dprive
Internet-Draft
Intended status: Standards Track
Expires: January 9, 2020

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

July 8, 2019

# DNS Zone Transfer using DNS Stateful Operations
# draft-zatda-dprive-xfr-using-dso-00

## Abstract

DNS zone transfers are transmitted in clear text, which gives attackers the opportunity to collect the content of a zone by eavesdropping on network connections. This document specifies use of DNS Stateful Operations to enable a subscribe/publish mechanism for zone transfers reducing the over head introduced by NOTITY/SOA interactions prior to zone transfer request. This additionally prevents zone contents collection via passive monitoring of zone transfers by restricting XFR using DSO to require TLS.

## Status of This Memo

## Copyright Notice

Table of Contents

## 1. Introduction

[I-D.hzpa-dprive-xfr-over-tls] enumerates the existing issues with clear text XFR mechanisms, outlines some use cases for using encrypted channels for zone transfer and also describes using TLS for zone transfers. It additionally discusses the various authentication

mechanisms that can be used to provide data and channel authentication, and channel confidentiality.

This draft describes the use of a DSO [RFC8490] based protocol to perform zone transfers.  This mechanism is heavily based on an existing use of DSO where DNS clients can subscribe to receive asynchronous notifications of changes to RRSets of interest: DNS PUSH Notifications [I-D.ietf-dnssd-push].  That specification was developed with DNS Service Discovery in mind, this document describes an analogous protocol (XFR-using-DSO) where DNS clients can subscribe to receive asynchronous notifications of changes to zones of interest, it is developed with efficient and confidential zone transfers between primaries and secondaries in mind.

In the XFR-using-DSO model, a DSO connection is first opened between the client and server, the client can then subscribe to one or more zones to be notified of changes and the server can publish changes to the zone over the connection.  Clients can choose to unsubscribe from zone updates at any time.

Servers could also use the DSO session to send command-style messages to the client, for example, to instruct a client to stop serving a zone or delete a zone.  No such commands are defined in this version of the specification, but will likely be added in a future version.

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

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

XuD: XFR-using-DOS mechanisms as specified in this document

## 3. Use Cases for XFR-using-DSO

This section includes additional use cases in addition to those specified in [I-D.hzpa-dprive-xfr-over-tls] that XuD can offer.

- Confidentiality. Since this mechanism could, in principle, eliminate the need for NOTIFY and SOA queries it can provide complete confidentiality for the entire zone transfer mechanism.

- Security. For some network configurations it is not desirable to have port 53 on the secondary open to an untrusted network for the sole purpose of receiving NOTIFYs. NOTIFYs can also be trivially spoofed unless secured with TSIG. For the DSO case, secondaries could initiate DSO connections to the primary and following that server-initiated DSO NOTIFY messages could be sent on that connection which could simultaneously be used for SOA and IXFR requests. This would allow a firewall to be restricted to just allowing outgoing connections from secondary to primary. Note that a similar but more constrained mechanism exists for IXFR whereby a short refresh period can be configured which triggers periodic SOA/IXFR requests from the secondary. TODO: Look at the details of the NSD implementation.

- Performance. For the DSO case, a new subscribe/publish mechanism could be envisaged that greatly reducing the number of messages required to perform one transfer.

- Improved error handling and retries. In the DSO case new explicit error codes could be defined that allow a server to indicate the reason for a failed or aborted XFR request. Also a new client initiated message could be used to gracefully cancel AXFRs.

- New command channel. For the DSO case it would be possible to include new server-initiated 'control' commands e.g. 'stop serving this zone', 'delete this zone'.

QUESTION: Is there any case where the primary might want to initiate the DSO connection to the secondary?

## 4. Overview

The figure below provides an outline of the XuD protocol.

Figure 1: XuD protocol [1]

A DNS XuD client subscribes for zone notifications for a particular zone by connecting to the appropriate authoritative server for that zone, and sending DSO message(s) indicating the zone(s) of interest.

When the client loses interest in receiving further updates to these zones, it unsubscribes.

The authoritative server for a DNS zone is any server capable of generating the correct change notifications for a zone. It may be a primary, secondary, or stealth name server [RFC7719].

Standard DNS Queries MAY be sent over a XuD (i.e., DSO) session. For any zone for which the server is authoritative, it MUST respond authoritatively for queries on names falling within that zone both for normal DNS queries and for XuD subscriptions. For names for which the server is acting as a recursive resolver, e.g. when the server is the local recursive resolver, for any query for which it supports XuD subscriptions, it MUST also support standard queries.

XuD imposes less load on the responding server than rapid polling would, but XuD notifications do still have a cost, so XuD clients MUST only create XuD subscriptions for zones they are authorised to transfer.

Generally, as described in the DNS Stateful Operations specification [RFC8490], a client must not keep a session to a server open indefinitely if it has no subscriptions (or other operations) active on that session. A client MAY close a session as soon as it becomes idle, and then if needed in the future, open a new session when required. Alternatively, a client MAY speculatively keep an idle session open for some time, subject to the constraint that it MUST NOT keep a session open that has been idle for more than the session's idle timeout (15 seconds by default) [RFC8490].

## 5. Transport

XuD clients MUST use DNS Stateful Operations [RFC8490] running over TLS over TCP [RFC7858].

The connection for XuD SHOULD be established using port 853, as specified in [RFC7858], unless there is mutual agreement between the secondary and primary to use a port other than port 853 for XuD.

QUESTION: Is there a use case to allow XuD over TCP where confidentiality is not an issue e.g when the zone contents are already publicly available?

## 6. State Considerations

Each XuD server is capable of handling some finite number of XuD subscriptions. This number will vary from server to server and is based on physical machine characteristics, network bandwidth, and

operating system resource allocation. After a client establishes a session to a DNS server, each subscription is individually accepted or rejected. Servers may employ various techniques to limit subscriptions to a manageable level. Correspondingly, the client is free to establish simultaneous sessions to alternate DNS servers that support XuDs for the zone and distribute subscriptions at the client's discretion. In this way, both clients and servers can react to resource constraints.

## 7. Protocol Operation

The XuD protocol is a session-oriented protocol, and makes use of DNS Stateful Operations (DSO) [RFC8490].

For details of the DSO message format refer to the DNS Stateful Operations specification [RFC8490]. Those details are not repeated here.

XuD clients and servers MUST support DSO. A single server can support DNS Queries, DNS Updates, and XuD (using DSO) on the same TCP port.

A XuD exchange begins with the client making a TLS/TCP connection to the appropriate server.

A typical XuD client will immediately issue a DSO Keepalive operation to request a session timeout and/or keepalive interval longer than the the 15-second default values, but this is not required. A XuD client MAY issue other requests on the session first, and only issue a DSO Keepalive operation later if it determines that to be necessary. Sending either a DSO Keepalive operation or a XuD subscription over the TLS/TCP connection to the server signals the client's support of DSO and serves to establish a DSO session.

In accordance with the current set of active subscriptions, the server sends relevant asynchronous XuD notifications to the client. Note that a client MUST be prepared to receive (and silently ignore) XuD notifications for subscriptions it has previously removed, since there is no way to prevent the situation where a XuD notification is in flight from server to client while the client's unsubscribe message cancelling that subscription is simultaneously in flight from client to server.

### 7.1. XuD SUBSCRIBE-XFR

After connecting, and requesting a longer idle timeout and/or keepalive interval if necessary, a XuD client then indicates its desire to receive XuD notifications for a given zone by sending a

SUBSCRIBE-XFR request to the server. A SUBSCRIBE-XFR request is encoded in a DSO message [RFC8490]. This specification defines a primary DSO TLV for XuD SUBSCRIBE-XFR Requests (tentatively DSO Type Code 0x50).

DSO messages with the SUBSCRIBE-XFR TLV as the Primary TLV are not permitted in early data.

The entity that initiates a SUBSCRIBE-XFR request is by definition the client. A server MUST NOT send a SUBSCRIBE-XFR request over an existing session from a client. If a server does send a SUBSCRIBE-XFR request over a DSO session initiated by a client, this is a fatal error and the client should immediately abort the connection with a TLS close_notify alert. See Section 6.1 of [RFC8446].

TODO: Need to define a DSO version of TSIG to cover the SUBSCRIBE-XFR and DSO-XFR responses, since the Additional section count in DSO message MUST be zero. Note the client only needs to use TSIG in the SUBSCRIBE-XFR message to prove it is authorised to request zone transfers, but all DSO-XFR messages should be signed if primary TSIG is required for the authentication model in use.

### 7.1.1. SUBSCRIBE-XFR Request

A SUBSCRIBE-XFR request begins with the standard DSO 12-byte header [RFC8490], followed by the SUBSCRIBE-XFR primary TLV. A SUBSCRIBE-XFR request message is illustrated in Figure 2.

The MESSAGE ID field MUST be set to a unique value, that the client is not using for any other active operation on this DSO session. For the purposes here, a MESSAGE ID is in use on this session if the client has used it in a request for which it has not yet received a response, or if the client has used it for a subscription which it has not yet cancelled using UNSUBSCRIBE-XFR. In the SUBSCRIBE-XFR response the server MUST echo back the MESSAGE ID value unchanged.

The other header fields MUST be set as described in the DSO specification [RFC8490]. The DNS OPCODE field contains the OPCODE value for DNS Stateful Operations (6). The four count fields MUST be zero, and the corresponding four sections MUST be empty (i.e., absent).

The DSO-TYPE is SUBSCRIBE-XFR (tentatively 0x50).

The DSO-LENGTH is the length of the DSO-DATA that follows, which specifies the name and class of the zone and optionally the SOA value of the client's version of the zone.

If the client has no copy of the zone it MUST omit the SOA value to indicate to the server that a DSO-AXFR is required in response (see the next section).

```
                                      1  1  1  1  1  1
        0  1  2  3  4  5  6  7  8  9  0  1  2  3  4  5
      +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+  \
      |                  MESSAGE ID                   |   \
      +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+    |
      |QR| OPCODE(6) |         Z          |   RCODE   |    |
      +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+    |
      |             QDCOUNT (MUST BE ZERO)            |    |
      +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+     > HEADER
      |             ANCOUNT (MUST BE ZERO)            |    |
      +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+    |
      |             NSCOUNT (MUST BE ZERO)            |    |
      +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+    |
      |             ARCOUNT (MUST BE ZERO)            |   /
      +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+  /
      | DSO-TYPE = SUBSCRIBE-XFR (tentatively 0x50)   |
      +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
      |   DSO-LENGTH (number of octets in DSO-DATA)   |
      +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+  \
      |                                               |   \
      \                     NAME                      \    |
      \                                               \    |
      +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+     > DSO-DATA
      |                     CLASS                     |    |
      +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+    |
      |                   SOA value                   |   /
      +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+  /
```

Figure 2: SUBSCRIBE-XFR Request

The DSO-DATA for a SUBSCRIBE-XFR request MUST contain exactly one NAME, CLASS and SOA value. Since SUBSCRIBE-XFR requests are sent over TCP, multiple SUBSCRIBE-XFR DSO request messages can be concatenated in a single TCP stream and packed efficiently into TCP segments.

If accepted, the subscription will stay in effect until the client cancels the subscription using UNSUBSCRIBE-XFR or until the DSO session between the client and the server is closed.

SUBSCRIBE-XFR requests on a given session MUST be unique. A client MUST NOT send a SUBSCRIBE-XFR message that duplicates the NAME, CLASS and SOA value of an existing active subscription on that DSO session. For the purpose of this matching, the established DNS case-

insensitivity for US-ASCII letters applies (e.g., "example.com" and "Example.com" are the same). If a server receives such a duplicate SUBSCRIBE-XFR message this is an error and the server MUST immediately terminate the connection with a TLS close_notify alert.

QUESTION: Is there a use case where a client may want to signal that the version of the zone it holds has been updated via another mechanism and the zone transfer should restart from a different SOA than that currently exchanged between client and server?

DNS wildcarding is not supported. SUBSCRIBE-XFR requests received for zones containing wildcards are considered an error (see below).

A CLASS of 'ANY' (255) is not supported.

### 7.1.2. SUBSCRIBE-XFR Response

Each SUBSCRIBE-XFR request generates exactly one SUBSCRIBE-XFR response from the server. A SUBSCRIBE-XFR request message is illustrated in Figure 3.

A SUBSCRIBE-XFR response begins with the standard DSO 12-byte header [RFC8490]. The QR bit in the header is set indicating it is a response. The header MAY be followed by one or more optional TLVs, such as a Retry Delay TLV.

The MESSAGE ID field MUST echo the value given in the Message ID field of the SUBSCRIBE-XFR request. This is how the client knows which request is being responded to.

A SUBSCRIBE-XFR response message MUST NOT include a SUBSCRIBE-XFR TLV. If a client receives a SUBSCRIBE-XFR response message containing a SUBSCRIBE-XFR TLV then the response message is processed but the SUBSCRIBE-XFR TLV MUST be silently ignored.

```
                                       1  1  1  1  1  1
         0  1  2  3  4  5  6  7  8  9  0  1  2  3  4  5
       +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+  \
       |                  MESSAGE ID                   |   \
       +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+    |
       |QR| OPCODE(6) |         Z          |   RCODE   |    |
       +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+    |
       |             QDCOUNT (MUST BE ZERO)            |    |
       +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+     > HEADER
       |             ANCOUNT (MUST BE ZERO)            |    |
       +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+    |
       |             NSCOUNT (MUST BE ZERO)            |    |
       +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+    |
       |             ARCOUNT (MUST BE ZERO)            |   /
       +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+  /
```

Figure 3: SUBSCRIBE-XFR Response Message

In the SUBSCRIBE-XFR response the RCODE indicates whether or not the subscription was accepted.  Supported RCODEs are as follows:

| Mnemonic | Value | Description |
|---|---|---|
| NOERROR | 0 | SUBSCRIBE-XFR successful. |
| FORMERR | 1 | Server failed to process request due to a malformed request. |
| SERVFAIL | 2 | Server failed to process request due to a problem with the server. |
| NOTIMP | 4 | Server does not implement DSO. |
| REFUSED | 5 | Server refuses to process request for policy or security reasons. |
| NOTAUTH | 9 | Server is not authoritative for the requested name. |
| DSOTYPENI | 11 | SUBSCRIBE-XFR operation not supported. |

Table 1: SUBSCRIBE-XFR Response codes

This document specifies only these RCODE values for SUBSCRIBE-XFR Responses.  Servers sending SUBSCRIBE-XFR Responses SHOULD use one of these values.  Note that NXDOMAIN is not a valid RCODE in response to a SUBSCRIBE-XFR Request.  However, future circumstances may create situations where other RCODE values are appropriate in SUBSCRIBE-XFR Responses, so clients MUST be prepared to accept SUBSCRIBE-XFR Responses with any other RCODE value.

If the server sends a nonzero RCODE in the SUBSCRIBE-XFR response, that means:

a the client is (at least partially) misconfigured,

b the server resources are exhausted, or

c there is some other unknown failure on the server.

In any case, the client shouldn't retry the subscription to this server right away. If a client has other authoritative servers configured for a given zone an alternative server can be tried immediately.

If the client has other successful subscriptions to this server, these subscriptions remain even though additional subscriptions may be refused. Neither the client nor the server are required to close the connection, although, either end may choose to do so.

If the server sends a nonzero RCODE then it SHOULD append a Retry Delay TLV [RFC8490] to the response specifying a delay before the client attempts this operation again. Recommended values for the delay for different RCODE values are given below. These recommended values apply both to the default values a server should place in the Retry Delay TLV, and the default values a client should assume if the server provides no Retry Delay TLV.

For RCODE = 1 (FORMERR) the delay may be any value selected by the implementer. A value of five minutes is RECOMMENDED, to reduce the risk of high load from defective clients.

For RCODE = 2 (SERVFAIL) the delay should be chosen according to the level of server overload and the anticipated duration of that overload. By default, a value of one minute is RECOMMENDED. If a more serious server failure occurs, the delay may be longer in accordance with the specific problem encountered.

For RCODE = 4 (NOTIMP), which occurs on a server that doesn't implement DNS Stateful Operations [RFC8490], it is unlikely that the server will begin supporting DSO in the next few minutes, so the retry delay SHOULD be one hour. Note that in such a case, a server that doesn't implement DSO is unlikely to place a Retry Delay TLV in its response, so this recommended value in particular applies to what a client should assume by default.

For RCODE = 5 (REFUSED), which occurs on a server that implements XuDs, but is currently configured to disallow XuDs, the retry delay may be any value selected by the implementer and/or configured by the

operator. Since it is possible that the misconfiguration may be repaired at any time, the retry delay should not be set too high. By default, a value of 5 minutes is RECOMMENDED.

For RCODE = 9 (NOTAUTH), which occurs on a server that implements XuDs, but is not configured to be authoritative for the requested name, the retry delay may be any value selected by the implementer and/or configured by the operator. Since it is possible that the misconfiguration may be repaired at any time, the retry delay should not be set too high. By default, a value of 5 minutes is RECOMMENDED.

For RCODE = 11 (DSOTYPENI), which occurs on a server that implements DSO but doesn't implement XuD, it is unlikely that the server will begin supporting XuD in the next few minutes, so the retry delay SHOULD be one hour.

For other RCODE values, the retry delay should be set by the server as appropriate for that error condition. By default, a value of 5 minutes is RECOMMENDED.

For RCODE = 9 (NOTAUTH), the time delay applies to requests for other names falling within the same zone. Requests for names falling within other zones are not subject to the delay. For all other RCODEs the time delay applies to all subsequent requests to this server.

After sending an error response the server MAY allow the session to remain open, or MAY send a Retry Delay Operation TLV instructing the client to close the session, as described in the DSO specification [RFC8490]. Clients MUST correctly handle both cases.

## 7.2. XuD Notifications

Once a subscription has been successfully established, the server generates DSO-IXFR messages to send to the client as appropriate. In the case that the server could not provide a DSO-IXFR message based on the SOA received from the client an initial DSO-AXFR message will be sent immediately following the SUBSCRIBE-XFR Response. Subsequent changes to the zone are then communicated to the client in subsequent DSO-IXFR messages.

Until an UNSUBSCRIBE-XFR message is received the server MUST assume that the client is updating the client's version of the zone with the notifications sent and can therefore hold state on the SOA version the client holds. It MUST use this to generate the DSO-IXFR messages sent on a XuD session.

### 7.2.1. DSO-IXFR Message

A DSO-IXFR unidirectional message begins with the standard DSO 12-byte header [RFC8490], followed by the DSO-IXFR primary TLV. A DSO-IXFR message is illustrated in Figure 4.

In accordance with the definition of DSO unidirectional messages, the MESSAGE ID field MUST be zero. There is no client response to a DSO-IXFR message.

The other header fields MUST be set as described in the DSO specification [RFC8490]. The DNS OPCODE field contains the OPCODE value for DNS Stateful Operations (6). The four count fields MUST be zero, and the corresponding four sections MUST be empty (i.e., absent).

The DSO-TYPE is DSO-IXFR (tentatively 0x51).

The DSO-LENGTH is the length of the DSO-DATA that follows, which specifies the changes being communicated.

The DSO-DATA contains one or more change notifications. A DSO-IXFR Message MUST contain at least one change notification. If a DSO-IXFR Message is received that contains no change notifications, this is a fatal error, and the receiver MUST immediately terminate the connection with a TLS close_notify alert.

```
                                      1  1  1  1  1  1
        0  1  2  3  4  5  6  7  8  9  0  1  2  3  4  5
      +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+  \
      |           MESSAGE ID (MUST BE ZERO)           |   \
      +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+    |
      |QR| OPCODE(6) |         Z          |   RCODE   |    |
      +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+    |
      |             QDCOUNT (MUST BE ZERO)            |    |
      +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+     > HEADER
      |             ANCOUNT (MUST BE ZERO)            |    |
      +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+    |
      |             NSCOUNT (MUST BE ZERO)            |    |
      +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+    |
      |             ARCOUNT (MUST BE ZERO)            |   /
      +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+  /
      |      DSO-TYPE = DSO-IXFR (tentatively 0x51)   |
      +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
      |   DSO-LENGTH (number of octets in DSO-DATA)   |
      +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+  \
      |                                               |   \
      |                                               |    |
      |            IXFR BODY AS PER RFC1995           |     > DSO-DATA
      |                                               |    |
      |                                               |   /
      +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+  /
```

Figure 4: DSO-IXFR Message

The DSO-DATA in a DSO-IXFR message is identical to the contents of a [RFC1995] IXFR message that would be sent to communicate the same zone incremental zone transfer over UDP or TCP i.e. the set of one or more difference sequences that follow the DNS Header in an IXFR message.

When processing the records received in a DSO-IXFR Message, the receiving client MUST validate that the zone being updated correspond with at least one currently active subscription on that session. Specifically, the SOA name and CLASS MUST match the SOA name and CLASS given in a SUBSCRIBE-XFR request, subject to the usual established DNS case-insensitivity for US-ASCII letters.

### 7.2.2. Fallback to AXFR

The format of the DSO-AXFR message is a standard DSO header with DSO-TYPE of DSO-AXFR (tentatively DSO Type Code 0x52) and the body is identical to a [RFC5936] AXFR response body.

TODO: More detail here.

If the SUBSCRIBE-XFR message contained no SOA value, the server MUST send a DSO-AXFR message as its first message on the connection.

Alternatively if incremental zone transfer is not available, the entire zone MAY be returned in a DSO-AXFR message.

QUESTION: Should we bother with a separate DSO-AXFR message or just allow full zone transfer inside the DSO-IXFR message as with [RFC1995] IXFR? A separate message type makes is more explicit and IXFR was constrained by having to respond to a IXFR request.

### 7.3. XuD UNSUBSCRIBE-XFR

To cancel an individual subscription without closing the entire DSO session, the client sends an UNSUBSCRIBE-XFR message over the established DSO session to the server. The UNSUBSCRIBE-XFR message is encoded as a DSO unidirectional message [RFC8490]. This specification defines a primary unidirectional DSO TLV for XuD UNSUBSCRIBE-XFR Messages (tentatively DSO Type Code 0x53).

A server MUST NOT initiate an UNSUBSCRIBE-XFR message. If a server does send an UNSUBSCRIBE-XFR message over a DSO session initiated by a client, this is a fatal error and the client should immediately abort the connection with a TLS close_notify alert.

#### 7.3.1. UNSUBSCRIBE-XFR Message

An UNSUBSCRIBE-XFR unidirectional message begins with the standard DSO 12-byte header [RFC8490], followed by the UNSUBSCRIBE-XFR primary TLV. An UNSUBSCRIBE-XFR message is illustrated in Figure 5.

In accordance with the definition of DSO unidirectional messages, the MESSAGE ID field MUST be zero. There is no server response to an UNSUBSCRIBE-XFR message.

The other header fields MUST be set as described in the DSO specification [RFC8490]. The DNS OPCODE field contains the OPCODE value for DNS Stateful Operations (6). The four count fields MUST be zero, and the corresponding four sections MUST be empty (i.e., absent).

The DSO-TYPE is UNSUBSCRIBE-XFR (tentatively 0x53).

The DSO-LENGTH field contains the value 2, the length of the 2-octet MESSAGE ID contained in the DSO-DATA.

The DSO-DATA contains the value given in the MESSAGE ID field of an active SUBSCRIBE-XFR request. This is how the server knows which

SUBSCRIBE-XFR request is being cancelled. After receipt of the UNSUBSCRIBE-XFR message, the SUBSCRIBE-XFR request is no longer active.

It is allowable for the client to issue an UNSUBSCRIBE-XFR message for a previous SUBSCRIBE-XFR request for which the client has not yet received a SUBSCRIBE-XFR response. This is to allow for the case where a client starts and stops a subscription in less than the round-trip time to the server. The client is NOT required to wait for the SUBSCRIBE-XFR response before issuing the UNSUBSCRIBE-XFR message.

Consequently, it is possible for a server to receive an UNSUBSCRIBE-XFR message that does not match any currently active subscription. This can occur when a client sends a SUBSCRIBE-XFR request, which subsequently fails and returns an error code, but the client sent an UNSUBSCRIBE-XFR message before it became aware that the SUBSCRIBE-XFR request had failed. Because of this, servers MUST silently ignore UNSUBSCRIBE-XFR messages that do not match any currently active subscription.

```
                                     1  1  1  1  1  1
       0  1  2  3  4  5  6  7  8  9  0  1  2  3  4  5
     +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+  \
     |           MESSAGE ID (MUST BE ZERO)           |   \
     +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+    |
     |QR| OPCODE(6) |         Z          |   RCODE   |    |
     +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+    |
     |             QDCOUNT (MUST BE ZERO)            |    |
     +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+     > HEADER
     |             ANCOUNT (MUST BE ZERO)            |    |
     +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+    |
     |             NSCOUNT (MUST BE ZERO)            |    |
     +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+    |
     |             ARCOUNT (MUST BE ZERO)            |   /
     +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+  /
     | DSO-TYPE = UNSUBSCRIBE-XFR (tentatively 0x53) |
     +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
     |                DSO-LENGTH (2)                 |
     +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+  \
     |              SUBSCRIBE-XFR MESSAGE ID             |   > DSO-DATA
     +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+  /
```

Figure 5: UNSUBSCRIBE-XFR Message

QUESTION: Do we need the equivalent of a RECONFIRM message from DNS PUSH Notifications [I-D.ietf-dnssd-push]?

### 7.4. Authentication

The authentication considerations are largely the same as those presented in [I-D.hzpa-dprive-xfr-over-tls].

### 7.5. Multi-primary configurations

The multi-primary considerations share some of the same issues as those presented in [I-D.hzpa-dprive-xfr-over-tls] but are different because the client is not performing SOA queries.

TODO: More detail required here.

### 7.6. DNS Stateful Operations TLV Context Summary

This document defines four new DSO TLVs. As suggested in Section 8.2 of the DNS Stateful Operations specification [RFC8490], the valid contexts of these new TLV types are summarized below.

The client TLV contexts are:

C-P: Client request message, primary TLV

C-U: Client unidirectional message, primary TLV

C-A: Client request or unidirectional message, additional TLV

CRP: Response back to client, primary TLV

CRA: Response back to client, additional TLV

| TLV Type | C-P | C-U | C-A | CRP | CRA |
|---|---|---|---|---|---|
| SUBSCRIBE-XFR | X | | | | |
| DSO-IXFR | | | | | |
| DSO-AXFR | | | | | |
| UNSUBSCRIBE-XFR | | X | | | |

Table 2: DSO TLV Client Context Summary

The server TLV contexts are:

S-P: Server request message, primary TLV

S-U: Server unidirectional message, primary TLV

S-A: Server request or unidirectional message, additional TLV

SRP: Response back to server, primary TLV

SRA: Response back to server, additional TLV

| TLV Type | S-P | S-U | S-A | SRP | SRA |
|---|---|---|---|---|---|
| SUBSCRIBE-XFR | | | | | |
| DSO-IXFR | | X | | | |
| DSO-AXFR | | X | | | |
| UNSUBSCRIBE-XFR | | | | | |

Table 3: DSO TLV Server Context Summary

## 8. IANA Considerations

This document also defines four new DNS Stateful Operation TLV types to be recorded in the IANA DSO Type Code Registry.

| Name | Value | Early Data | Status | Definition |
|---|---|---|---|---|
| SUBSCRIBE-XFR | TBA (0x50) | NO | Standards Track | Section 7.1 |
| DSO-IXFR | TBA (0x51) | NA | Standards Track | Section 7.1 |
| DSO-AXFR | TBA (0x51) | NA | Standards Track | Section 7.2 |
| UNSUBSCRIBE-XFR | TBA (0x52) | NA | Standards Track | Section 7.2 |

Table 5: IANA DSO TLV Type Code Assignment

## 9. Implementation Considerations

TBD

## 10. Implementation Status

TBD

## 11. Security Considerations

This document specifies a security measure against a DNS risk: the risk that an attacker collects entire DNS zones through eavesdropping on clear text DNS zone transfers. It presents a new Security Consideration for DNS. Some questions to discuss are:

- Should DoT in this new case be required to use only TLS 1.3 and higher to avoid residual exposure?

- How should padding be used in IXFR?

- Should there be an option to 'pad' an AXFR response (i.e. a set of AXFR responses on a given connection) to hide the zone size?

## 12. Acknowledgements

## 13. Changelog

draft-zatda-dprive-xfr-using-dso-00

- Initial commit

## 14. References

### 14.1. Normative References

[I-D.hzpa-dprive-xfr-over-tls]
Zhang, H., Aras, P., Toorop, W., Dickinson, S., and A. Mankin, "DNS Zone Transfer over TLS", draft-hzpa-dprive-xfr-over-tls-01 (work in progress), March 2019.

[I-D.ietf-dnssd-push]
Pusateri, T. and S. Cheshire, "DNS Push Notifications", draft-ietf-dnssd-push-21 (work in progress), July 2019.

[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>.

[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>.

[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>.

[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>.

## 14.2. Informative References

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

[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>.

[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>.

## 14.3. URIs

[1] https://github.com/Sinodun/draft-xfr-using-dso/blob/master/draft-01-svg/XuD_Protocol.svg

Authors' Addresses

Han Zhang
Salesforce
San Francisco, CA
United States

Email: hzhang@salesforce.com

Pallavi Aras
Salesforce
Herndon, VA
United States

Email: paras@salesforce.com

Willem Toorop
NLnet Labs
Science Park 400
Amsterdam  1098 XH
The Netherlands

Email: willem@nlnetlabs.nl

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

Email: sara@sinodun.com

Allison Mankin
Salesforce
Herndon, VA
United States

Email: allison.mankin@gmail.com