Stream: Internet Engineering Task Force (IETF)
RFC: 9250
Category: Standards Track
Published: May 2022
ISSN: 2070-1721
Authors: C. Huitema, *Private Octopus Inc.*; S. Dickinson, *Sinodun IT*; A. Mankin, *Salesforce*

RFC 9250
DNS over Dedicated QUIC Connections

Abstract

This document describes the use of QUIC to provide transport confidentiality for DNS. The encryption provided by QUIC has similar properties to those provided by TLS, while QUIC transport eliminates the head-of-line blocking issues inherent with TCP and provides more efficient packet-loss recovery than UDP. DNS over QUIC (DoQ) has privacy properties similar to DNS over TLS (DoT) specified in RFC 7858, and latency characteristics similar to classic DNS over UDP. This specification describes the use of DoQ as a general-purpose transport for DNS and includes the use of DoQ for stub to recursive, recursive to authoritative, and zone transfer scenarios.

Status of This Memo

This is an Internet Standards Track document.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Further information on Internet Standards is available in Section 2 of RFC 7841.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at https://www.rfc-editor.org/info/rfc9250.

Copyright Notice

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 [RFC9000] [RFC9001]. DNS over QUIC is referred to here as DoQ, in line with "DNS Terminology" [DNS-TERMS].

The goals of the DoQ mapping are:

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

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

- the "stub to recursive resolver" scenario (also called the "stub to recursive" scenario in this document)
- the "recursive resolver to authoritative nameserver" scenario (also called the "recursive to authoritative" scenario in this document), and
- the "nameserver to nameserver" scenario (mainly used for zone transfers (XFR) [RFC1995] [RFC5936]).

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

The specific non-goals of this document are:

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

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

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

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**", "**NOT RECOMMENDED**", "**MAY**", and "**OPTIONAL**" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

3. Design Considerations

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

3.1. Provide DNS Privacy

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

QUIC connection setup includes the negotiation of security parameters using TLS, as specified in "Using TLS to Secure QUIC" [RFC9001], enabling encryption of the QUIC transport. Transmitting DNS messages over QUIC will provide essentially the same privacy protections as DoT [RFC7858] including Strict and Opportunistic usage profiles [RFC8310]. Further discussion on this is provided in Section 7.

3.2. Design for Minimum Latency

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

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

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

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

These considerations are reflected in the mapping of DNS traffic to QUIC streams in Section 4.2.

3.3. Middlebox Considerations

Using QUIC might allow a protocol to disguise its purpose from devices on the network path using encryption and traffic analysis resistance techniques like padding, traffic pacing, and traffic shaping. This specification does not include any measures that are designed to avoid such classification; the padding mechanisms defined in Section 5.4 are intended to obfuscate the specific records contained in DNS queries and responses, but not the fact that this is DNS traffic. Consequently, firewalls and other middleboxes might be able to distinguish DoQ from other protocols that use QUIC, like HTTP, and apply different treatment.

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

3.4. No Server-Initiated Transactions

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

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

4. Specifications

4.1. Connection Establishment

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

4.1.1. Port Selection

By default, a DNS server that supports DoQ **MUST** listen for and accept QUIC connections on the dedicated UDP port 853 (Section 8), unless there is a mutual agreement to use another port.

By default, a DNS client desiring to use DoQ with a particular server **MUST** establish a QUIC connection to UDP port 853 on the server, unless there is a mutual agreement to use another port.

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

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

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 [RFC9000], the QUIC transport specification.

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

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

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

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

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

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

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

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

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

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

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

4.2.1. DNS Message IDs

When sending queries over a QUIC connection, the DNS Message ID **MUST** be set to 0. The stream mapping for DoQ allows for unambiguous correlation of queries and responses, so the Message ID field is not required.

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

When forwarding a DNS message from DoQ over another transport, a DNS Message ID **MUST** be generated according to the rules of the protocol that is in use. When forwarding a DNS message from another transport over DoQ, the Message ID **MUST** be set to 0.

4.3. DoQ Error Codes

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

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

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

DOQ_PROTOCOL_ERROR (0x2): The DoQ implementation encountered a protocol error and is forcibly aborting the connection.

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

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

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

DOQ_ERROR_RESERVED (0xd098ea5e): An alternative error code used for tests.

See Section 8.4 for details on registering new error codes.

4.3.1. Transaction Cancellation

In QUIC, sending STOP_SENDING requests that a peer cease transmission on a stream. If a DoQ client wishes to cancel an outstanding request, it **MUST** issue a QUIC STOP_SENDING, and it **SHOULD** use the error code DOQ_REQUEST_CANCELLED. It **MAY** use a more specific error code registered according to Section 8.4. The STOP_SENDING request may be sent at any time but will have no effect if the server response has already been sent, in which case the client will simply discard the incoming response. The corresponding DNS transaction **MUST** be abandoned.

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

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

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

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

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

4.3.2. Transaction Errors

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

If a server is incapable of sending a DNS response due to an internal error, it **SHOULD** issue a QUIC RESET_STREAM frame. The error code **SHOULD** be set to DOQ_INTERNAL_ERROR. The corresponding DNS transaction **MUST** be abandoned. Clients **MAY** limit the number of unsolicited QUIC RESET_STREAM frames received on a connection before choosing to close the connection.

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

4.3.3. Protocol Errors

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

- a client or server receives a message with a non-zero Message ID
- a client or server receives a STREAM FIN before receiving all the bytes for a message indicated in the 2-octet length field
- a client receives a STREAM FIN before receiving all the expected responses
- a server receives more than one query on a stream
- a client receives a different number of responses on a stream than expected (e.g., multiple responses to a query for an A record)
- a client receives a STOP_SENDING request
- the client or server does not indicate the expected STREAM FIN after sending requests or responses (see Section 4.2)
- an implementation receives a message containing the edns-tcp-keepalive EDNS(0) Option [RFC7828] (see Section 5.5.2)
- a client or a server attempts to open a unidirectional QUIC stream
- a server attempts to open a server-initiated bidirectional QUIC stream
- a server receives a "replayable" transaction in 0-RTT data (for servers not willing to handle this case, see Section 4.5)

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

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

4.3.4. Alternative Error Codes

This specification describes specific error codes in Sections 4.3.1, 4.3.2, and 4.3.3. These error codes are meant to facilitate investigation of failures and other incidents. New error codes may be defined in future versions of DoQ or registered as specified in Section 8.4.

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

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

4.4. Connection Management

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

- idle timeout
- immediate close
- stateless reset

Clients and servers implementing DoQ **SHOULD** negotiate use of the idle timeout. Closing on idle timeout is done without any packet exchange, which minimizes protocol overhead. Per Section 10.1 of [RFC9000], the QUIC transport specification, the effective value of the idle timeout is computed as the minimum of the values advertised by the two endpoints. Practical considerations on setting the idle timeout are discussed in Section 5.5.2.

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

Clients **MAY** discard their connections to the server before the idle timeout expires. A client that has outstanding queries **SHOULD** close the connection explicitly using QUIC's CONNECTION_CLOSE mechanism and the DoQ error code DOQ_NO_ERROR.

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

4.5. Session Resumption and 0-RTT

A client **MAY** take advantage of the session resumption and 0-RTT mechanisms supported by QUIC transport [RFC9000] and QUIC TLS [RFC9001] if the server supports them. Clients **SHOULD** consider potential privacy issues associated with session resumption before deciding to use this mechanism and specifically evaluate the trade-offs presented in the various sections of this document. The privacy issues are detailed in Sections 7.1 and 7.2, and the implementation considerations are discussed in Section 5.5.3.

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

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

- Queue the offending transaction and only process it after the QUIC handshake has been completed, as defined in Section 4.1.1 of [RFC9001].
- Reply to the offending transaction with a response code REFUSED and an Extended DNS Error Code (EDE) "Too Early" using the extended RCODE mechanisms defined in [RFC6891] and the extended DNS errors defined in [RFC8914]; see Section 8.3.
- Close the connection with the error code DOQ_PROTOCOL_ERROR.

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 2-octet message length field in DNS over TCP [RFC1035] and DoT [RFC7858], and by the definition of the "application/dns-message" for DoH [RFC8484]. DoQ enforces the same restriction.

The Extension Mechanisms for DNS (EDNS(0)) [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 scenario, the authentication requirements are the same as described in DoT [RFC7858] and "Usage Profiles for DNS over TLS and DNS over DTLS" [RFC8310]. [RFC8932] states that DNS privacy services **SHOULD** provide credentials that clients can use to authenticate the server. Given this, and to align with the authentication model for DoH, DoQ stubs **SHOULD** use a Strict usage profile. Client authentication for the encrypted stub to recursive scenario is not described in any DNS RFC.

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

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

5.2. Fallback to Other Protocols on Connection Failure

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

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

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

5.3. Address Validation

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

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

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

5.4. Padding

Implementations **MUST** protect against the traffic analysis attacks described in Section 7.5 by the judicious injection of padding. This could be done either by padding individual DNS messages using the EDNS(0) Padding Option [RFC7830] or by padding QUIC packets (see Section 19.1 of [RFC9000]).

In theory, padding at the QUIC packet level could result in better performance for the equivalent protection, because the amount of padding can take into account non-DNS frames such as acknowledgements or flow control updates, and also because QUIC packets can carry multiple

DNS messages. However, applications can only control the amount of padding in QUIC packets if the implementation of QUIC exposes adequate APIs. This leads to the following recommendations:

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

Implementations might choose not to use a QUIC API for padding if it is significantly simpler to reuse existing DNS message padding logic that is applied to other encrypted transports.

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

5.5. Connection Handling

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

5.5.1. Connection Reuse

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

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

5.5.2. Resource Management

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

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

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

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

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

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

5.5.3. Using 0-RTT and Session Resumption

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

Session resumption and 0-RTT data transmission create privacy risks detailed in Sections 7.1 and 7.2. The following recommendations are meant to reduce the privacy risks while enjoying the performance benefits of 0-RTT data, subject to the restrictions specified in Section 4.5.

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

Clients could receive address validation tokens from the server using the NEW_TOKEN mechanism; see Section 8 of [RFC9000]. The associated tracking risks are mentioned in Section 7.3. Clients **SHOULD** only use the address validation tokens when they are also using session resumption thus avoiding additional tracking risks.

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

5.5.4. Controlling Connection Migration for Privacy

DoQ implementations might consider using the connection migration features defined in Section 9 of [RFC9000]. These features enable connections to continue operating as the client's connectivity changes. As detailed in Section 7.4, these features trade off privacy for latency. By default, clients **SHOULD** be configured to prioritize privacy and start new sessions if their connectivity changes.

5.6. Processing Queries in Parallel

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

5.7. Zone Transfer

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

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

5.8. Flow Control Mechanisms

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

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

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

6. Security Considerations

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

The security considerations of DoQ should be comparable to those of DoT [RFC7858]. DoT as specified in [RFC7858] only addresses the stub to recursive scenario, but the considerations about person-in-the-middle attacks, middleboxes, and caching of data from cleartext connections also apply for DoQ to the resolver to authoritative server scenario. As stated in Section 5.1, the authentication requirements for securing zone transfer using DoQ are the same as those for zone transfer over DoT; therefore, the general security considerations are entirely analogous to those described in [RFC9103].

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

7. Privacy Considerations

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

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

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

These issues are developed in Sections 7.1 and 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" [RFC9076]. The attack is partially mitigated by reducing the observability of this traffic. The mandatory replay protection mechanisms in TLS

1.3 [RFC8446] limit but do not eliminate the risk of replay. 0-RTT packets can only be replayed within a narrow window, which is only wide enough to account for variations in clock skew and network transmission.

The recommendation for TLS 1.3 [RFC8446] is that the capability to use 0-RTT data should be turned off by default and only enabled if the user clearly understands the associated risks. In the case of DoQ, allowing 0-RTT data provides significant performance gains, and there is a concern that a recommendation to not use it would simply be ignored. Instead, a set of practical recommendations is provided in Sections 4.5 and 5.5.3.

The specifications in Section 4.5 block the most obvious risks of replay attacks, as they only allow for transactions that will not change the long-term state of the server.

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

7.2. Privacy Issues with Session Resumption

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

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

The recommendations in Section 5.5.3 are designed to mitigate these risks. Using session tickets only once mitigates the risk of tracking by third parties. Refusing to resume a session if addresses change mitigates the incremental risk of tracking by the server (but the risk of tracking by IP address remains).

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

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

7.3. Privacy Issues with Address Validation Tokens

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

The recommendations in Section 5.5.3 mitigates this risk by tying the usage of the NEW_TOKEN to that of session resumption, though this recommendation does not cover the case where the client is unaware of the address change.

7.4. Privacy Issues with Long Duration Sessions

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

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

The recommendation in Section 5.5.4 mitigates the privacy concerns related to long duration sessions using multiple client addresses.

7.5. Traffic Analysis

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

Implementations **SHOULD** use the mechanisms defined in Section 5.4 to mitigate this attack.

8. IANA Considerations

8.1. Registration of a DoQ Identification String

This document creates a new registration for the identification of DoQ in the "TLS 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 a Dedicated Port

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

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

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

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

Service Name: domain-s

Port Number: 853

Transport Protocol(s): UDP

Assignee: IESG

Contact: IETF Chair

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

Reference: [RFC7858][RFC8094] This document

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

8.3. Reservation of an Extended DNS Error Code: Too Early

IANA has registered the following value in the "Extended DNS Error Codes" registry [RFC8914]:

INFO-CODE: 26

Purpose: Too Early

Reference: This document

8.4. DNS-over-QUIC Error Codes Registry

IANA has added a registry for "DNS-over-QUIC Error Codes" on the "Domain Name System (DNS) Parameters" web page.

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

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

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

Registrations in this registry **MUST** include the following fields:

Value: The assigned codepoint

Status: "Permanent" or "Provisional"

Contact: Contact details for the registrant

In addition, permanent registrations **MUST** include:

Error: A short mnemonic for the parameter

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

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

Provisional registrations of codepoints are intended to allow for private use and experimentation with extensions to DoQ. However, provisional registrations could be reclaimed and reassigned for other purposes. In addition to the parameters listed above, provisional registrations **MUST** include:

Date: The date of last update to the registration

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

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

Status: Permanent

Contact: DPRIVE WG

Specification: Section 4.3

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

Table 1: Initial DNS-over-QUIC Error Codes Entries

9. References

9.1. Normative References

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

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

[RFC1995] Ohta, M., "Incremental Zone Transfer in DNS", RFC 1995, DOI 10.17487/RFC1995, August 1996, <https://www.rfc-editor.org/info/rfc1995>.

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

[RFC5936] Lewis, E. and A. Hoenes, Ed., "DNS Zone Transfer Protocol (AXFR)", RFC 5936, DOI 10.17487/RFC5936, June 2010, <https://www.rfc-editor.org/info/rfc5936>.

[RFC6891] Damas, J., Graff, M., and P. Vixie, "Extension Mechanisms for DNS (EDNS(0))", STD 75, RFC 6891, DOI 10.17487/RFC6891, April 2013, <https://www.rfc-editor.org/info/rfc6891>.

[RFC7301] Friedl, S., Popov, A., Langley, A., and E. Stephan, "Transport Layer Security (TLS) Application-Layer Protocol Negotiation Extension", RFC 7301, DOI 10.17487/RFC7301, July 2014, <https://www.rfc-editor.org/info/rfc7301>.

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

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

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

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

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[RFC8310] Dickinson, S., Gillmor, D., and T. Reddy, "Usage Profiles for DNS over TLS and DNS over DTLS", RFC 8310, DOI 10.17487/RFC8310, March 2018, <https://www.rfc-editor.org/info/rfc8310>.

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

[RFC8467] Mayrhofer, A., "Padding Policies for Extension Mechanisms for DNS (EDNS(0))", RFC 8467, DOI 10.17487/RFC8467, October 2018, <https://www.rfc-editor.org/info/rfc8467>.

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

[RFC9000] Iyengar, J., Ed. and M. Thomson, Ed., "QUIC: A UDP-Based Multiplexed and Secure Transport", RFC 9000, DOI 10.17487/RFC9000, May 2021, <https://www.rfc-editor.org/info/rfc9000>.

[RFC9001] Thomson, M., Ed. and S. Turner, Ed., "Using TLS to Secure QUIC", RFC 9001, DOI 10.17487/RFC9001, May 2021, <https://www.rfc-editor.org/info/rfc9001>.

[RFC9103] Toorop, W., Dickinson, S., Sahib, S., Aras, P., and A. Mankin, "DNS Zone Transfer over TLS", RFC 9103, DOI 10.17487/RFC9103, August 2021, <https://www.rfc-editor.org/info/rfc9103>.

9.2. Informative References

[BCP195] Sheffer, Y., Holz, R., and P. Saint-Andre, "Recommendations for Secure Use of Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS)", BCP 195, RFC 7525, May 2015.

Moriarty, K. and S. Farrell, "Deprecating TLS 1.0 and TLS 1.1", BCP 195, RFC 8996, March 2021.

<https://www.rfc-editor.org/info/bcp195>

[DNS-TERMS] Hoffman, P. and K. Fujiwara, "DNS Terminology", Work in Progress, Internet-Draft, draft-ietf-dnsop-rfc8499bis-03, 28 September 2021, <https://datatracker.ietf.org/doc/html/draft-ietf-dnsop-rfc8499bis-03>.

[DNS0RTT] Kahn Gillmor, D., "DNS + 0-RTT", Message to DNS-Privacy WG mailing list, 6 April 2016, <https://www.ietf.org/mail-archive/web/dns-privacy/current/msg01276.html>.

[GREASING-QUIC] Thomson, M., "Greasing the QUIC Bit", Work in Progress, Internet-Draft, draft-ietf-quic-bit-grease-02, 10 November 2021, <https://datatracker.ietf.org/doc/html/draft-ietf-quic-bit-grease-02>.

[HTTP/3] Bishop, M., Ed., "Hypertext Transfer Protocol Version 3 (HTTP/3)", Work in Progress, Internet-Draft, draft-ietf-quic-http-34, 2 February 2021, <https://datatracker.ietf.org/doc/html/draft-ietf-quic-http-34>.

[RFC1996] Vixie, P., "A Mechanism for Prompt Notification of Zone Changes (DNS NOTIFY)", RFC 1996, DOI 10.17487/RFC1996, August 1996, <https://www.rfc-editor.org/info/rfc1996>.

[RFC3833] Atkins, D. and R. Austein, "Threat Analysis of the Domain Name System (DNS)", RFC 3833, DOI 10.17487/RFC3833, August 2004, <https://www.rfc-editor.org/info/rfc3833>.

[RFC6335] Cotton, M., Eggert, L., Touch, J., Westerlund, M., and S. Cheshire, "Internet Assigned Numbers Authority (IANA) Procedures for the Management of the Service Name and Transport Protocol Port Number Registry", BCP 165, RFC 6335, DOI 10.17487/RFC6335, August 2011, <https://www.rfc-editor.org/info/rfc6335>.

[RFC7828] Wouters, P., Abley, J., Dickinson, S., and R. Bellis, "The edns-tcp-keepalive EDNS0 Option", RFC 7828, DOI 10.17487/RFC7828, April 2016, <https://www.rfc-editor.org/info/rfc7828>.

[RFC7873] Eastlake 3rd, D. and M. Andrews, "Domain Name System (DNS) Cookies", RFC 7873, DOI 10.17487/RFC7873, May 2016, <https://www.rfc-editor.org/info/rfc7873>.

[RFC8094] Reddy, T., Wing, D., and P. Patil, "DNS over Datagram Transport Layer Security (DTLS)", RFC 8094, DOI 10.17487/RFC8094, February 2017, <https://www.rfc-editor.org/info/rfc8094>.

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

[RFC8490] Bellis, R., Cheshire, S., Dickinson, J., Dickinson, S., Lemon, T., and T. Pusateri, "DNS Stateful Operations", RFC 8490, DOI 10.17487/RFC8490, March 2019, <https://www.rfc-editor.org/info/rfc8490>.

[RFC8932] Dickinson, S., Overeinder, B., van Rijswijk-Deij, R., and A. Mankin, "Recommendations for DNS Privacy Service Operators", BCP 232, RFC 8932, DOI 10.17487/RFC8932, October 2020, <https://www.rfc-editor.org/info/rfc8932>.

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

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

Appendix A. The NOTIFY Service

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

Section 4.5 says "The 0-RTT mechanism **MUST NOT** be used to send DNS requests that are not "replayable" transactions". This specification supports sending a NOTIFY in 0-RTT data because although a NOTIFY technically changes the state of the receiving server, the effect of replaying NOTIFYs has negligible impact in practice.

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

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

Acknowledgements

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

The privacy issue with 0-RTT data and session resumption was 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 draft version of this document, and to Robert Evans for the discussion of 0-RTT privacy issues. Early reviews by Paul Hoffman and Martin Thomson and interoperability tests conducted by Stephane Bortzmeyer helped improve the definition of the protocol.

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

Authors' Addresses

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

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

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