DNSOP Working Group
Internet-Draft
Intended status: Standards Track
Expires: February 15, 2017

R. Bellis
ISC
S. Cheshire
Apple Inc.
J. Dickinson
S. Dickinson
Sinodun
A. Mankin
Salesforce
T. Pusateri
Unaffiliated

August 14, 2016

DNS Session Signaling
draft-ietf-dnsop-session-signal-00

Abstract

The EDNS(0) Extension Mechanism for DNS is explicitly defined to only have "per-message" semantics. This document defines a new Session Signaling Opcode used to carry persistent "per-session" type-length-values (TLVs), and defines an initial set of TLVs used to manage session timeouts and termination.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on February 15, 2017.

Table of Contents

1. Introduction

The use of transports other than UDP for DNS is being increasingly specified, for example, DNS-over-TCP [RFC7766], DNS-over-TLS [RFC7858] and recent work on DNS Push Notifications [I-D.ietf-dnssd-push]. Such transports frequently use persistent, long-lived sessions and therefore when using them for transporting DNS messages it is of benefit to have a mechanism that can establish parameters associated with those sessions, such as timeouts. In such situations it is also advantageous to support server initiated messages.

The EDNS(0) Extension Mechanism for DNS [RFC6891] is explicitly defined to only have "per-message" semantics. Whilst EDNS(0) has been used to signal at least one session related parameter (the EDNS TCP KeepAlive option [RFC7828]) the result is less than optimal due to the restrictions imposed by the EDNS(0) semantics and the lack of server initiated signalling. This document defines a new Session Signaling Opcode used to carry persistent "per-session" type-length-values (TLVs), and defines an initial set of TLVs used to manage session timeouts and termination.

With EDNS(0), multiple options may be packed into a single OPT RR, and there is no generalized mechanism for a client to be able to tell whether a server has processed or otherwise acted upon each individual option within the combined OPT RR. The specifications for each individual option need to define how each different option is to be acknowledged, if necessary.

With Session Signaling, in contrast, each Session Signaling operation is communicated in its own separate DNS message, and the RCODE in the response indicates the success or failure of that operation.

It should be noted that the message format for Session Signaling operations (see Section 3.2) differs from the DNS packet format used for standard queries and responses, in that it has a shorter header (four octets instead of usual twelve octets).

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 "Key words for use in RFCs to Indicate Requirement Levels" [RFC2119].

The terms "initiator" and "responder" correspond respectively to the initial sender and subsequent receiver of a Session Signaling TLV, regardless of which was the "client" and "server" in the usual DNS sense.

The term "sender" may apply to either an initiator or responder.

The term "session" in the context of this document means the exchange of DNS messages using an end-to-end transport protocol where:

- The connection between client and server is persistent and relatively long-lived (i.e. minutes or hours, rather than seconds).

- Either end of the connection may initiate messages to the other

- Messages are delivered in order

3. Protocol Details

Session Signaling messages MUST only be carried in protocols and in environments where a session may be established according to the definition above. Standard DNS over TCP [RFC1035], and DNS over TLS [RFC7858] are suitable protocols. DNS over plain UDP is not appropriate since it fails on the requirement for in-order message delivery, and, in the presence of NAT gateways and firewalls with short UDP timeouts, it fails to provide a persistent bi-directional communication channel unless an excessive amount of keepalive traffic is used.

Session Signaling messages relate only to the specific session in which they are being carried. Where a middle box (e.g. a DNS proxy, forwarder, or session multiplexer) is in the path the message MUST NOT be blindly forwarded in either direction by that middle box. This does not preclude the use of these messages in the presence of a NAT box that rewrites IP-layer or transport-layer headers but otherwise maintains the effect of a single session.

A client MAY attempt to initiate Session Signaling messages at any time on a connection; receiving a NOTIMP response in reply indicates that the server does not implement Session Signaling, and the client SHOULD NOT issue further Session Signaling messages on that connection.

A server SHOULD NOT initiate Session Signaling messages until a client-initiated Session Signaling message is received first, unless in an environment where it is known in advance by other means that both client and server support Session Signaling. This requirement is to ensure that the clients that do not support Session Signaling do not receive unsolicited inbound Session Signaling messages that they would not know how to handle.

3.1. Session Lifecycle

A session begins when a client makes a new connection to a server.

The client may perform as many DNS operations as it wishes on the newly created connection. Operations SHOULD be pipelined (i.e., the client doesn't need wait for a reply before sending the next message). The server MUST act on messages in the order they are received, but responses to those messages MAY be sent out of order, if appropriate.

When a server implementing this specification receives a new connection from a client, it MUST begin by internally assigning an initial idle timeout of 30 seconds to that connection. At both servers and clients, the generation or reception of any complete DNS message, including DNS requests, responses, updates, or Session Signaling messages, resets the idle timer for that connection (see [RFC7766]).

If, at any time during the life of the connection, half the idle-timeout value (i.e., 15 seconds by default) elapses without any DNS messages being sent or received on a connection, then the connection is considered stale and the client MUST take action. When this happens the client MUST either send at least one new message to reset the idle timer - such as a Session Signaling Idle Timeout message (see Section 4.2.1), or any other valid DNS message - or close the connection.

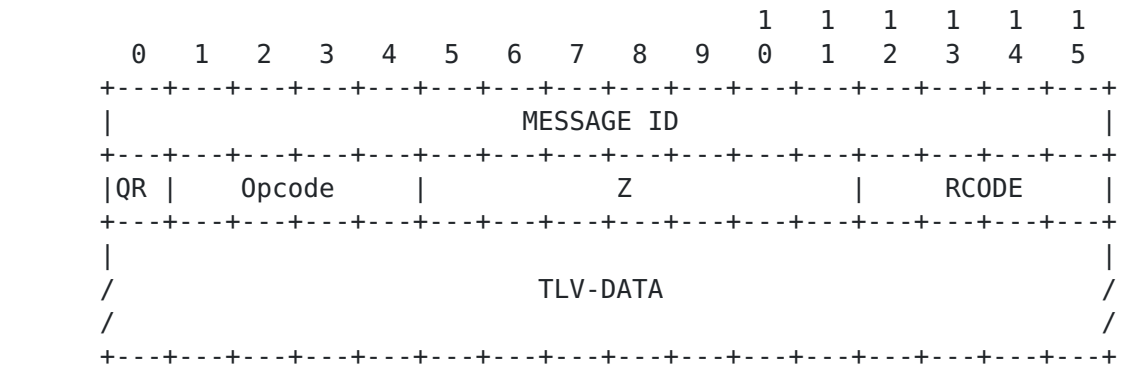
If, at any time during the life of the connection, the full idle-timeout value (i.e., 30 seconds by default) elapses without any DNS messages being sent or received on a connection, then the connection is considered delinquent and the server SHOULD forcibly terminate the connection. For sessions over TCP (or over TLS-over-TCP), to avoid the burden of having a connection in TIME-WAIT state, instead of closing the connection gracefully with a TCP FIN the server SHOULD abort the connection with a TCP RST. (In the Sockets API this is achieved by setting the SO_LINGER option to zero before closing the socket.)

If the client wishes to keep an idle connection open for longer than the default duration without having to send traffic every 15 seconds, then it uses the Session Signaling Idle Timeout message to request a longer idle timeout (see Section 4.2.1).

A client is not required to wait until half of the idle-timeout value before closing a connection. A client MAY close a connection at any time, at the client's discretion, if it has no further need for the connection at that time.

3.2. Message Format

A Session Signaling message begins with the first 4 octets of the standard DNS message header [RFC1035], with the Opcode field set to the Session Signaling Opcode. A Session Signaling message does not contain the QDCOUNT, ANCOUNT, NSCOUNT and ARCOUNT fields fields used in standard DNS queries and responses. This 4-octet header is followed by a single Session Signaling TLV.



The MESSAGE ID, QR, Opcode and RCODE fields have their usual meanings [RFC1035].

The Z bits are currently unused, and SHOULD be set to zero (0) in requests and responses unless re-defined in a later specification.

3.3. Message Handling

On a connection between a client and server that support Session Signaling, either end may unilaterally send Session Signaling messages at any point in the lifetime of a session, and therefore either client or server may be the initiator of a message. The initiator MUST set the value of the QR bit in the DNS header to zero (0), and the responder MUST set it to one (1).

<<TODO: Specify behaviour on receipt of a message from an initiator with the QR bit set to 1, etc.>>

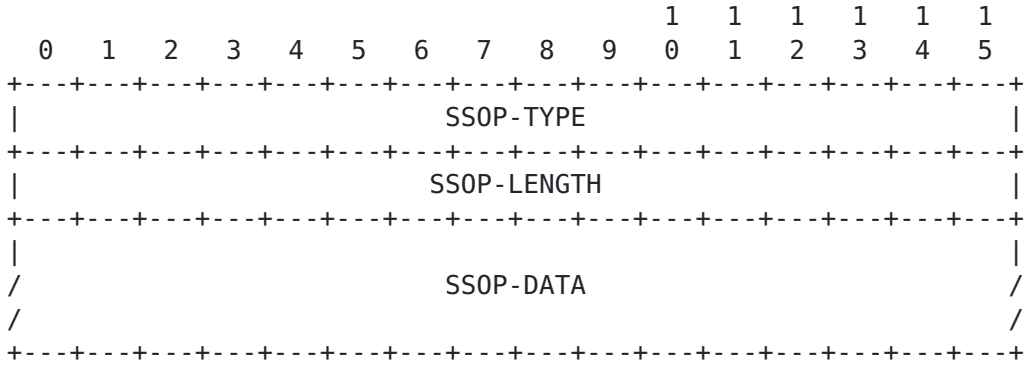
Every Session Signaling request message MUST elicit a response (which MUST have the same ID in the DNS message header as in the request).

<< RB: should the presence of a SS message create a "sequencing point", such that all pending responses must be answered? SC: I do not believe so. We can define an explicit SS "sequencing point" opcode for this if necessary. >>

The RCODE value in a response uses a subset of the standard (non-extended) RCODE values from the IANA DNS RCODEs registry, interpreted as follows:

Code	Mnemonic	Description
0	NOERROR	TLV processed successfully
1	FORMERR	TLV format error
4	NOTIMP	Session Signaling not supported
5	REFUSED	TLV declined for policy reasons

3.4. TLV Format



SSOP-TYPE: A 16 bit field in network order giving the type of the current Session Signaling TLV per the IANA DNS Session Signaling Type Codes Registry.

<< SC: I changed SESSION-TYPE to SSOP-TYPE because to me SESSION-TYPE read as "type of session" which it is not. It is Session Signaling Operation Type, Session Signaling Operation Length, Session Signaling Operation Data. >>

SSOP-LENGTH: A 16 bit field in network order giving the size in octets of SSOP-DATA.

SSOP-DATA: Type-code specific.

4. Mandatory TLVs

4.1. Session Management Support TLVs

4.1.1. "Not Implemented"

Since the "NOTIMP" RCODE in the DNS message header is used to indicate lack of support for the Session Signaling Opcode itself, a different mechanism is used to indicate lack of support of a specific SSOP-TYPE. If a server that supports Session Signaling receives a Session Signaling query message (QR bit zero) with an SSOP-TYPE it does not support, it returns a response message (QR bit one) containing a single Session Signaling SSOP-NOTIMP TLV (0). The MESSAGE ID in the message header serves to tell the client which of its requests was not understood.

The SSOP-NOTIMP TLV has no SSOP-DATA.

4.2. Session Management TLVs

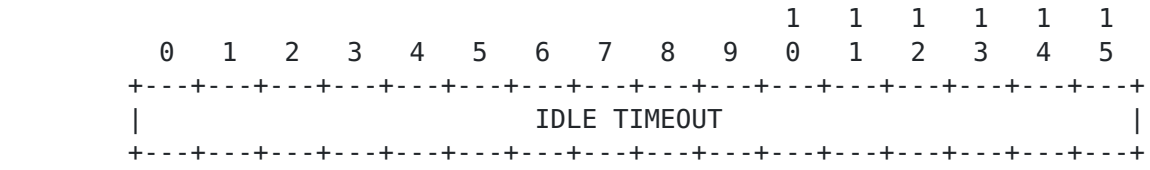
4.2.1. Idle Timeout

The Idle Timeout TLV (1) is be used by a client to reset a connection's idle timer, and at the same time to request what the idle timeout should be from this point forward in the connection.

The Idle Timeout TLV also MAY be initiated by a server, to unilaterally inform the client of a new idle timeout this point forward in this connection.

It is not required that the Idle Timeout TLV be used in every session. While many Session Signaling operations (such as DNS Push Notifications [I-D.ietf-dnssd-push]) will be used in conjunction with a long-lived connection, this is not required, and in some cases the default 30-second timeout may be perfectly appropriate.

The SSOP-DATA for the the Idle Timeout TLV is as follows:



IDLE TIMEOUT: the idle timeout for the current session, specified as a 16 bit word in network order in units of 100 milliseconds. This is the timeout at which the server will forcibly terminate the connection with a TCP RST (or equivalent for other protocols); after half this interval the client MUST take action to either preserve the connection, or close it if it is no longer needed.

In a client-initiated Session Signaling Idle Timeout message, the IDLE TIMEOUT contains the client's requested value for the idle timeout.

In a server response to a client-initiated message, the IDLE TIMEOUT contains the server's chosen value for the idle timeout, which the client MUST respect. This is modeled after the DHCP protocol, where the client requests a certain lease lifetime, but the server is the ultimate authority for deciding what lease lifetime is actually granted.

In a server-initiated Session Signaling Idle Timeout message, the IDLE TIMEOUT unilaterally informs the client of the new idle timeout this point forward in this connection.

In a client response to a server-initiated message, there is no SSOP-DATA. SSOP-LENGTH is zero.

<<TODO: Specify the behaviour when a server sends a 0 IDLE TIMEOUT.>>

The Idle Timeout TLV (3) has similar intent to the EDNS TCP Keepalive Option [RFC7828]. A client/server pair that supports Session Signaling MUST NOT use the EDNS TCP KeepAlive option within any message once bi-directional Session Signaling support has been confirmed.

<< SC: And if client or server does use EDNS TCP Keepalive, then other end should... close connection? Silently ignore? >>

4.2.2. Terminate Session

There may be rare cases where a server is overloaded and wishes to shed load. If the server simply closes connections, the likely behaviour of clients is to detect this as a network failure, and reconnect.

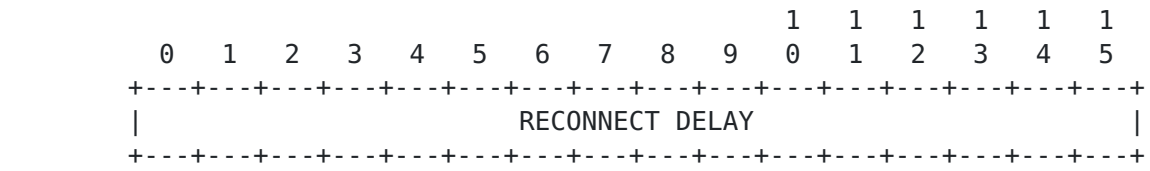
To avoid this reconnection implosion, the server sends a Terminate Session TLV (2) to inform the client of the overload situation. After sending a Terminate Session TLV the server MUST NOT send any further messages on the connection.

The Terminate Session TLV (2) MUST NOT be initiated by a client.

<<TODO: Specify behaviour if the Terminate Session TLV is initiated by a client.>>

Upon receipt of the Terminate Session TLV (2) the client MUST make note of the reconnect delay for this server, and then immediately close the connection.

The SSOP-DATA is as follows:



RECONNECT DELAY: a time value, specified as a 16 bit word in network order in units of 100 milliseconds, within which the client MUST NOT establish a new session to the current server.

The RECOMMENDED value is 10 seconds. << RB: text required here about default values for load balancers, etc >>

5. IANA Considerations

5.1. DNS Session Signaling Opcode Registration

IANA are directed to assign the value TBD for the Session Signaling Opcode in the DNS OpCodes Registry.

5.2. DNS Session Signaling Type Codes Registry

IANA are directed to create the DNS Session Signaling Type Codes Registry, with initial values as follows:

Type	Name	Status	Reference
0	SSOP-NOTIMP	Standard	RFC-TBD1
1	SSOP-Keepalive	Standard	RFC-TBD1
2	SSOP-Terminate	Standard	RFC-TBD1
3 - 63	Unassigned, reserved for session management TLVs		
64 - 63487	Unassigned		
63488 - 64511	Reserved for local / experimental use		
64512 - 65535	Reserved for future expansion		

Registration of additional Session Signaling Type Codes requires Expert Review. << RB: definition of process required? >>

6. Security Considerations

If this mechanism is to be used with DNS over TLS, then these messages are subject to the same constraints as any other DNS over TLS messages and MUST NOT be sent in the clear before the TLS session is established.

7. Acknowledgements

TBW

8. References

8.1. Normative References

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

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

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

[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, <http://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, <http://www.rfc-editor.org/info/rfc7828>.

8.2. Informative References

[I-D.ietf-dnssd-push] Pusateri, T. and S. Cheshire, "DNS Push Notifications", draft-ietf-dnssd-push-08 (work in progress), July 2016.

[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, <http://www.rfc-editor.org/info/rfc7858>.

Authors' Addresses

Ray Bellis
Internet Systems Consortium, Inc.
950 Charter Street
Redwood City CA 94063
USA

Phone: +1 650 423 1200
Email: ray@isc.org

Stuart Cheshire
Apple Inc.
1 Infinite Loop
Cupertino CA 95014
USA

Phone: +1 408 974 3207
Email: cheshire@apple.com

John Dickinson
Sinodun Internet Technologies
Magadalen Centre
Oxford Science Park
Oxford OX4 4GA
United Kingdom

Email: jad@sinodun.com

Sara Dickinson
Sinodun Internet Technologies
Magadalen Centre
Oxford Science Park
Oxford OX4 4GA
United Kingdom

Email: sara@sinodun.com

Allison Mankin
Salesforce

Email: allison.mankin@gmail.com

Tom Pusateri
Unaffiliated

Email: pusateri@bangj.com