Workgroup: Network Working Group
Internet-Draft:
draft-peltan-edns-presentation-format-00
Updates: 8427 (if approved)
Published: 23 November 2022
Intended Status: Standards Track
Expires: 27 May 2023
Authors: L. Peltan T. Carpay
CZ.NIC NLnet Labs

EDNS Presentation and JSON Format

Abstract

This document describes textual and JSON representation format of EDNS option. It also modifies the escaping rules of JSON representation of DNS messages, previously defined in RFC8427.

Status of This Memo

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

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

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

This Internet-Draft will expire on 27 May 2023.

Table of Contents

1. Introduction

A DNS record[RFC1035] of any type can be converted between its binary Wire format and textual Presentation format. The Wire format is used in DNS messages transferred over the Internet, while the Presentation format is used not only in Zone Files (called "master files" in the referenced document), but also to display the contents of DNS messages to humans by debugging utilities, and possible other use-cases.

The Presentation format can be however processed also programatically and also converted back to Wire Format unambiguously.

The EDNS[RFC6891] option pseudorecord does not appear in Zone Files, but it sometimes needs to be converted to human-readable or even machine-readable textual representation. This document describes such a Presentation Format of the OPT pseudorecord. It is advised to use this when displaying an OPT pseudorecord to humans. It is recommended to use this when the textual format is expected to be machine-processed further.

The JSON[RFC8259] representation[RFC8427] of DNS messages is also helpful as both human-readable and machine-readable format (despite the limitation in non-preservation of the order of options, which prevents reversing the conversion unambiguosly), but it did not define JSON representation of EDNS option pseudorecord. This document defines it.

The aforementioned document[RFC8427] also defined ambiguous and possibly conflicting rules for escaping special characters when representing DNS names in JSON. This documents modifies and clarifies those rules.

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.

*EDNS(0) signifies EDNS version 0.

*"Decimal value" means an integer displayed in decimals with no leading zeroes.

* Base16 is the representation of arbitrary binary data by an even number of case-insensitive hexadecimal digits ([RFC4648], Section 8).

* "Followed by" in terms of strings denotes their concatenation, with no other characters nor space between them.

* Backslash is the character called also Reverse Solidus, ASCII code 0x5c.

* Zero-octet is an octet with all bits set to 0, i.e. ASCII code 0x00.

* "Note" denotes a sentence that is not normative. Instead, it points out some non-obvious consequences of previous statements.

3. Version-independent Presentation Format

EDNS versions other than 0 are not yet specified, but an OPT pseudorecord with version field set to value other than zero might in theory appear in DNS messages. This section specifies how to convert such OPT pseudorecord to Presentation format. This procedure SHOULD NOT be used for EDNS(0). One possible exception is displaying a malformed EDNS(0) record.

OPT pseudorecord is in this case represented the same way as a RR of unknown type according to [RFC3597], Section 5. In specific:

* Owner Name is the Owner Name of the OPT record. Note that this is always . (DNS Root Domain Name) unless malformed.

* TTL is Decimal value of the 32-bit big-endian integer appearing at the TTL position of OPT pseudorecord Wire format, see [RFC6891], Section 6.1.3.

* CLASS is a text representation of the 16-bit integer at the CLASS position of OPT pseudorecord Wire format (UDP payload size happens to appear there). This will usually result in CLASS#### (where #### will be the Decimal value), but it might also result for example in IN or CH if the value is 1 or 4, respectively.

* TYPE is either TYPE41 or OPT.

* RDATA is formatted by \#, its length as Decimal value, and data as Base16 as per [RFC3597], Section 5.

Example:

```
. 16859136 CLASS1232 TYPE41 \# 6 000F00020015
```

4. EDNS(0) Presentation Format

The EDNS(0) Presentation Format follows RR format of the master file ([RFC1035], Section 5.1), including quotation of non-printable characters, multi-line format using round brackets, and semicolons denoting comments.

Depending on use-case, implementations MAY choose to display only RDATA. In the case the resource-record-like Presentation format is desired, the following applies:

- *Owner Name MUST be . (DNS Root Domain Name).

- *TTL MAY be omitted. If it is present, it MUST be 0 (zero). Note that this differs from DNS RR wire-to-text conversion, as well as Version-independent Presentation Format (Section 3).

- *CLASS MAY be omitted. If it is present, it MUST be ANY.

- *TYPE MUST be EDNS0.

RDATA consists of at least three <character-string>s ([RFC1035], Section 5.1), one for each field. Each field consists of a Field-name, followed by an equal sign (=), followed by a Field-value. If the Field-value is empty or omitted, the equal sign MUST be omitted as well. For each field, the Field-name and the Field-value are defined by this document, or by the specification of the respective EDNS Option. If it is not, a generic Field-name and Field-value from Section 4.4 applies. However, those generics MAY be used for any Option at all times.

The first three fields, Flags (Section 4.1), Extended RCODE (Section 4.2), and UDP Payload Size (Section 4.3) MUST always be present. The rest of the fields are based on Options in the OPT record [RFC6891], Section 6.1.2. They MUST be presented in the same order as they appear in wire format. It is recommended to use the multi-line format with comments at each field, together with a more human-readable form of the contents of each option when available. See Examples (Section 5).

4.1. Flags

The first field's Field-name is FLAGS and its Field-value is 0 (zero) if the EDNS flags is zero.

Otherwise, the Field-value consists of comma-separated list of the items BIT##, where ## is a Decimal value. BITn is present in the list if and only if n-th bit (the most significant bit being 0-th) of flags is set to 1. If the Flag of the bit is specified in

[IANA.EDNS.Flags], the Flag SHOULD be used instead of BIT##. (So far, the only known Flag is DO.)

Examples:

```
FLAGS=0

FLAGS=DO,BIT1

FLAGS=BIT3,BIT7,BIT15
```

4.2. Extended RCODE

The second field's Field-name is RCODE and its Field-value is RCODE###, where ### stands for the DNS message extended RCODE as Decimal value, computed from both the OPT record and the DNS Message Header. If the lower four bits of extended RCODE in DNS Message Header can not be used, the Field-value is UNKNOWNRCODE###, where ### stands for the DNS message extended RCODE as Decimal value, with the lower four bits set to zero (i.e. the four-bit left shift still applies). If the extended RCODE has been computed completely and it is listed in [IANA.RCODEs], its Name should be used instead of RCODE###. The Name is case-insensitive.

Examples:

```
RCODE=NXDOMAIN

RCODE=RCODE3841

RCODE=UNKNOWNRCODE3840
```

4.3. UDP Payload Size

The third field's Field-name is UDPSIZE and its Field-value is the UDP payload size as Decimal value.

4.4. Unrecognized Option

EDNS options that are not part of this specification and their own specifications do not specify their Field-name and Field-value MUST be displayed according this subsection. Other options (specified below or otherwise) MAY be displayed so as well.

Unrecognized option Field-name is OPT##, where ## stands for its OPTION-CODE, and Field-value is its OPTION-VALUE displayed as Base16.

4.5. LLQ Option

The LLQ (OPTION-CODE 1 [RFC8764]) Field-name is LLQ and Field-value is comma-separated tuple of LLQ-VERSION, LLQ-OPCODE, LLQ-ERROR, LLQ-ID, and LLQ-LEASE as Decimal values. The numeric values of LLQ-OPCODE and LLQ-ERROR MAY be substituted with their textual representations listed in [RFC8764], Section 3.1.

Examples:

```
LLQ=1,1,0,0,3600

LLQ=1,LLQ-SETUP,NO-ERROR,0,3600
```

4.6. NSID Option

The NSID (OPTION-CODE 3 [RFC5001]) Field-name is NSID and Field-value is its OPTION-VALUE displayed as Base16.

It is recommended to add a comment with ASCII representation of the value.

4.7. DAU, DHU and N3U Options

The DAU, DHU, and N3U (OPTION-CODES 5, 6, 7, respectively [RFC6975]) Field-names are DAU, DHU, and N3U, respectively, and their Field-values consist of comma-separated lists of ALG-CODEs as Decimal values or the textual representations of the ALG-CODEs (called mnemonic in the referenced documents) found in their respective IANA registries [IANA.EDNS.DAU][IANA.EDNS.DHU][IANA.EDNS.N3U].

Examples:

```
DAU=RSASHA256,RSASHA512,ECDSAP256SHA256,ECDSAP384SHA384,ED25519
DHU=SHA-1,SHA-256,SHA-384
N3U=SHA-1

DAU=8,10,13,14,15
DHU=1,2,4
N3U=1
```

4.8. Edns-Client-Subnet Option

The EDNS Client Subnet (OPTION-CODE 8 [RFC7871]) Field-name is ECS and if FAMILY is neither IPv4 (1) nor IPv6 (2), its Field-value is the whole OPTION-VALUE as Base16. Otherwise, it consists of the textual IPv4 or IPv6 address ([RFC1035], Section 3.4.1, [RFC4291], Section 2.2), followed by a slash (/), followed by SOURCE PREFIX-LENGTH as Decimal value, followed by another slash, followed by

SCOPE PREFIX-LENGTH as Decimal value. If SCOPE PREFIX-LENGTH is zero, it MUST be omitted together with the second slash.

Examples:

```
ECS=1.2.3.4/24

ECS=1234::2/56/48

ECS=00052000010203040506070 8
```

4.9. EDNS EXPIRE Option

The EDNS EXPIRE (OPTION-CODE 9 [RFC7314]) Field-name is EXPIRE and its Field-value, if present, is displayed as Decimal value.

4.10. Cookie Option

The DNS Cookie (OPTION-CODE 10 [RFC7873]) Field-name is COOKIE and its Field-value consists of the Client Cookie as Base16, followed by a comma, followed by the Server Cookie as Base16. The comma and Server Cookie are displayed only if OPTION-LENGTH is greater than 8.

4.11. Edns-Tcp-Keepalive Option

The edns-tcp-keepalive (OPTION-CODE 11 [RFC7828]) Field-name is KEEPALIVE and its Field-value is the TIMEOUT in seconds displayed as decimal number with exactly one decimal digit and a dot as decimal separator.

4.12. Padding Option

The Padding (OPTION-CODE 12 [RFC7830]) Field-name is PADDING and its Field-value is its OPTION-VALUE displayed as Base16. If the OPTION-VALUE consists only of zero-octets, it SHOULD be substituted with an alternative Field-value [###], where ### stands for OPTION-LENGTH as Decimal value.

4.13. CHAIN Option

The CHAIN (OPTION-CODE 13 [RFC7901]) Field-name is CHAIN and its Field-value, the Closest trust point, is displayed as a textual Fully-Qualified Domain Name.

4.14. Edns-Key-Tag Option

The edns-key-tag (OPTION-CODE 14 [RFC8145], Section 4) Field-name is KEYTAG and its Field-value is displayed as a comma-separated list of Decimal values.

4.15. Extended DNS Error Option

The Extended DNS Error (OPTION-CODE 15 [RFC8914]) Field-name is EDE and the Field-value is its INFO-CODE as Decimal value. It is recommended to add a comment with the Purpose of the given code (first presented in [RFC8914], Section 5.2 and then governed by [IANA.EDNS.EDE]).

If the EXTRA-TEXT is nonempty, it MUST be displayed as another field, with Field-name EDETXT and Field-value being the EXTRA-TEXT string as-is.

Note that RFC1035-style escaping applies to all non-printable and non-ASCII characters, including some eventual UTF-8 bi-characters and possible trailing zero-octet. Also note that any presence of spaces requires the whole <character-string> to be enclosed in quotes, not just the Field-value.

Examples:

```
EDE=18 ; Prohibited

EDE=6 ; DNSSEC_Bogus
"EDETXT=signature too short"
```

5. Examples of EDNS(0) Presentation Format

The following examples shall illustrate the features of EDNS(0) Presentation format described above. They may not make really sense and should not appear in normal DNS operation.

```
. 0 IN EDNS0 (
    FLAGS=DO
    RCODE=BADCOOKIE
    UDPSIZE=1232
    EXPIRE=86400
    COOKIE=36714f2e8805a93d,4654b4ed3279001b
    EDE=18 ; Prohibited
    "EDETXT=bad cookie\000"
    OPT1234=000004d2
    PADDING=[113]
    )

. 0 IN EDNS0 ( FLAGS=0 RCODE=BADSIG UDPSIZE=4096 EXPIRE
               NSID=6578616d706c652e636f6d2e ; example.com.
               DAU=8,10 KEEPALIVE=60.0 CHAIN=zerobyte\000.com.
               KEYTAG=36651,6113 PADDING=df24d08b0258c7de )
```

6. Version-independent JSON representation

EDNS versions other than 0 are not yet specified, but an OPT pseudorecord with version field set to value other than zero might in theory appear in DNS messages. This section specifies how to represent such OPT pseudorecord in JSON. This procedure SHOULD NOT be used for EDNS(0). One possible exception is displaying a malformed EDNS(0) record.

The OPT pseudorecord is in this case represented in JSON as on object called EDNS with following members:

- *NAME - String with the Owner Name of the OPT record. Note that this is always . (DNS Root Domain Name) unless malformed. See Section 9 for representing DNS names in JSON.

- *TTL - Integer with the 32-bit big-endian value appearing at the TTL position of OPT pseudorecord Wire format, see [RFC6891], Section 6.1.3.

- *CLASS - Integer with the 16-bit value at the CLASS position of OPT pseudorecord Wire format (UDP payload size happens to appear there).

- *TYPE - Integer with the value 41. This member MAY be omitted.

- *RDATAHEX - String with the pseudorecord RDATA formatted as Base16.

Example:

```
"EDNS": {
    "NAME": ".",
    "TTL": 16859136,
    "CLASS": 1232,
    "RDATAHEX": "000f00020015"
}
```

7. EDNS(0) Representation in JSON

The EDNS(0) OPT record can be represented in JSON as an object called EDNS0. It MUST contain the three members (name-value pairs), Flags (Section 7.1), Extended RCODE (Section 7.2), and UDP Payload Size (Section 7.3). The rest of the members are based on Options in the OPT record [RFC6891], Section 6.1.2. For each member, its name and value are defined by this document, or by the specification of the respective EDNS Option. If it is not, a generic name and value from Section 7.4 applies. However, those generics MAY be used for any Option at all times. Note that the order of members is not preserved in JSON.

7.1. Flags

The JSON member name is FLAGS and its value is an Array of Strings BIT##, where ## is a Decimal value. BITn is present in the Array if and only if n-th bit (the most significant bit being 0-th) of flags is set to 1. If the Flag of the bit is specified in [IANA.EDNS.Flags], the Flag SHOULD be used instead of BIT##. (So far, the only known Flag is DO.)

7.2. Extended RCODE

The JSON member name is RCODE and its value is a String containing Field-value from Section 4.2.

7.3. UDP Payload Size

The JSON member name is UDPSIZE and its value is an Integer with UDP payload size.

7.4. Unrecognized Option

EDNS options that are not part of this specification and their own specifications do not specify their JSON member name and value MUST be displayed according this subsection. Other options (specified below or otherwise) MAY be displayed so as well.

Unrecognized option JSON member name is OPT##, where ## stands for its OPTION-CODE as Decimal value, and its value is a String containing its OPTION-VALUE encoded as Base16.

7.5. LLQ Option

The LLQ (OPTION-CODE 1 [RFC8764]) JSON member name is LLQ and its value is an Object with members LLQ-VERSION, LLQ-OPCODE, LLQ-ERROR, LLQ-ID, and LLQ-LEASE, each representing the respective value as Integer. Note that only numeric representation of these values is possible.

Example:

```
"LLQ": { "LLQ-VERSION": 1, "LLQ-OPCODE": 1, "LLQ-ERROR": 0,
         "LLQ-ID": 0, "LLQ-LEASE": 3600 }
```

7.6. NSID Option

The NSID (OPTION-CODE 3 [RFC5001]) JSON member name is NSIDHEX and its value is a String with OPTION-VALUE encoded as Base16.

Optionally, one more member of EDNS0 Object MAY be added as well, with the name NSID and the value being a String with the OPTION-

VALUE interpreted as UTF-8. Note that in that case, JSON escaping routines ([RFC8259], Section 7) take place, possibly using the \uXXXX notation.

7.7. DAU, DHU and N3U Options

The DAU, DHU, and N3U (OPTION-CODES 5, 6, 7, respectively [RFC6975]) JSON member names are DAU, DHU, and N3U, respectively, and their values are Arrays of Integers with ALG-CODEs.

Example:

```
"DAU": [ 8, 10, 13, 14, 15 ]
"DHU": [ 1, 2, 4 ]
"N3U": [ 1 ]
```

7.8. Edns-Client-Subnet Option

The EDNS Client Subnet (OPTION-CODE 8 [RFC7871]) JSON member name is ECS and its value is an Object with following members:

- *FAMILY - Integer with FAMILY
- *IP - String with the textual IPv4 or IPv6 address ([RFC1035], Section 3.4.1, [RFC4291], Section 2.2), or a String with ADDRESS encoded as Base16 if FAMILY is neither 1 or 2
- *SOURCE - Integer with SOURCE PREFIX-LENGTH
- *SCOPE - Integer with SCOPE PREFIX-LENGTH, omitted if zero

Examples:

```
"ECS": {
    "FAMILY": 1,
    "IP": "1.2.3.4",
    "SOURCE": 24
}

"ECS": {
    "FAMILY": 2,
    "IP": "1234::2",
    "SOURCE": 56,
    "SCOPE": 48
}
```

```
"ECS": {
    "FAMILY": 5,
    "IP": "0102030405060708"
    "SOURCE": 32
}
```

7.9. EDNS EXPIRE Option

The EDNS EXPIRE (OPTION-CODE 9 [RFC7314]) JSON member name is EXPIRE and its value is either an Integer or null.

7.10. Cookie Option

The DNS Cookie (OPTION-CODE 10 [RFC7873]) JSON member name is COOKIE and its value is an Array containing a String with the Client Cookie encoded as Base16 and, if present, another String with Server Cookie encoded as Base16.

7.11. Edns-Tcp-Keepalive Option

The edns-tcp-keepalive (OPTION-CODE 11 [RFC7828]) JSON member name is KEEPALIVE and its value is the TIMEOUT in seconds formatted as a Number [RFC8259], Section 6 (possibly a non-Integer).

7.12. Padding Option

The Padding (OPTION-CODE 12 [RFC7830]) JSON member name is PADDING and its value is a String containing Field-value from Section 4.12.

7.13. CHAIN Option

The CHAIN (OPTION-CODE 13 [RFC7901]) JSON member name is CHAIN and its value is a String with the OPTION-VALUE in the form of a textual Fully-Qualified Domain Name. See Section 9 for representing DNS names in JSON.

7.14. Edns-Key-Tag Option

The edns-key-tag (OPTION-CODE 14 [RFC8145], Section 4) JSON member name is KEYTAG and its value is an Array of Integers.

7.15. Extended DNS Error Option

The Extended DNS Error (OPTION-CODE 15 [RFC8914]) JSON member name is EDE and its value is an Object with following members:

*INFO-CODE - Integer with the INFO-CODE

*Purpose - String with Purpose of the INFO-CODE ([RFC8914], Section 5.2)

*EXTRA-TEXT - String with the EXTRA-TEXT

The EXTRA-TEXT member MUST be omitted if empty. If its value contains non-printable or special (backslash, quote) characters, they MUST be escaped by the means of JSON Strings ([RFC8259], Section 7).

8. Examples of EDNS(0) Representation in JSON

The following examples are the JSON representations of the examples in Section 5. They may not make really sense and should not appear in normal DNS operation.

```
"EDNS0": {
    "FLAGS": [ "DO" ],
    "RCODE": "BADCOOKIE",
    "UDPSIZE": 1232,
    "EXPIRE": 86400,
    "COOKIE": [ "36714f2e8805a93d", "4654b4ed3279001b" ],
    "EDE": {
        "INFO-CODE": 18,
        "Purpose": "Prohibited",
        "EXTRA-TEXT": "bad cookie\u0000"
    },
    "OPT1234": "000004d2",
    "PADDING": "[113]"
}
```

```
"EDNS0": { "FLAGS": [ ], "RCODE": "BADSIG", "UDPSIZE": 4096,
           "EXPIRE": null, "NSIDHEX": "6578616d706c652e636f6d2e",
           "NSID": "example.com.", "DAU": [ 8, 10 ], "KEEPALIVE": 60.0,
           "CHAIN": "zerobyte\\000.com.", "KEYTAG": [ 36651, 6113 ],
           "PADDING": "df24d08b0258c7de" }
```

9. Update Representing DNS Messages in JSON

This section is not related to EDNS. This section updates [RFC8427], Section 2.6, including erratum 5439, which introduced contradicting MUSTs for escaping of backslashes.

In order to solve this contradiction and correctly represent a DNS name in JSON, it MUST be first converted to textual Presentation format according to [RFC1035], Section 5.1 (called master file format in the referenced document), and the resulting <character-string> subsequently is inserted into JSON as String ([RFC8259], Section 7).

Note that the previous paragraph prescribes the following escaping strategy: In the first step every problematic character (non-printable, backslash, dot within Label, or any octet) is either

substituted with the sequence \DDD, where DDD is the three-digit decimal ASCII code, or in some cases (backslash, dot, any printable character) just prepended with a backslash. In the second step, every quote (") and backslash (\) in the resulting <character-string> is prepended with another backslash. Note that the JSON escaping sequence \uXXXX (where XXXX is a hexadecimal Unicode code) is thus never needed.

Moreover, following requirements from [RFC8427] still hold: The name MUST be represented as an absolute Fully-Qualified Domain Name. Internationalized Domain Name (IDN) labels MUST be expressed in their A-label form, as described in [RFC5890].

Example: the name with the Wire format 04005C2E2203646F6D00 can be represented in JSON as:

```
"NAME": "\\000\\\\\\046\".com."
```

but also as (among other ways):

```
"NAME": "\\000\\092\\.\\\".c\\om."
```

10. IANA Considerations

None.

11. Security Considerations

None.

12. Acknowledgements

TODO

13. Implementation Status

Note to the RFC Editor: please remove this entire appendix before publication.

None yet.

14. Change History

Note to the RFC Editor: please remove this entire appendix before publication.

*edns-presentation-format-00

Initial public draft.

15. References

15.1. Normative References

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

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

[RFC8259] Bray, T., Ed., "The JavaScript Object Notation (JSON) Data Interchange Format", STD 90, RFC 8259, DOI 10.17487/RFC8259, December 2017, <https://www.rfc-editor.org/info/rfc8259>.

[RFC8427] Hoffman, P., "Representing DNS Messages in JSON", RFC 8427, DOI 10.17487/RFC8427, July 2018, <https://www.rfc-editor.org/info/rfc8427>.

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

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

[RFC4648] Josefsson, S., "The Base16, Base32, and Base64 Data Encodings", RFC 4648, DOI 10.17487/RFC4648, October 2006, <https://www.rfc-editor.org/info/rfc4648>.

[RFC3597] Gustafsson, A., "Handling of Unknown DNS Resource Record (RR) Types", RFC 3597, DOI 10.17487/RFC3597, September 2003, <https://www.rfc-editor.org/info/rfc3597>.

[RFC8764] Cheshire, S. and M. Krochmal, "Apple's DNS Long-Lived Queries Protocol", RFC 8764, DOI 10.17487/RFC8764, June 2020, <https://www.rfc-editor.org/info/rfc8764>.

[RFC5001] Austein, R., "DNS Name Server Identifier (NSID) Option", RFC 5001, DOI 10.17487/RFC5001, August 2007, <https://www.rfc-editor.org/info/rfc5001>.

[RFC6975] Crocker, S. and S. Rose, "Signaling Cryptographic Algorithm Understanding in DNS Security Extensions

(DNSSEC)", RFC 6975, DOI 10.17487/RFC6975, July 2013, <https://www.rfc-editor.org/info/rfc6975>.

[RFC7871] Contavalli, C., van der Gaast, W., Lawrence, D., and W. Kumari, "Client Subnet in DNS Queries", RFC 7871, DOI 10.17487/RFC7871, May 2016, <https://www.rfc-editor.org/info/rfc7871>.

[RFC4291] Hinden, R. and S. Deering, "IP Version 6 Addressing Architecture", RFC 4291, DOI 10.17487/RFC4291, February 2006, <https://www.rfc-editor.org/info/rfc4291>.

[RFC7314] Andrews, M., "Extension Mechanisms for DNS (EDNS) EXPIRE Option", RFC 7314, DOI 10.17487/RFC7314, July 2014, <https://www.rfc-editor.org/info/rfc7314>.

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

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

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

[RFC7901] Wouters, P., "CHAIN Query Requests in DNS", RFC 7901, DOI 10.17487/RFC7901, June 2016, <https://www.rfc-editor.org/info/rfc7901>.

[RFC8145] Wessels, D., Kumari, W., and P. Hoffman, "Signaling Trust Anchor Knowledge in DNS Security Extensions (DNSSEC)", RFC 8145, DOI 10.17487/RFC8145, April 2017, <https://www.rfc-editor.org/info/rfc8145>.

[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>.

[RFC5890] Klensin, J., "Internationalized Domain Names for Applications (IDNA): Definitions and Document Framework", RFC 5890, DOI 10.17487/RFC5890, August 2010, <https://www.rfc-editor.org/info/rfc5890>.

15.2. Informative References

[IANA.EDNS.Flags]
"EDNS Header Flags", n.d., <https://www.iana.org/assignments/dns-parameters/dns-parameters.xhtml#dns-parameters-13>.

[IANA.RCODEs] "DNS RCODEs", n.d., <https://www.iana.org/assignments/dns-parameters/dns-parameters.xhtml#dns-parameters-6>.

[IANA.EDNS.EDE] "EDNS Extended Error Codes", n.d., <https://www.iana.org/assignments/dns-parameters/dns-parameters.xhtml#extended-dns-error-codes>.

[IANA.EDNS.DAU] "DNS Security Algorithm Numbers", n.d., <https://www.iana.org/assignments/dns-sec-alg-numbers/dns-sec-alg-numbers.xhtml>.

[IANA.EDNS.DHU] "DNSSEC DS RR Type Digest Algorithms", n.d., <https://www.iana.org/assignments/ds-rr-types/ds-rr-types.xhtml#ds-rr-types-1>.

[IANA.EDNS.N3U] "DNSSEC NSEC3 Hash Algorithms", n.d., <https://www.iana.org/assignments/dnssec-nsec3-parameters/dnssec-nsec3-parameters.xhtml#dnssec-nsec3-parameters-3>.

Authors' Addresses

Libor Peltan
CZ.NIC

Email: libor.peltan@nic.cz

Tom Carpay
NLnet Labs

Email: tom@nlnetlabs.nl