Internet Engineering Task Force (IETF)
Request for Comments: 9103
Updates: 1995, 5936, 7766
Category: Standards Track
ISSN: 2070-1721

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

August 2021

DNS Zone Transfer over TLS

Abstract

DNS zone transfers are transmitted in cleartext, which gives attackers the opportunity to collect the content of a zone by eavesdropping on network connections. The DNS Transaction Signature (TSIG) mechanism is specified to restrict direct zone transfer to authorized clients only, but it does not add confidentiality. This document specifies the use of TLS, rather than cleartext, to prevent zone content collection via passive monitoring of zone transfers: XFR over TLS (XoT). Additionally, this specification updates RFC 1995 and RFC 5936 with respect to efficient use of TCP connections and RFC 7766 with respect to the recommended number of connections between a client and server for each transport.

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/rfc9103.

Table of Contents

1. Introduction

DNS has a number of privacy vulnerabilities, as discussed in detail in [RFC9076]. Query privacy between stub resolvers and recursive

resolvers 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 [DPRIVE-DNSOQUIC]. There is ongoing work on DNS privacy requirements for exchanges between recursive resolvers and authoritative servers and some suggestions for how signaling of DoT support by authoritative name servers might work. However, there is currently no RFC that specifically defines recursive-to-authoritative DNS over TLS (ADoT).

[RFC9076] establishes that a stub resolver's DNS query transactions are not public and that they need 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 [RFC5155] and [RFC5936].

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

* Person-laptop.example.org

* MX-for-Location.example.org

* Service-tenant-from-another-org.example.org

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

Neither of the RFCs mentioned in [RFC9076] contemplate 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 that use NSEC, i.e., queries for the authenticated denial-of-existence records allow a client to walk through the entire zone contents. [RFC5155] specifies NSEC3, a mechanism to provide measures against zone enumeration for DNSSEC-signed zones (a goal was to make it as hard to enumerate a DNSSEC-signed zone as an unsigned zone). Whilst this is widely used, it has been demonstrated that zone walking is possible for precomputed NSEC3 using attacks, such as those described in [NSEC3-attacks]. This prompted further work on an alternative mechanism for DNSSEC-authenticated denial of existence (NSEC5 [NSEC5]); however, questions remain over the practicality of this mechanism.

[RFC5155] does not address data obtained outside zone enumeration (nor does [NSEC5]). Preventing eavesdropping of zone transfers (as described in this document) is orthogonal to preventing zone enumeration, though they aim to protect the same information.

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

Section 8 of the NIST document "Secure Domain Name System (DNS) Deployment Guide" [NIST-GUIDE] discusses restricting access for zone transfers using Access Control Lists (ACLs) and TSIG in more detail. It also discusses the possibility that specific deployments might choose to use a lower-level network layer to protect zone transfers, e.g., IPsec.

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

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

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

* [RFC1995] (IXFR) and [RFC5936] (AXFR) are both updated to add further specification on efficient use of TCP connections.

* Section 6.2.2 of [RFC7766] ("DNS Transport over TCP - Implementation Requirements") is updated with a new recommendation about the number of connections between a client and server for each transport.

2. Terminology

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

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

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

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

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

XoT: XFR-over-TLS mechanisms, as specified in this document, which apply to both AXFR over TLS and IXFR over TLS (XoT is pronounced ’zot’ since X here stands for ’zone transfer’)

AXoT: AXFR over TLS

IXoT: IXFR over TLS

3. 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 name servers hosting a zone (including both those acting as hidden primaries/secondaries or directly serving the zone) as sensitive information. The proposed mechanism does not attempt to obscure such information. The reasons for this include:

* much of this information can be obtained by various methods, including active scanning of the DNS, and

* an attacker who can monitor network traffic can rather easily infer relations between name servers simply from traffic patterns, even when some or all of the traffic is encrypted (in terms of current deployments).

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

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

4. Design Considerations for XoT

The following principles were considered in the design for XoT:

Confidentiality: Clearly using an encrypted transport for zone transfers will defeat zone content leakage that can occur via passive surveillance.

Authentication: Use of single or mutual TLS (mTLS) authentication (in combination with ACLs) can complement and potentially be an alternative to TSIG.

Performance:

* Existing AXFR and IXFR mechanisms have the burden of backwards compatibility with older implementations based on the original specifications in [RFC1034] and [RFC1035]. For example, some older AXFR servers don’t support using a TCP connection for multiple AXFR sessions or XFRs of different zones because they have not been updated to follow the guidance in [RFC5936]. Any implementation of XoT would obviously be required to implement optimized and interoperable transfers, as described in [RFC5936], e.g., transfer of multiple zones over one connection.

* Current usage of TCP for IXFR is suboptimal in some cases, i.e., connections are frequently closed after a single IXFR.

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 query for the SOA (start of zone authority) record (based on the refresh timer) to determine if an AXFR was required.

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

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

In this document, the term 'XFR mechanism' is used to describe the entire set of message exchanges between a secondary and a primary that concludes with a successful AXFR or IXFR request/response. This set may or may not include:

* NOTIFY messages

* SOA queries

* Fallback from IXFR to AXFR

* Fallback from IXFR over UDP to IXFR over TCP

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

5.1. AXFR Mechanism

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

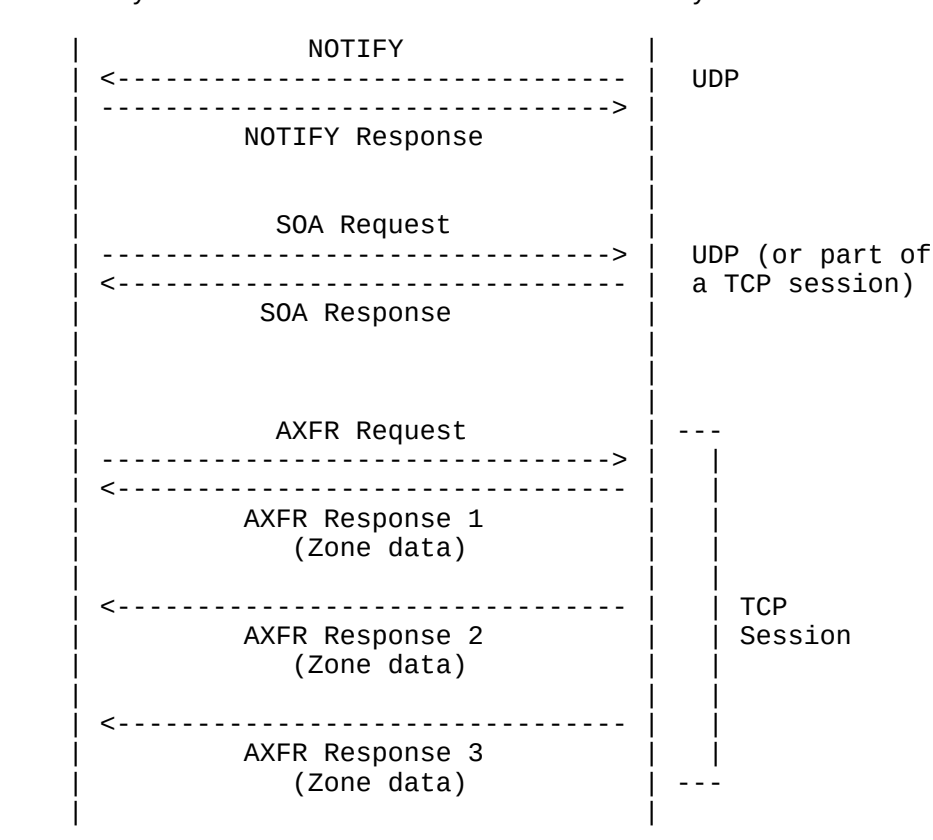


Figure 1: AXFR Mechanism

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 an SOA query to the primary to obtain the serial number of the zone held by the primary.

3. If the primary serial is higher than the secondary's serial (using Serial Number Arithmetic [RFC1982]), the secondary makes an AXFR request (over TCP) to the primary, after which the AXFR

data flows in one or more AXFR responses on the TCP connection. [RFC5936] defines this specific step as an 'AXFR session', i.e., as an AXFR query message and the sequence of AXFR response messages returned for it.

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

Additionally, Sections 4.1, 4.1.1, and 4.1.2 of [RFC5936] provide improved guidance for AXFR clients and servers with regard to reuse 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.

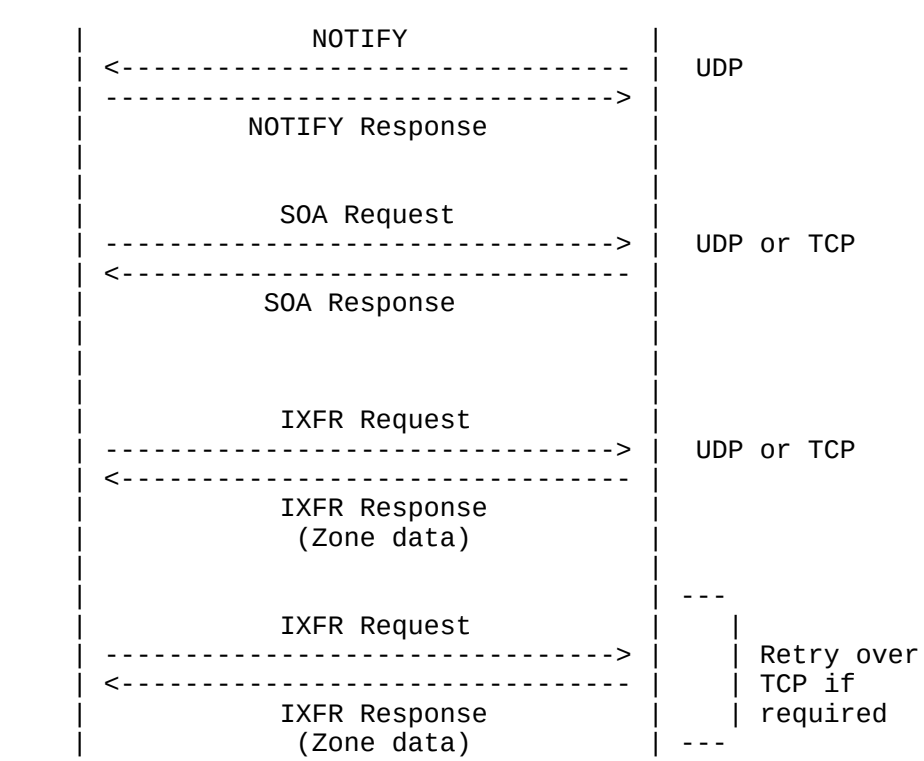


Figure 2: IXFR Mechanism

1. An IXFR is normally (but not always) preceded by a NOTIFY (over UDP) from the primary to the secondary. A secondary may also initiate an IXFR based on a refresh timer or scheduled/triggered zone maintenance.

2. The secondary will normally (but not always) make an SOA query to the primary to obtain the serial number of the zone held by the primary.

3. If the primary serial is higher than the secondary's serial (using Serial Number Arithmetic [RFC1982]), the secondary makes an IXFR request to the primary, after which the primary sends an

IXFR response.

[RFC1995] specifies that IXFR 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 an additional step where the secondary must fall back to AXFR because, e.g., the primary does not support IXFR.

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

5.3. Data Leakage of NOTIFY and SOA Message Exchanges

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

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

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 query type (QTYPE) for NOTIFY defined at the time of this writing is SOA, this does not pose a potential leak.

5.3.2. SOA

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

6. Updates to Existing Specifications

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

Both [RFC1995] and [RFC5936] were published sometime before TCP became a widely supported transport for DNS. [RFC1995], in fact, says nothing with respect to optimizing IXFRs over TCP or reusing already open TCP connections to perform IXFRs or other queries. Therefore, there arguably is an implicit assumption that a TCP connection is used for one and only one IXFR request. Indeed, many major open-source implementations take this approach (at the time of this writing). And whilst [RFC5936] gives guidance on connection reuse for AXFR, it predates 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 and

* remove any need for XoT implementations to support legacy behavior for XoT connections that XFR-over-TCP implementations have historically often supported.

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

* Implementations MUST use [RFC7766] ("DNS Transport over TCP - Implementation Requirements") to optimize the use of TCP connections.

* Whilst [RFC7766] states that "DNS clients SHOULD pipeline their queries" on TCP connections, it did not distinguish between XFRs and other queries for this behavior. It is now recognized that XFRs are not as latency sensitive as other queries and can be significantly more complex for clients to handle, both because of the large amount of state that must be kept and because there may be multiple messages in the responses. For these reasons, it is clarified here that a valid reason for not pipelining queries is when they are all XFR queries, i.e., clients sending multiple XFRs MAY choose not to pipeline those queries. Clients that do not pipeline XFR queries therefore have no additional requirements to handle out-of-order or intermingled responses (as described later), since they will never receive them.

* Implementations SHOULD use the edns-tcp-keepalive EDNS(0) option [RFC7828] to manage persistent connections. This is more flexible than the alternative of simply using fixed timeouts.

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

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

6.1. Update to RFC 1995 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 RFC 5936 for AXFR over TCP

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

intermingled, and AXFR-over-TCP clients compliant with this specification, which pipeline AXFR requests, MUST be able to handle this.

6.3. Updates to RFCs 1995 and 5936 for XFR over TCP

6.3.1. Connection Reuse

As specified, XFR-over-TCP clients SHOULD reuse 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 reusing existing connections might include:

* As already noted in [RFC7766], separate connections for different zones might be preferred for operational reasons. In this case, the number of concurrent connections for zone transfers SHOULD be limited to the total number of zones transferred between the client and server.

* A configured limit for the number of outstanding queries or XFR requests allowed on a single TCP connection has been reached.

* 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 EDNS(0) 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 statement:

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

In this document, the above is clarified to indicate that implementations capable of both AXFR and IXFR and compliant with this specification SHOULD:

* use the same TCP connection for both AXFR and IXFR requests to the same primary,

* pipeline such requests (if they pipeline XFR requests in general) and MAY intermingle them, and

* send the response(s) for each request as soon as they are available, i.e., responses MAY be sent intermingled.

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

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. Servers SHOULD return SERVFAIL if this limit is hit, since it is a transient error and a retry at a later time might succeed (there is no previous specification for this behavior).

6.3.4. The edns-tcp-keepalive EDNS(0) Option

XFR clients that send the edns-tcp-keepalive EDNS(0) 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 EDNS(0) [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.

In this document, the above is clarified to indicate 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 EDNS(0) 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 provisions that implementations may want to offer options to fallback to legacy behavior when interoperating with servers known to not support [RFC5936]. For purposes of interoperability, IXFR and AXFR implementations may want to continue offering such configuration options, as well as supporting some behaviors that were underspecified prior to this work (e.g., performing IXFR and AXFRs on separate connections). However, XoT connections should have no need to do so.

6.4. Update to RFC 7766

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

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

Whilst this recommends a particular behavior for the clients using TCP, it does not relax the requirement for servers to handle 'mixed'

traffic (regular queries and zone transfers) on any open TCP/TLS connection. It also overlooks the potential that other transports might want to take the same approach with regard to using separate connections for different purposes.

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

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

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

* one TCP connection for regular queries

* one TCP connection for zone transfers

* one TLS connection for regular queries

* one TLS connection for zone transfers

* one DoH connection for regular queries

* one DoH connection for zone transfers

Section 6.3.1 provides specific details of the reasons why more than one connection for a given transport might be required for zone transfers from a particular client.

7. XoT Specification

7.1. Connection Establishment

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

7.2. TLS Versions

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

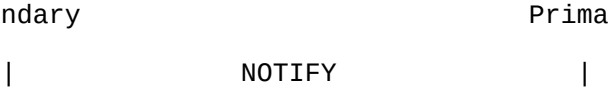
7.3. Port Selection

The connection for XoT SHOULD be established using port 853, as specified in [RFC7858], unless there is mutual agreement between the primary and secondary 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.

7.4. High-Level XoT Descriptions

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

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



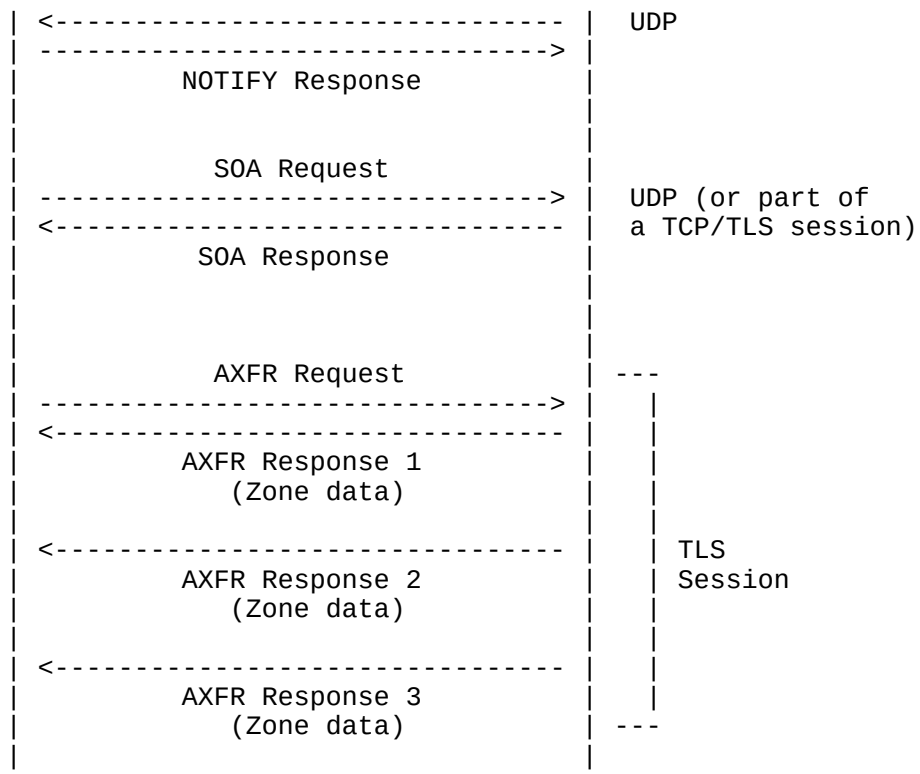


Figure 3: AXoT Mechanism

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

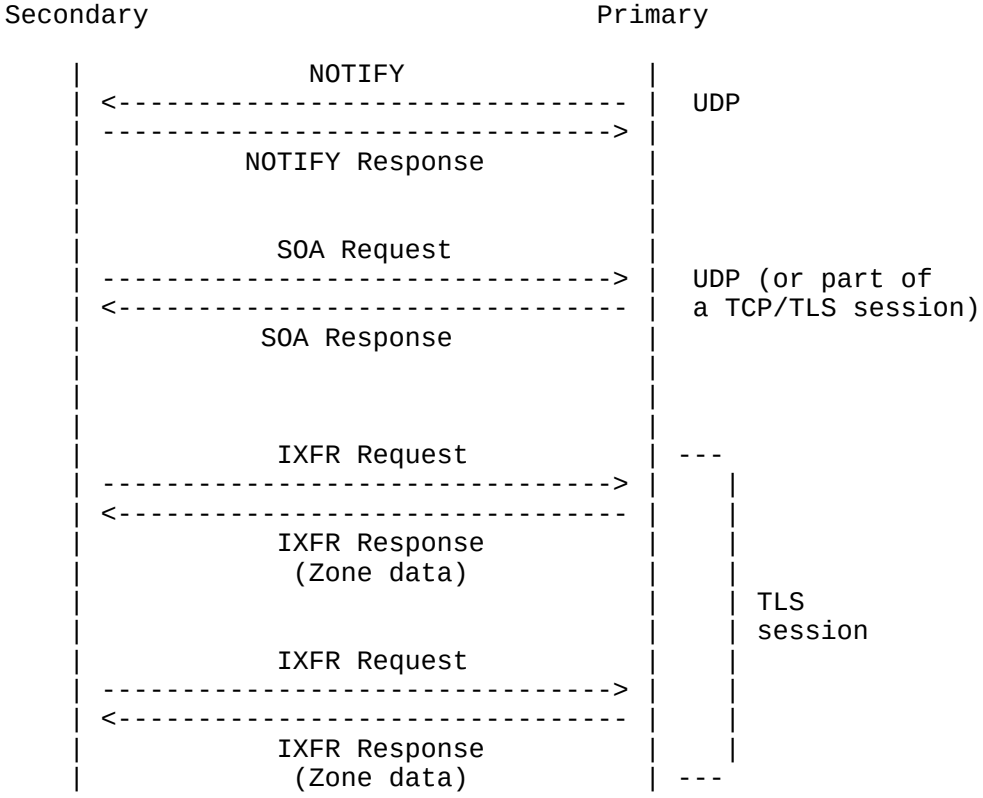


Figure 4: IXoT Mechanism

7.5. XoT Transfers

For a zone transfer between two endpoints to be considered protected with XoT, all XFR requests and responses for that zone MUST be sent

over TLS connections, where at a minimum:

* The client MUST authenticate the server by use of an authentication domain name using a Strict Privacy profile, as described in [RFC8310].

* The server MUST validate the client is authorized to request or proxy a zone transfer by using one or both of the following methods:

  - mutual TLS (mTLS)

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

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

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

7.6. 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, TCP, or TLS.

7.7. XoT vs. ADoT

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

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

[RFC5936] clearly states that non-AXFR session traffic can use an open connection; however, this requirement needs to be reevaluated when considering the application of 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:

* the utilization of EDE codes by XoT servers in response to queries on TLS connections that they are not willing to answer (see Section 7.8)

* the operational and policy options that an operator of a XoT

server has with regard to managing TLS connections and messages (see Appendix A)

7.8. Response RCODES

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

Historically, servers have used the REFUSED RCODE for many situations; therefore, 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 to 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.

7.9. AXoT Specifics

7.9.1. Padding AXoT Responses

The goal of padding AXoT responses is two fold:

* to obfuscate the actual size of the transferred zone to minimize information leakage about the entire contents of the zone

* to obfuscate the incremental changes to the zone between SOA updates to minimize information leakage about zone update activity and growth

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

It is noted here that, depending on the padding policies eventually developed for XoT, the requirement to obfuscate the total zone size might require a server to create ’empty’ AXoT responses, that is, AXoT responses that contain no RRs 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, for future proofing, and in order to guarantee support for future padding policies, it is stated 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.

Recommendations 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.

7.10. IXoT Specifics

7.10.1. Condensation of Responses

[RFC1995] says that 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.

7.10.2. Fallback to AXFR

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

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

7.10.3. Padding of IXoT Responses

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

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

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

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

7.11. Name Compression and Maximum Payload Sizes

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

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

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

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

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

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

9. Authentication Mechanisms

To provide context to the requirements in Section 7.5, this section provides a brief summary of some of the existing authentication and validation mechanisms (both transport independent and TLS specific) that are available when performing zone transfers. Section 10 then discusses in more detail specifically how a combination of TLS authentication, TSIG, and IP-based ACLs interact for XoT.

In this document, the mechanisms are classified based on the following properties:

Data Origin Authentication (DO):
: Authentication 1) of the fact that the DNS message originated from the party with whom credentials were shared and 2) of the data integrity of the message contents (the originating party may or may not be the 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 endpoints of a TCP/TLS connection) from passive surveillance.

Channel Authentication (CA):

Authentication of the identity of the 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 [RFC8945] provides a mechanism for two or more parties to use shared secret keys that 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 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:

* if clients know authentication information for a server, they SHOULD try to authenticate the server,

* if this fails or clients do not know the information, they MAY fallback to using TLS without authentication, or

* clients MAY fallback to using cleartext if TLS is not available.

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

Properties: None guaranteed.

9.3.2. Strict TLS

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

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

9.3.3. Mutual TLS

This is an extension to Strict TLS [RFC8310] that requires that a client is configured with an authentication domain name (and/or SPKI pin set) and a client certificate. The client offers the certificate for authentication by the server, and the client can authenticate the server the same way as in Strict TLS. This provides a defense for

both parties against active surveillance, providing bidirectional authentication and end-to-end channel confidentiality.

Properties: Channel confidentiality and mutual channel authentication.

9.4. IP-Based ACL on the Primary

Most DNS server implementations offer an option to configure an IP-based 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, ZONEMD [RFC8976] ("Message Digest for DNS Zones") is described here. The ZONEMD message digest is a mechanism that can be used to verify the content of a standalone zone. It is designed to be independent of the transmission channel or mechanism, allowing a general consumer of a zone to do origin authentication of the entire zone contents. Note that the current version of [RFC8976] states:

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

It is complementary but orthogonal to 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 that 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 7.5 addresses the issue of ensuring that the transfers are encrypted between the two endpoints directly involved in the current transfers. The following table summarizes the properties of a selection of the mechanisms discussed in Section 9. The two-letter abbreviations for the properties are used below: (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 endpoints are cryptographically authenticated.

* Using secondary-side Strict TLS with a primary-side IP-based ACL and TSIG/SIG(0) combination provides sufficient protection to be acceptable.

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

11. Policies for Both AXoT and IXoT

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

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

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

An individual zone transfer is not considered protected by XoT unless both the client and server are configured to use only XoT, and the overall zone transfer is not considered protected until all members of the transfer group are configured to use only XoT with all other transfers servers (see Section 12).

A XoT policy MUST specify if:

* mutual TLS is used and/or

* an IP-based ACL and TSIG/SIG(0) combination is used.

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

Certain aspects of the policies can be relatively easy to test 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 coordinating 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 to the multi-primary case where the primaries are migrating to XoT.

13. Operational Considerations

If the options described in Section 12 are available, such configuration options MUST only be used in a 'migration mode' and therefore should be used with great care.

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

14. IANA Considerations

This document has no IANA actions.

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 cleartext 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, or

* the risk of zone contents being obtained via zone enumeration techniques.

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

16. References

16.1. Normative References

[DoT-ALPN] IANA, "TLS Application-Layer Protocol Negotiation (ALPN) Protocol IDs", <https://www.iana.org/assignments/tls-extensiontype-values/>.

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

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

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

[RFC1996] Vixie, P., "A Mechanism for Prompt Notification of Zone

Changes (DNS NOTIFY)", RFC 1996, DOI 10.17487/RFC1996, August 1996, <https://www.rfc-editor.org/info/rfc1996>.

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

[RFC2931] Eastlake 3rd, D., "DNS Request and Transaction Signatures (SIG(0)s)", RFC 2931, DOI 10.17487/RFC2931, September 2000, <https://www.rfc-editor.org/info/rfc2931>.

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

[RFC6973] Cooper, A., Tschofenig, H., Aboba, B., Peterson, J., Morris, J., Hansen, M., and R. Smith, "Privacy Considerations for Internet Protocols", RFC 6973, DOI 10.17487/RFC6973, July 2013, <https://www.rfc-editor.org/info/rfc6973>.

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

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

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

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

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

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

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

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

[RFC8945] Dupont, F., Morris, S., Vixie, P., Eastlake 3rd, D., Gudmundsson, O., and B. Wellington, "Secret Key Transaction Authentication for DNS (TSIG)", STD 93, RFC 8945, DOI 10.17487/RFC8945, November 2020, <https://www.rfc-editor.org/info/rfc8945>.

16.2. Informative References

[BIND] ISC, "BIND 9.16.16", <https://www.isc.org/bind/>.

[DPRIVE-DNSOQUIC]
Huitema, C., Dickinson, S., and A. Mankin, "Specification of DNS over Dedicated QUIC Connections", Work in Progress, Internet-Draft, draft-ietf-dprive-dnsoquic-03, 12 July 2021, <https://datatracker.ietf.org/doc/html/draft-ietf-dprive-dnsoquic-03>.

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

[NSD] NLnet Labs, "NSD 4.3.6", <https://www.nlnetlabs.nl/projects/nsd/about/>.

[NSEC3-attacks]
Goldberg, S., Naor, N., Papadopoulos, D., Reyzin, L., Vasant, S., and A. Ziv, "Stretching NSEC3 to the Limit: Efficient Zone Enumeration Attacks on NSEC3 Variants", February 2015, <https://www.cs.bu.edu/~goldbe/papers/nsec3attacks.pdf>.

[NSEC5] Vcelak, J., Goldberg, S., Papadopoulos, D., Huque, S., and D. Lawrence, "NSEC5, DNSSEC Authenticated Denial of Existence", Work in Progress, Internet-Draft, draft-vcelak-nsec5-08, 29 December 2018, <https://datatracker.ietf.org/doc/html/draft-vcelak-nsec5-08>.

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

[RFC5155] Laurie, B., Sisson, G., Arends, R., and D. Blacka, "DNS Security (DNSSEC) Hashed Authenticated Denial of Existence", RFC 5155, DOI 10.17487/RFC5155, March 2008, <https://www.rfc-editor.org/info/rfc5155>.

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

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

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

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

[TLS-ESNI] Rescorla, E., Oku, K., Sullivan, N., and C. A. Wood, "TLS Encrypted Client Hello", Work in Progress, Internet-Draft, draft-ietf-tls-esni-13, 12 August 2021, <https://datatracker.ietf.org/doc/html/draft-ietf-tls-esni-13>.

Appendix A. XoT Server Connection Handling

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

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

A.1. Listening Only on a Specific IP Address for TLS

Obviously, a name server that 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 XoT to listen for TLS, 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 either using a proxy or directly in the 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 implementations of a DNS authoritative server support such an option.

A.3. SNI-Based TLS Acceptance

Primaries could also choose to only accept TLS connections based on a Server Name Indication (SNI) that was published only to their secondaries.

Pros: Reduces the number of accepted connections.

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

A.4. Transport-Specific Response Policies

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

As an aside, whilst [RFC7766] makes a general purpose distinction in the advice to clients about their usage of connections (between regular queries and zone transfers), this is not strict, and nothing

in the DNS protocol prevents using the same connection for both types of traffic. Hence, a server cannot know the intention of any client that connects to it; it can only inspect the messages it receives on such a connection and make per-query decisions about whether or not to answer those queries.

Example policies a XoT server might implement are:

strict: REFUSE all queries on TLS connections, except SOA and authorized XFR requests

moderate: REFUSE all queries on TLS connections until one is received that is signed by a recognized TSIG/SIG(0) key, then answer all queries on the connection after that

complex: apply a heuristic to determine which queries on a TLS connections to REFUSE

relaxed: answer all non-XoT queries on all TLS connections with the same policy applied to TCP queries

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

Cons: The server must handle the burden of accepting all TLS connections just to perform XFRs with a small number of secondaries. Client behavior to a REFUSED response is not clearly defined (see Section 7.8). Currently, none of the major open-source implementations of a DNS authoritative server offer an option for different response policies in different transports (but such functionality could potentially be implemented using a proxy).

A.4.1. SNI-Based Response Policies

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

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

Cons: As above.

Acknowledgements

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

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

Contributors

Significant contributions to the document were made by:

Han Zhang
Salesforce
San Francisco, CA
United States of America

Email: hzhang@salesforce.com

Authors’ Addresses

Willem Toorop
NLnet Labs
Science Park 400
1098 XH Amsterdam
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
Brave Software
Vancouver BC
Canada

Email: shivankaulsahib@gmail.com

Pallavi Aras
Salesforce
Herndon, VA
United States of America

Email: paras@salesforce.com

Allison Mankin
Salesforce
Herndon, VA
United States of America

Email: allison.mankin@gmail.com