" [RFC5936] or IXFR "Incremental Zone Transfer in DNS" [RFC1885], as these mechanisms are not relevant to the stub to recursive resolver scenario.

2. No attempt is made to evade potential blocking of DNS over QUIC traffic by middleboxes.

3. No attempt to support server initiated transactions, are these are not relevant for the "stub to recursive resolver" scenario, see Section 3.5.

Users interested in zone transfers should continue using TCP based solutions and will also want to take note of work in progress to support "DNS Zone Transfer-over-TLS" [I-D.ietf-dprive-xfr-over-tls].

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.

In this document, Section 3 presents the reasoning that guided the proposed design. Section 4 specifies the actual mapping of DoQ. Section 5 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", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC8174].

3. Design Considerations

This section and its subsection present the design guidelines that were used for DoQ. This section is informative in nature.

3.1. Scope is Limited to the Stub to Resolver Scenario

Usage scenarios for the DNS protocol can be broadly classified in three groups: stub to recursive resolver, recursive resolver to authoritative server, and server to server. This design focuses only on the "stub to recursive resolver" scenario following the approach taken in DoT [RFC7858] and "Usage Profiles for DNS over TLS and DNS over DTLS" [RFC8310].

QUESTION: Should this document specify any aspects of configuration of discoverability differently to DoT?

No attempt is made to address the recursive to authoritative scenarios. Authoritative resolvers are discovered dynamically through NS records. It is noted that at the time of writing work is ongoing in the DPRIVE working group to attempt to address the analogous problem for DoT [I-D.ietf-dprive-phase2-requirements]. In

the absence of an agreed way for authoritative to signal support for QUIC transport, recursive resolvers would have to resort to some trial and error process. At this stage of QUIC deployment, this would be mostly errors, and does not seem attractive. This could change in the future.

The DNS protocol is also used for zone transfers. In the AXFR zone transfer scenario [RFC5936], the client emits a single AXFR query, and the server responds with a series of AXFR responses. This creates a unique profile, in which a query results in several responses. Supporting that profile would complicate the mapping of DNS queries over QUIC streams. Zone transfers are not used in the stub to recursive scenario that is the focus here, and seem to be currently well served by using DNS over TCP. There is no attempt to support either AXFR or IXFR in this proposed mapping of DNS to QUIC.

3.2. Provide DNS Privacy

DoT [RFC7858] defines how to mitigate some of the issues described in "DNS Privacy Considerations" [RFC7626] 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" [I-D.ietf-quic-tls], 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 7.

3.3. Design for Minimum Latency

QUIC is specifically designed to reduce the delay between HTTP queries and HTTP responses. This is achieved through three main components:

1. Support for 0-RTT data during session resumption.

2. Support for advanced error recovery procedures as specified in "QUIC Loss Detection and Congestion Control" [I-D.ietf-quic-recovery].

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. Fast resumption of QUIC connections to manage the disconnect-on-idle feature of QUIC without incurring retransmission time-outs.

4. 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 will be reflected in the mapping of DNS traffic to QUIC streams in Section 4.2.

3.4. No Specific Middlebox Bypass Mechanism

The mapping of DNS over QUIC is defined for minimal overhead and maximum performance. This means a different traffic profile than HTTP3 over QUIC. This difference can be noted by firewalls and middleboxes. There may be environments in which HTTP3 over QUIC will be able to pass through, but DoQ will be blocked by these middle boxes.

3.5. No Server Initiated Transactions

As stated in Section 1, this document does not specify support for server initiated transactions because these are not relevant for the "stub to recursive resolver" scenario. Note that "DNS Stateful Operations" (DSO) [RFC8490] are only applicable for DNS over TCP and DNS over TLS. DSO is not applicable to DNS over HTTP since HTTP has its own mechanism for managing sessions, and this is incompatible with the DSO; the same is true for DNS over QUIC.

4. Specifications

4.1. Connection Establishment

DoQ connections are established as described in the QUIC transport specification [I-D.ietf-quic-transport]. During connection establishment, DoQ support is indicated by selecting the ALPN token "doq" in the crypto handshake.

4.1.1. Draft Version Identification

RFC Editor's Note: Please remove this section prior to publication of a final version of this document.

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-dnsoquic-00 is identified using the string "doq-i00".

4.1.2. 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 8), unless it has mutual agreement with its clients to use a port other than TBD for DoQ. In order to use a port other than TBD, both clients and servers would need a configuration option in their software.

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 it has mutual agreement with its server to use a port other than port TBD for DoQ. Such another port MUST NOT be port 53 or port 853. This recommendation against use of port 53 for DoQ is to avoid confusion between DoQ and the use of DNS over UDP [RFC1035]. Similarly, using port 853 would cause confusion between DoQ and DNS over DTLS [RFC8094].

4.2. Stream Mapping and Usage

The mapping of DNS traffic over QUIC streams takes advantage of the QUIC stream features detailed in Section 2 of the QUIC transport specification [I-D.ietf-quic-transport].

The stub to resolver DNS traffic follows a simple pattern in which the client sends a query, and the server provides a response. This design specifies that for each subsequent query on a QUIC connection

the client MUST select the next available client-initiated bidirectional stream, in conformance with the QUIC transport specification [I-D.ietf-quic-transport].

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 on the same stream, and MUST indicate through the STREAM FIN mechanism that no further data will be sent on that stream.

Therefore, a single client initiated DNS transaction consumes a single stream. This means that the client's first query occurs on QUIC stream 0, the second on 4, and so on.

4.2.1. Transaction Errors

Peers normally complete transactions by sending a DNS response 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 initiator of the transaction by sending back a response with the Response Code set to SERVFAIL.

If a peer is incapable of sending a DNS response due to an internal error, it may issue a QUIC Stream Reset with error code DOQ_INTERNAL_ERROR. The corresponding transaction MUST be abandoned.

4.3. DoQ Error Codes

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

DOQ_NO_ERROR (0x00): No error. This is used when the connection or stream needs to be closed, but there is no error to signal.

DOQ_INTERNAL_ERROR (0x01): The DoQ implementation encountered an internal error and is incapable of pursuing the transaction or the connection.

DOQ_TRANSPORT_PARAMETER_ERROR (0x02): One or some of the transport parameters proposed by the peer are not acceptable.

4.4. Connection Management

Section 10 of the QUIC transport specification [I-D.ietf-quic-transport] specifies that connections can be closed in three ways:

- idle timeout
- immediate close
- stateless reset

Clients and servers implementing DNS over QUIC SHOULD negotiate use of the idle timeout. Closing on idle timeout is done without any packet exchange, which minimizes protocol overhead. Per section 10.2 of 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 5.6.2.

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 will check whether the idle time is sufficient lower than the idle timer. If it is, the client will send the DNS query over the existing connection. If not, the client will establish a new connection and send the query over that connection.

Clients MAY discard their connection to the server before the idle timeout expires. If they do that, they SHOULD close the connection explicitly, using QUIC's CONNECTION_CLOSE mechanisms, and indicating the Application reason "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 MAY receive a stateless reset indication. If a connection fails, all queries in progress over the connection MUST be considered failed, and a Server Failure (SERVFAIL, [RFC1035]) SHOULD be notified to the initiator of the transaction.

4.5. Connection Resume and 0-RTT

A stub resolver MAY take advantage of the connection resume mechanisms supported by QUIC transport [I-D.ietf-quic-transport] and QUIC TLS [I-D.ietf-quic-tls]. Stub resolvers SHOULD consider potential privacy issues associated with session resume before

deciding to use this mechanism. These privacy issues are detailed in Section 7.2.

When resuming a session, a stub resolver MAY take advantage of the 0-RTT mechanism supported by QUIC. The 0-RTT mechanism MUST NOT be used to send data that is not "replayable" transactions. For example, a stub resolver MAY transmit a Query as 0-RTT, but MUST NOT transmit an Update.

4.6. 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 maximum size of messages is controlled in QUIC by the transport parameters:

- initial_max_stream_data_bidi_local: when set by the client, specifies the amount of data that servers can send on a "response" stream without waiting for a MAX_STREAM_DATA frame.

- initial_max_stream_data_bidi_remote: when set by the server, specifies the amount of data that clients can send on a "query" stream without waiting for a MAX_STREAM_DATA frame.

Clients and servers MUST set these two parameters to the value 65535. If they receive a different value, they SHOULD close the QUIC connection with an application error "Invalid Parameter".

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.

5. Implementation Requirements

5.1. 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]. There is no need to authenticate the client’s identity in either scenario.

5.2. Fall Back to Other Protocols on Connection Failure

If the establishment of the DoQ connection fails, clients SHOULD 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, including timeouts, connection refusals, and QUIC handshake failures, and not request DoQ from them 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 DoQ connection failures.

5.3. Address Validation

Section 8 of the QUIC transport specification [I-D.ietf-quic-transport] 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 the QUIC transport specification [I-D.ietf-quic-transport]). 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 the QUIC transport specification [I-D.ietf-quic-transport]). 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.

5.4. DNS Message IDs

When sending queries over a QUIC connection, the DNS Message ID MUST be set to zero.

5.5. Padding

There are mechanisms specified for padding individual DNS messages in "The EDNS(0) Padding Option" [RFC7830] and for padding within QUIC packets (see Section 8.6 of the QUIC transport specification [I-D.ietf-quic-transport]).

Implementations SHOULD NOT use DNS options for padding individual DNS messages, because QUIC transport MAY transmit multiple STREAM frames containing separate DNS messages in a single QUIC packet. Instead, implementations SHOULD use QUIC PADDING frames to align the packet length to a small set of fixed sizes, aligned with the recommendations of the "Padding Policies for Extension Mechanisms for DNS (EDNS(0))" [RFC8467].

5.6. 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 attempts to specify which and how those considerations apply to DoQ.

5.6.1. Connection Reuse

Historic implementations of DNS stub resolvers are known to open and close TCP connections for each DNS query. To avoid excess QUIC connections, each with a single query, clients SHOULD reuse a single QUIC connection to the recursive resolver.

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.

5.6.2. Resource Management and Idle Timeout Values

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)
- Security Considerations (Section 10)

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 10.2 of the QUIC transport specification [I-D.ietf-quic-transport]. 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). If any message sent on a DoQ

connection contains an edns-tcp-keepalive EDNS(0) Option, this is a fatal error and the recipient of the defective message MUST forcibly abort the connection immediately.

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.

Clients that are willing to use QUIC's 0-RTT mechanism can reestablish connections and send transactions on the new connection with minimal delay overhead. These clients MAY chose low values of the idle timer.

5.7. Processing Queries in Parallel

As specified in Section 7 of "DNS Transport over TCP - Implementation Requirements" [RFC7766], 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.

5.8. Flow Control Mechanisms

Servers and Clients manage flow control as specified in QUIC.

Servers MAY use the "maximum stream ID" option of the QUIC transport to limit the number of streams opened by the client. This mechanism will effectively limit the number of DNS queries that a client can send on a single DoQ connection.

6. Security Considerations

The security considerations of DoQ should be comparable to those of DoT [RFC7858].

7. Privacy Considerations

DoQ is specifically designed to protect the DNS traffic between stub and resolver from observations by third parties, and thus protect the privacy of queries sent by the stub. However, the recursive resolver has full visibility of the stub's traffic, and could be used as an observation point, as discussed in the revision of "DNS Privacy Considerations" [I-D.ietf-dprive-rfc7626-bis]. These considerations do not differ between DoT and DoQ and are not discussed further here.

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 resume, which can provide linkability between successive client sessions.

These issues are developed in Section 7.1 and Section 7.2.

7.1. 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" [RFC7626]. The attack is partially mitigated by reducing the observability of this traffic. However, the risk is amplified for 0-RTT data, because the attacker might replay it at chosen times, several times.

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.

QUESTION: Should 0-RTT only be used with Opportunistic profiles (i.e. disabled by default for Strict only)?

7.2. Privacy Issues With Session Resume

The QUIC session resume mechanism reduces the cost of re-establishing sessions and enables 0-RTT data. There is a linkability issue associated with session resume, if the same resume token is used several times, but this risk is mitigated by the mechanisms incorporated in QUIC and in TLS 1.3. With these mechanisms, clients and servers can cooperate to avoid linkability by third parties. However, the server will always be able to link the resumed session to the initial session. This creates a virtual long duration session. The series of queries in that session can be used by the server to identify the client.

Enabling the server to link client sessions through session resume is probably not a large additional risk if the client's connectivity did not change between the sessions, since the two sessions can probably be correlated by comparing the IP addresses. On the other hand, if the addresses did change, the client SHOULD consider whether the linkability risk exceeds the performance benefits. This evaluation will obviously depend on the level of trust between stub and recursive.

7.3. 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 dozen 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 5.5 to mitigate this attack.

8. IANA Considerations

8.1. 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

8.2. Reservation of Dedicated Port

IANA is required to add the following value to 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], and such a review was requested using the early allocation process {{?RFC7120] for the well-known UDP port in this document. Since port 853 is reserved for 'DNS query-response

protocol run over TLS' consideration is requested for reserving port TBD for 'DNS query-response protocol run over QUIC'.

Service Name	domain-s
Transport Protocol(s)	TCP/UDP
Assignee	IESG
Contact	IETF Chair
Description	DNS query-response protocol run over QUIC
Reference	This document

8.2.1. Port number 784 for experimentations

RFC Editor's Note: Please remove this section prior to publication of a final version of this document.

Early experiments MAY use port 784. This port is marked in the IANA registry as unassigned.

9. 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 resume 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.

10. References

10.1. Normative References

[I-D.ietf-dnsop-terminology-ter]
Hoffman, P., "Terminology for DNS Transports and Location", draft-ietf-dnsop-terminology-ter-02 (work in progress), August 2020.

[I-D.ietf-quic-tls]
Thomson, M. and S. Turner, "Using TLS to Secure QUIC", draft-ietf-quic-tls-31 (work in progress), September 2020.

[I-D.ietf-quic-transport]
Iyengar, J. and M. Thomson, "QUIC: A UDP-Based Multiplexed and Secure Transport", draft-ietf-quic-transport-31 (work in progress), September 2020.

[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>.

[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>.

[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>.

[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>.

[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>.

[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>.

10.2. Informative References

[DNS0RTT] Kahn Gillmor, D., "DNS + 0-RTT", Message to DNS-Privacy WG mailing list, April 2016, <https://www.ietf.org/mail-archive/web/dns-privacy/current/msg01276.html>.

[I-D.ietf-dprive-phase2-requirements] Livingood, J., Mayrhofer, A., and B. Overeinder, "DNS Privacy Requirements for Exchanges between Recursive Resolvers and Authoritative Servers", draft-ietf-dprive-phase2-requirements-01 (work in progress), June 2020.

[I-D.ietf-dprive-rfc7626-bis] Wicinski, T., "DNS Privacy Considerations", draft-ietf-dprive-rfc7626-bis-07 (work in progress), October 2020.

[I-D.ietf-dprive-xfr-over-tls] Toorop, W., Dickinson, S., Sahib, S., Aras, P., and A. Mankin, "DNS Zone Transfer-over-TLS", draft-ietf-dprive-xfr-over-tls-02 (work in progress), July 2020.

[I-D.ietf-quic-http] Bishop, M., "Hypertext Transfer Protocol Version 3 (HTTP/3)", draft-ietf-quic-http-31 (work in progress), September 2020.

[I-D.ietf-quic-recovery] Iyengar, J. and I. Swett, "QUIC Loss Detection and Congestion Control", draft-ietf-quic-recovery-31 (work in progress), September 2020.

[RFC1885] Conta, A. and S. Deering, "Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6)", RFC 1885, DOI 10.17487/RFC1885, December 1995, <https://www.rfc-editor.org/info/rfc1885>.

[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>.

[RFC7626] Bortzmeyer, S., "DNS Privacy Considerations", RFC 7626, DOI 10.17487/RFC7626, August 2015, <https://www.rfc-editor.org/info/rfc7626>.

[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>.

[RFC7830] Mayrhofer, A., "The EDNS(0) Padding Option", RFC 7830, DOI 10.17487/RFC7830, May 2016, <https://www.rfc-editor.org/info/rfc7830>.

[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>.

[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>.

[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>.

Authors' Addresses

Christian Huitema
Private Octopus Inc.
427 Golfcourse Rd
Friday Harbor WA 98250
U.S.A

Email: huitema@huitema.net

Allison Mankin
Salesforce

Email: amankin@salesforce.com

Sara Dickinson
Sinodun IT
Oxford Science Park
Oxford OX4 4GA
U.K.

Email: sara@sinodun.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.

Table of Contents

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

Confidentialty 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:

- Users
- Operators
- Implementors / Software Developers
- 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:

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

- The "upstream" facing side of recursive resolvers
- 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.

- Maybe different technical perspectives for operators
  - Intelligence (sharing information)
  - SLD popularity for marketing
- Focus initially on Second Level Domains (SLDs) initially
  - Is there a difference for TLDs vs. SLDs from a "protocol" perspective?
- Monitoring and aggregated data analysis
- 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)

- 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)

- Decide between TLS and connectionless (such as COSE-based messages)

- 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:

- Non-functional requirements, e.g. diversity of implementations
- Horizontal vs. vertical scaling, for example similar to http servers
- Use of DANE [RFC6698] for authentication: strict vs. opportunistic
- Incremental deployment
- 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?
- (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-dprive/

[2] https://datatracker.ietf.org/wg/dprive/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

Alexander Mayrhofer
nic.at GmbH

Email: alex.mayrhofer.ietf@gmail.com

Benno Overeinder
NLnet Labs

Email: benno@NLnetLabs.nl

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

W. Toorop
NLnet Labs
S. Dickinson
Sinodun IT
S. Sahib
P. Aras
A. Mankin
Salesforce

November 23, 2020

DNS Zone Transfer-over-TLS
draft-ietf-dprive-xfr-over-tls-04

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 use of TLS, rather then clear text, to prevent zone content collection via passive monitoring of zone transfers: XFR-over-TLS (XoT). Additionally, this specification updates RFC1995, RFC5936 and RFC7766.

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 http://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 27, 2021.

Table of Contents

1. Introduction

DNS has a number of privacy vulnerabilities, as discussed in detail in [RFC7626]. 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-dnsoquic]. 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 authoritatives might work, e.g., [I-D.vandijk-dprive-ds-dot-signal-and-pin]. However there is currently no RFC that specifically defines recursive to authoritative DNS-over-TLS (ADoT).

[RFC7626] 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

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 [RFC7626] 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 existences 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 an DNSSEC signed zone as an unsigned zone). Whilst this is widely used, zone walking is now possible with NSEC3 due to crypto-breaking advances. This has 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 draft) is orthogonal to preventing zone enumeration, though they aim to protect the same information.

[RFC5936] specifies using TSIG [RFC2845] 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. Some operators use SSH tunneling or IPSec to encrypt the transfer data.

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 is noted that in all the common open source implementations such ACLs are applied on a per query basis. Since requests typically occur on TCP connections authoritatives must cater for accepting any TCP connection and then handling the authentication of each XFR request individually.

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

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

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

Note that in this document we choose to use the terms 'primary' and 'secondary' for two servers engaged in zone transfers.

DNS terminology is as described in [RFC8499].

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: Generic XFR-over-TLS mechanisms as specified in this document

AXoT: AXFR-over-TLS

IXoT: IXFR over-TLS

4. Use Cases for XFR-over-TLS

- 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 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 XFR-over-TLS (XoT) would obviously be required to implement optimized and interoperable transfers as described in [RFC5936], e.g., transfer of multiple zones over one connection.

- Performance. Current usage of TCP for IXFR is sub-optimal in some cases i.e. connections are frequently closed after a single IXFR.

4.1. 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 mechanisms 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

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. 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 SOA query (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 phrase "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 encompasses the range of permutations that are possible and is useful to distinguish the 'XFR mechanism' from a single XFR request/response exchange.

5.1. AXFR Mechanism

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

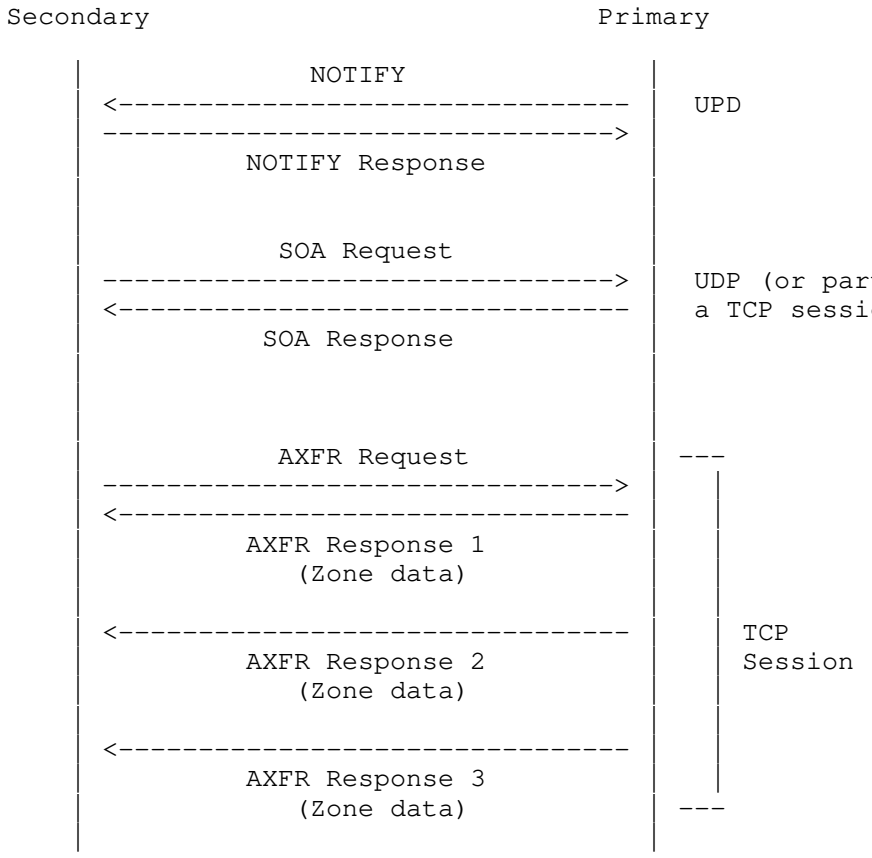

Figure 1. AXFR Mechanism

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 secondaries 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.

5.2. IXFR Mechanism

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

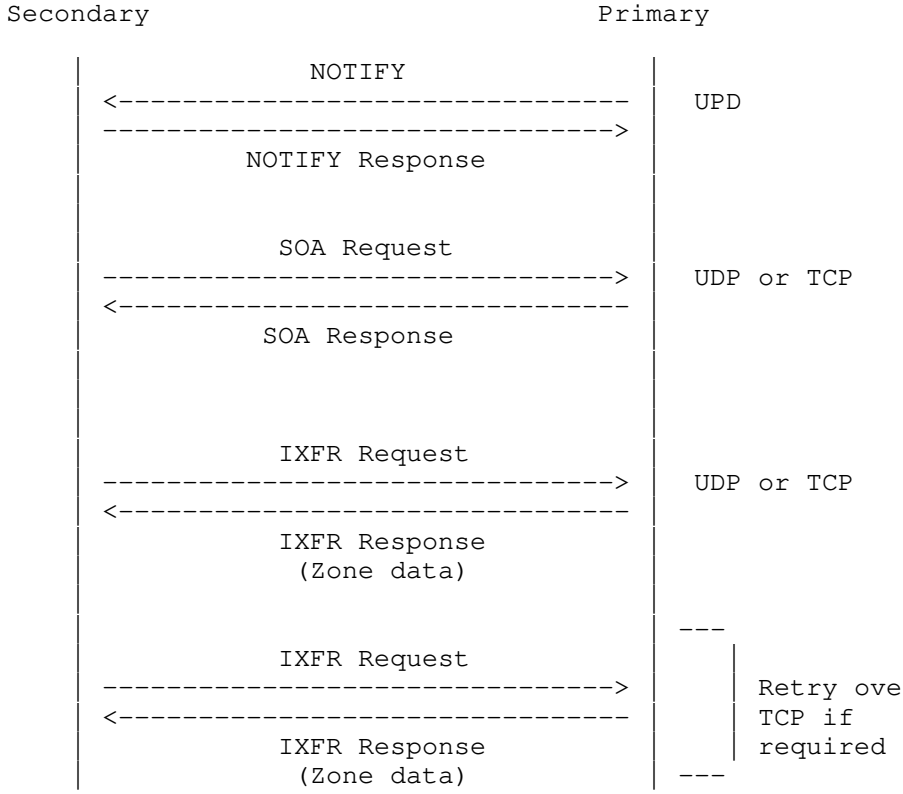

Figure 1. IXFR Mechanism

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 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 fourth 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 [1] and NSD [2]) 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.

5.3. Data Leakage of NOTIFY and SOA Message Exchanges

This section attempts to 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 RR is not discussed in the following sections.

5.3.1. 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 supported QTYPE for NOTIFY is SOA, this does not pose a potential leak.

5.3.2. SOA

For hidden primaries or secondaries the SOA response leaks only the degree of lag of any downstream secondary.

6. Updates to existing specifications

For convenience, the phrase 'XFR-over-TCP' is used in this document to mean both IXFR-over-TCP and AXFR-over-TCP and therefore statements that use it update both [RFC1995] and [RFC5936], and implicitly also apply to XoT. Differences in behavior specific to XoT are discussed in Section 7.

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 (probably unintentional) that a TCP connection is used for one and only one IXFR request. Indeed, several open source implementations currently take this approach. 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 that XFR-over-TCP implementations have historically often supported

Therefore this document updates both the previous specifications for XFR-over-TCP to clarify that implementations MUST use [RFC7766] (DNS Transport over TCP - Implementation Requirements) to optimize the use of TCP connections and SHOULD use [RFC7828] (The edns-tcp-keepalive EDNS0 Option) to manage persistent connections.

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. It also discusses 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 to an 'unauthorized' client (e.g. where the client is not listed in an ACL for zone transfers or does not sign the request with the correct TSIG key), the extended DNS error code 18 (Prohibited) can also be sent.

6.1. 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.

6.2. 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 MUST be able to handle this.

6.3. Updates to RFC1995 and RFC5936 for XFR-over-TCP

6.3.1. 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 6.4).

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

- 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 6.3.4)

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

6.3.2. 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 state:

> "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 request and MAY intermingle them
- send the response(s) for each request as soon as they are available i.e. responses MAY be sent intermingled

6.3.3. 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.

[OPEN QUESTION] Testing has shown that BIND returns SERVFAIL if the limit on concurrent transfers is reached since this is regarded as a soft limit and a retry can/should succeed. Should there be a specific recommendation here about what is returned re: SERVFAIL vs REFUSED?

[OPEN QUESTION] Is there a desire to define an additional XFR specific EDE code so that a client can determine why a specific XFR request was declined in this case e.g., Max concurrent XFR: too may concurrent transfers in progress. It could potentially contain a retry delay, or at least clients can apply a reasonable back-off for the retry. This could avoid retry storms which have been observed to actually increase the load on primaries in certain scenarios.

6.3.4. 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.

6.3.5. Backwards compatibility

Certain legacy behaviors were noted in [RFC5936], with provisos that implementations may want to offer options to fallback to legacy behavior when interoperating with servers known not to 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 implementations should have no need to do so.

6.4. 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 for XoT updates the 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

We provide specific details in the later sections of reasons where more than one connection for a given transport might be required for zone transfers from a particular client.

7. XoT specification

7.1. TLS versions

For improved security all implementations of this specification MUST use only TLS 1.3 [RFC8446] or later.

7.2. 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.

7.3. 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.

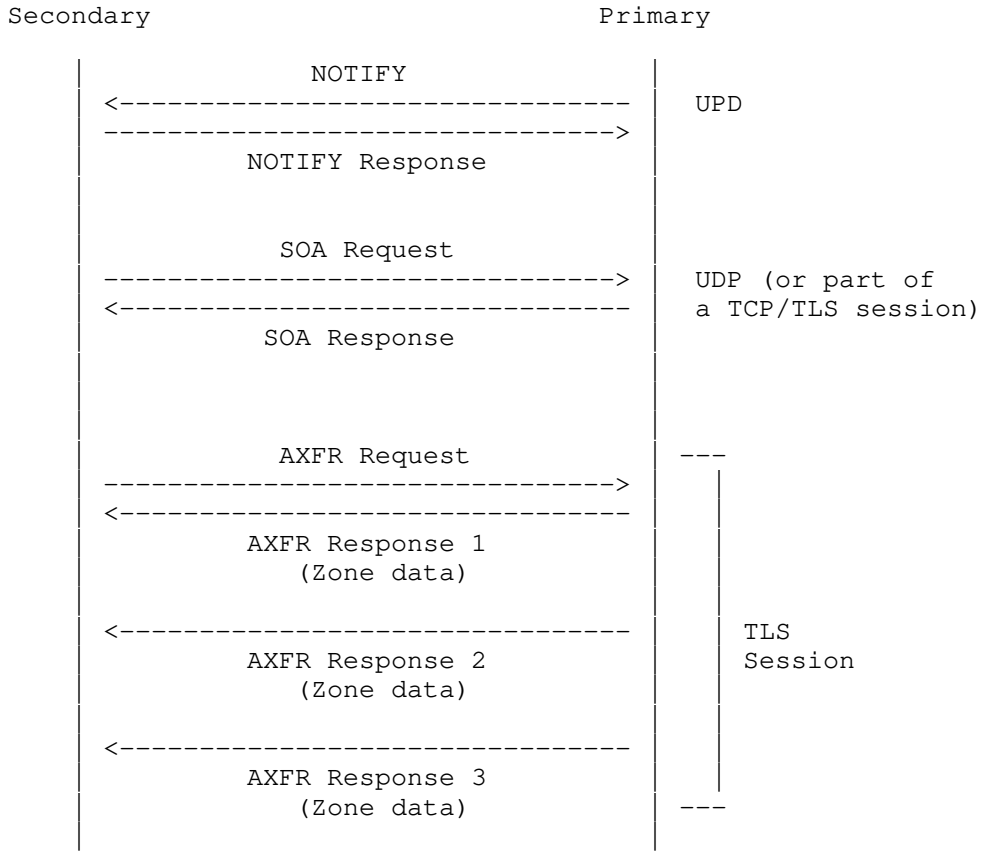

Figure 3. AXoT Mechanism

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

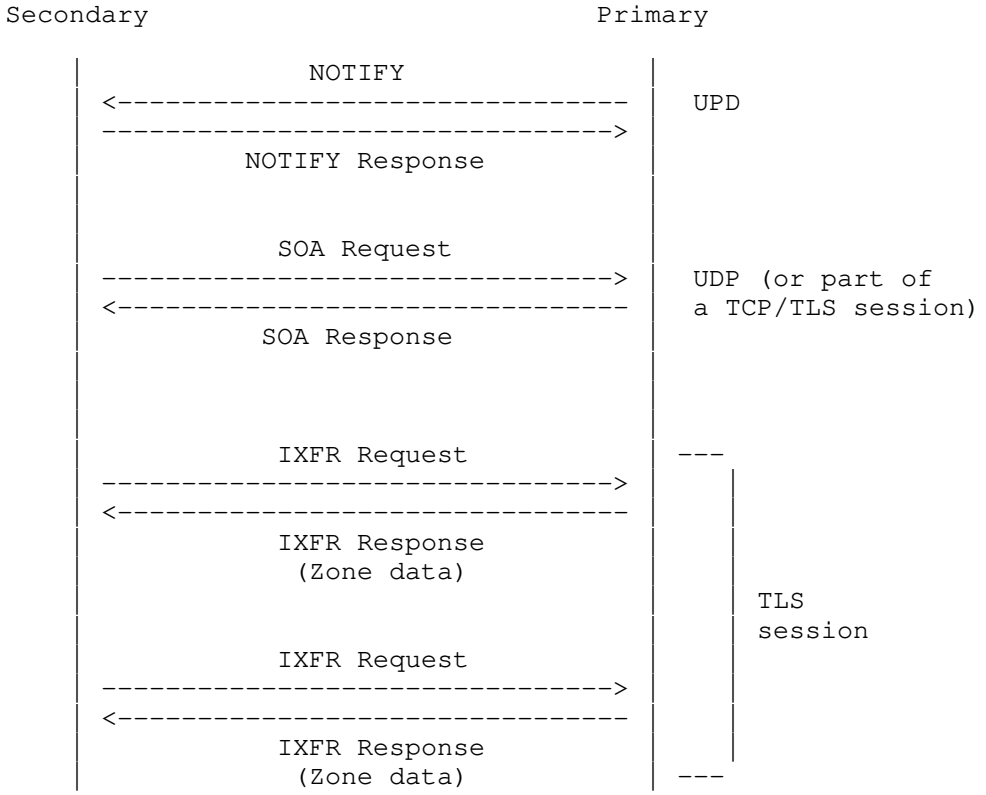

Figure 1. IXoT Mechanism

7.4. 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:
  - an IP based ACL (which can be either per-message or per-connection)
  - Mutual TLS (mTLS)

The server MAY also require a valid TSIG/SIG(0) signature, but this alone is not sufficient to authenticate the client or server.

Authentication mechanisms are discussed in full in Section 9 and the rationale for the above requirement in Section 10. Transfer group policies are discussed in Section 11.

7.5. XoT connections

The details in Section 6 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.

7.6. 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,S.S. 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 wish to support DoT for regular queries from recursive 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 7.7)

- o the operational and policy options that a XoT server operator has with regard to managing TLS connections and messages (see Appendix A)

7.7. 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 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.

[OPEN QUESTION] Should this instead be Prohibited (by policy), or should a new EDE be created for this case?

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.

7.8. AXoT specifics

7.8.1. Padding AXoT responses

The goal of padding AXoT responses would be two fold:

- o to obfuscate the actual size of the transferred zone to minimize information leakage about the entire contents of the zone.

- o 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 complicates any attempt to analyze the traffic size and timing to extract information.

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.

7.9. IXoT specifics

7.9.1. 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.

7.9.2. 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.

7.9.3. Padding of IXoT responses

The goal of padding IXoT responses would be 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 in size greatly 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 policies and the trade-offs involved are expected to be the subject of future work.

7.10. 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.

8. Multi-primary Configurations

Also known as multi-master 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 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 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.

9. Authentication mechanisms

To provide context to the requirements in section Section 7.4, 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 10 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).

9.1. TSIG

TSIG [RFC2845] 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

9.2. SIG(0)

SIG(0) [RFC2931] similarly also 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

9.3. TLS

9.3.1. 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. a MitM attack on the connection from client to server), and is not considered as useful for XoT.

Properties: None guaranteed.

9.3.2. 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 authentication (of the server).

9.3.3. 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 authentic 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 authentication.

9.4. 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.

9.5. ZONEMD

For completeness, we also describe Message Digest for DNS Zones (ZONEMD) [I-D.ietf-dnsop-dns-zone-digest] 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 [I-D.ietf-dnsop-dns-zone-digest] 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.

10. 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 7.4 addresses the issue of ensuring that the transfers is encrypted between the two endpoints directly involved in the current transfers. The following table summarized the properties of a selection of the mechanisms discussed in Section 9. 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: 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 authenticated

- Using Strict TLS and an IP based ACL on the primary also provides authentication of both end points

- Additional use of TSIG (or equally SIG(0)) can also provide data origin authentication which might be desirable for deployments that include a proxy between the secondary and primary, but is not part of the XoT requirement because it does nothing to guarantee channel confidentiality or authentication.

11. 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'.

Within any transfer group both AXFRs and IXFRs for a zone MUST all use the same policy, e.g., if AXFRs use AXoT all IXFRs MUST use IXoT.

In order to assure the confidentiality of the zone information, the entire transfer group MUST have a consistent policy of requiring confidentiality. If any do not, this is a weak link for attackers to exploit.

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 12).

A XoT policy should specify

- What kind of TLS is required (Strict or Mutual TLS)

- or if an IP based ACL is required.

- (optionally) if TSIG/SIG(0) is required

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.

12. 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.

Such configuration options MUST only be used in a 'migration mode' though and therefore should be used with care.

13. Implementation Status

The 1.9.2 version of Unbound [3] includes an option to perform AXoT (instead of AXFR-over-TCP). This requires the client (secondary) to authenticate the server (primary) using a configured authentication domain name.

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.

14. IANA Considerations

15. 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].

16. Acknowledgements

The authors thank Tony Finch, Peter van Dijk, Benno Overeinder, Shumon Huque and Tim Wicinski for review and discussions.

17. Contributors

Significant contributions to the document were made by:

Han Zhang
Salesforce
San Francisco, CA
United States

Email: hzhang@salesforce.com

18. Changelog

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

- Add Github repository
- Fix typos 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

19. References

19.1. Normative References

[I-D.vcelak-nsec5]
Vcelak, J., Goldberg, S., Papadopoulos, D., Huque, S., and D. Lawrence, "NSEC5, DNSSEC Authenticated Denial of Existence", draft-vcelak-nsec5-08 (work in progress), December 2018.

[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>.

[RFC2845] Vixie, P., Gudmundsson, O., Eastlake 3rd, D., and B. Wellington, "Secret Key Transaction Authentication for DNS (TSIG)", RFC 2845, DOI 10.17487/RFC2845, May 2000, <https://www.rfc-editor.org/info/rfc2845>.

[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>.

[RFC7626] Bortzmeyer, S., "DNS Privacy Considerations", RFC 7626, DOI 10.17487/RFC7626, August 2015, <https://www.rfc-editor.org/info/rfc7626>.

[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>.

19.2. Informative References

[I-D.ietf-dnsop-dns-zone-digest]
Wessels, D., Barber, P., Weinberg, M., Kumari, W., and W. Hardaker, "Message Digest for DNS Zones", draft-ietf-dnsop-dns-zone-digest-14 (work in progress), October 2020.

[I-D.ietf-dprive-dnsoquic]
Huitema, C., Mankin, A., and S. Dickinson, "Specification of DNS over Dedicated QUIC Connections", draft-ietf-dprive-dnsoquic-01 (work in progress), October 2020.

[I-D.ietf-dprive-phase2-requirements]
Livingood, J., Mayrhofer, A., and B. Overeinder, "DNS Privacy Requirements for Exchanges between Recursive Resolvers and Authoritative Servers", draft-ietf-dprive-phase2-requirements-02 (work in progress), November 2020.

[I-D.ietf-tls-esni]
Rescorla, E., Oku, K., Sullivan, N., and C. Wood, "TLS Encrypted Client Hello", draft-ietf-tls-esni-08 (work in progress), October 2020.

[I-D.vandijk-dprive-ds-dot-signal-and-pin]
Dijk, P., Geuze, R., and E. Bretelle, "Signalling Authoritative DoT support in DS records, with key pinning", draft-vandijk-dprive-ds-dot-signal-and-pin-01 (work in progress), July 2020.

[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>.

[RFC1982] Elz, R. and R. Bush, "Serial Number Arithmetic", RFC 1982, DOI 10.17487/RFC1982, August 1996, <https://www.rfc-editor.org/info/rfc1982>.

[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>.

[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>.

[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>.

19.3. URIs

[1] https://www.isc.org/bind/

[2] https://www.nlnetlabs.nl/projects/nsd/about/

[3] https://github.com/NLnetLabs/unbound/blob/release-1.9.2/doc/Changelog

Appendix A. XoT server connection handling

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 proxies having no natural way to forward the ALPN signal to DNS nameservers over TCP connections.

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. 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. TLS 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 connections and then apply a TLS specific response policy on a per query basis.

As an aside, whilst [RFC7766] makes a general purpose distinction to clients in the 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 below). Currently, none of the major open source DNS authoritative implementations offer an option for different response policies in different transports (but 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 1098 XH
The Netherlands

Email: willem@nlnetlabs.nl

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

Email: sara@sinodun.com

Shivan Sahib
Salesforce
Vancouver, BC
Canada

Email: ssahib@salesforce.com

Pallavi Aras
Salesforce
Herndon, VA
United States

Email: paras@salesforce.com

Allison Mankin
Salesforce
Herndon, VA
United States

Email: allison.mankin@gmail.com