DNSEXT Working Group
Internet-Draft
Obsoletes: 2671, 2673
(if approved)
Intended status: Standards Track
Expires: July 4, 2013

J. Damas
Bond Internet Systems
M. Graff

P. Vixie
Internet Systems Consortium

December 31, 2012

Extension Mechanisms for DNS (EDNS(0))
draft-ietf-dnsext-rfc2671bis-edns0-10

Abstract

The Domain Name System's wire protocol includes a number of fixed fields whose range has been or soon will be exhausted and does not allow requestors to advertise their capabilities to responders. This document describes backward compatible mechanisms for allowing the protocol to grow.

This document updates the EDNS(0) specification (and obsoletes RFC 2671) based on feedback from deployment experience in several implementations. It also obsoletes RFC 2673 ("Binary Labels in the Domain Name System") and adds considerations on the use of extended labels in the DNS.

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 July 4, 2013.

Copyright Notice

Table of Contents

1. Introduction

DNS [RFC1035] specifies a Message Format and within such messages there are standard formats for encoding options, errors, and name compression. The maximum allowable size of a DNS Message over UDP not using the extensions described in this document is limited to 512 bytes. Many of DNS's protocol limits, such as the maximum message size over UDP, are too small to efficiently support the additional information that can be conveyed in the DNS (e.g. several IPv6 addresses or DNSSEC signatures). Finally, RFC 1035 does not define any way for implementations to advertise their capabilities to any of the other actors they interact with.

[RFC2671] added an extension mechanism to DNS. This mechanism is widely supported and a number of new DNS uses and protocol extensions depend on the presence of these extensions. This memo refines that specification and obsoletes [RFC2671].

Unextended agents will not know how to interpret the protocol extensions defined in [RFC2671] and restated here. Extended agents need to be prepared for handling the interactions with unextended clients in the face of new protocol elements, and fall back gracefully to unextended DNS.

EDNS is a hop-by-hop extension to DNS. This means the use of EDNS is negotiated between each pair of hosts in a DNS resolution process. For instance the stub resolver communicating with the recursive resolver or the recursive resolver communicating with an authoritative server.

[RFC2671] specified extended label types. The only such label proposed was in [RFC2673] for a label type called "Bitstring Labels." For various reasons introducing a new label type was found to be extremely difficult, and [RFC2673] was moved to Experimental. This document deprecates Extended Labels, and therefore Binary Labels, obsoleting [RFC2673].

2. Terminology

"Requestor" is the side which sends a request. "Responder" is an authoritative, recursive resolver, or other DNS component which responds to questions. Other terminology is used as per its use in the references.

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

3. EDNS Support Requirement

EDNS provides a mechanism to improve the scalability of DNS as its uses get more diverse on the Internet. It does this by enabling the use of UDP transport for DNS messages with sizes beyond the limits specified in RFC 1035 as well as providing extra data space for additional flags and return codes (RCODEs). However, implementation experiencing indicates that adding new RCODEs should be avoided due to the difficulty in upgrading the installed base. Flags SHOULD be used only when necessary for DNS resolution to function.

For many uses, a EDNS Option Code may be preferred.

With time, some applications of DNS have made EDNS a requirement for their deployment. For instance, DNSSEC uses the additional flag space introduced in EDNS to signal the request to include DNSSEC data in a DNS response.

Given the increase in DNS response sizes when including larger data items such as AAAA Records, DNSSEC information (e.g. RRSIG or DNSKEY) or large TXT Records, the additional UDP payload capabilities provided by EDNS can help improve the scalability of the DNS by avoiding widespread use of TCP for DNS transport.

4. DNS Message changes

4.1. Message Header

The DNS Message Header's second full 16-bit word is divided into a 4-bit OPCODE, a 4-bit RCODE, and a number of 1-bit flags (see , section 4.1.1 [RFC1035]). Some of these were marked for future use, and most these have since been allocated. Also, most of the RCODE values are now in use. The OPT pseudo-RR specified below contains extensions to the RCODE bit field as well as additional flag bits.

4.2. Label Types

The first two bits of a wire format domain label are used to denote the type of the label. [RFC1035] allocates two of the four possible types and reserves the other two. More label types were defined in [RFC2671]. This document obsoletes the use of the 2-bit combination defined by [RFC2671] to identify extended label types.

4.3. UDP Message Size

Traditional DNS Messages are limited to 512 octets in size when sent over UDP [RFC1035]. Fitting the increasing amounts of data that can

be transported in DNS in this 512-byte limit is becoming more difficult. For instance, inclusion of DNSSEC records frequently requires a much larger response than a 512 byte message can hold.

EDNS(0) is intended to provide support for transporting these larger packet sizes while continuing to use UDP. It specifies a way to advertise additional features such as larger response size capability, which is intended to help avoid truncated UDP responses which then cause retry over TCP.

5. Extended Label Types

The first octet in the on-the-wire representation of a DNS label specifies the label type; the basic DNS specification [RFC1035] dedicates the two most significant bits of that octet for this purpose.

[RFC2671] defined DNS label type 0b01 for use as an indication for Extended Label Types. A specific Extended Label Type was selected by the 6 least significant bits of the first octet. Thus, Extended Label Types were indicated by the values 64-127 (0b01xxxxxx) in the first octet of the label.

Extended Label Types are extremely difficult to deploy due to lack of support in clients and intermediate gateways as described in [RFC3363] which moved [RFC2673] to experimental status, and [RFC3364], which describes the pros and cons. As such proposals that contemplate extended labels SHOULD weigh this deployment cost against the possibility of implementing functionality in other ways.

Finally, implementations MUST NOT generate or pass Binary Labels in their communications as they are now deprecated.

6. The OPT pseudo-RR

6.1. OPT Record Definition

6.1.1. Basic elements

An OPT pseudo-RR (sometimes called a meta-RR) MAY be added to the additional data section of a request.

The OPT RR has RR type 41.

If present in requests, compliant responders MUST include an OPT record in their respective responses.

An OPT record does not carry any DNS data. It is used only to contain control information pertaining to the question and answer sequence of a specific transaction. OPT RRs MUST NOT be cached, forwarded, or stored in or loaded from master files.

The OPT RR MAY be placed anywhere within the additional data section. When an OPT RR MUST is included within any DNS message only ONE OPT RR can be present. If a query message with more than one OPT RR is received, a FORMERR (RCODE=1) MUST be returned. The placement flexibility for the OPT RR does not override the need for the TSIG or SIG(0) RRs to be the last in the additional section whenever they are present.

6.1.2. Wire Format

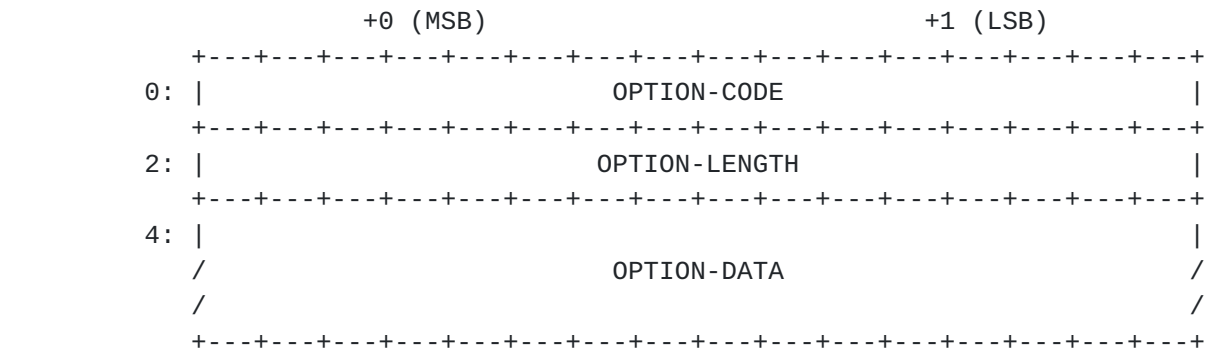
An OPT RR has a fixed part and a variable set of options expressed as {attribute, value} pairs. The fixed part holds some DNS meta data and also a small collection of basic extension elements which we expect to be so popular that it would be a waste of wire space to encode them as {attribute, value} pairs.

The fixed part of an OPT RR is structured as follows:

Field Name	Field Type	Description
NAME	domain name	MUST be 0 (root domain)
TYPE	u_int16_t	OPT (41)
CLASS	u_int16_t	requestor's UDP payload size
TTL	u_int32_t	extended RCODE and flags
RDLEN	u_int16_t	length of all RDATA
RDATA	octet stream	{attribute,value} pairs

OPT RR Format

The variable part of an OPT RR may contain zero or more options in the RDATA. Each option MUST be treated as a bit field. Each option is encoded as:



OPTION-CODE
Assigned by the Expert Review process as defined by the dnsext working group and the IESG.

OPTION-LENGTH
Size (in octets) of OPTION-DATA.

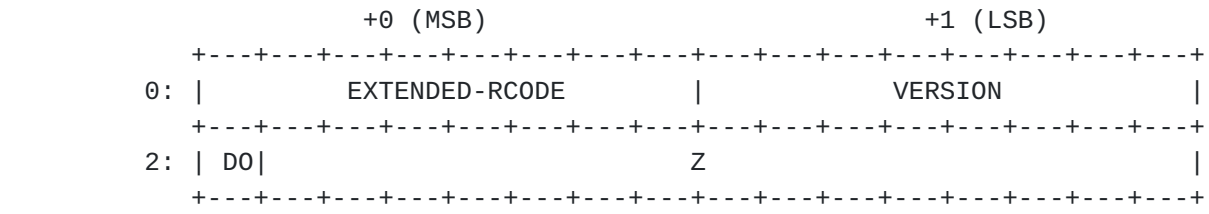
OPTION-DATA
Varies per OPTION-CODE. MUST be treated as a bit field.

The order of appearance of option tuples is not defined. If one option modifies the behavior of another or multiple options are related to one another in some way, they have the same effect regardless of ordering in the RDATA wire encoding.

Any OPTION-CODE values not understood by a responder or requestor MUST be ignored. Specifications of such options might wish to include some kind of signaled acknowledgement. For example, an option specification might say that if a responder sees and supports option XYZ, it MUST include option XYZ in its response.

6.1.3. OPT Record TTL Field Use

The extended RCODE and flags (which OPT stores in the RR TTL field) are structured as follows:



EXTENDED-RCODE
: Forms upper 8 bits of extended 12-bit RCODE (together with the 4 bits defined in [RFC1035]. Note that EXTENDED-RCODE value 0 indicates that an unextended RCODE is in use (values 0 through 15).

VERSION
: Indicates the implementation level of the setter. Full conformance with this specification is indicated by version ``0.'' Requestors are encouraged to set this to the lowest implemented level capable of expressing a transaction, to minimize the responder and network load of discovering the greatest common implementation level between requestor and responder. A requestor's version numbering strategy MAY ideally be a run time configuration option.
If a responder does not implement the VERSION level of the request, then it MUST respond with RCODE=BADVERS. All responses MUST be limited in format to the VERSION level of the request, but the VERSION of each response SHOULD be the highest implementation level of the responder. In this way a requestor will learn the implementation level of a responder as a side effect of every response, including error responses and including RCODE=BADVERS.

6.1.4. Flags

DO
: DNSSEC OK bit as defined by [RFC3225].

Z
: Set to zero by senders and ignored by receivers, unless modified in a subsequent specification.

6.2. Behaviour

6.2.1. Cache behaviour

The OPT record MUST NOT be cached.

6.2.2. Fallback

If a requestor detects that the remote end does not support EDNS(0), it MAY issue queries without an OPT record. It MAY cache this knowledge for a brief time in order to avoid fallback delays in the future. However, if DNSSEC or any future option using EDNS is required, no fallback should be performed as they are only signalled through EDNS. If an implementation detects that some servers for the zone support EDNS(0) while others would force the use of TCP to fetch

all data, preference MAY be given to those which support EDNS(0). Implementers SHOULD analyse this choice and the impact on both endpoints.

6.2.3. Requestor's Payload Size

The requestor's UDP payload size (encoded in the RR CLASS field) is the number of octets of the largest UDP payload that can be reassembled and delivered in the requestor's network stack. Note that path MTU, with or without fragmentation, could be smaller than this.

Values lower than 512 MUST be treated as equal to 512.

The requestor SHOULD place a value in this field that it can actually receive. For example, if a requestor sits behind a firewall which will block fragmented IP packets, a requestor SHOULD NOT choose a value which will cause fragmentation. Doing so will prevent large responses from being received, and can cause fallback to occur. This knowledge may be auto-detected by the implementation or provided by a human administrator.

Note that a 512-octet UDP payload requires a 576-octet IP reassembly buffer. Choosing between 1280 and 1410 bytes for IP (v4 or v6) over Ethernet would be reasonable.

Bigger values SHOULD be considered by implementers to be used where fragmentation is not a concern. Implementations SHOULD use their largest configured or implemented values as a starting point in an EDNS transaction in the absence of previous knowledge about the destination server.

Choosing a very large value will guarantee fragmentation at the IP layer, and may prevent answers from being received due to a single fragment loss or misconfigured firewalls.

The requestor's maximum payload size can change over time. It MUST NOT be cached for use beyond the transaction in which it is advertised.

6.2.4. Responder's Payload Size

The responder's maximum payload size can change over time, but can be reasonably expected to remain constant between two closely spaced sequential transactions; for example, an arbitrary QUERY used as a probe to discover a responder's maximum UDP payload size, followed immediately by an UPDATE which takes advantage of this size. This is considered preferable to the outright use of TCP for oversized

requests, if there is any reason to suspect that the responder implements EDNS, and if a request will not fit in the default 512 payload size limit.

6.2.5. Payload Size Selection

Due to transaction overhead, it is not recommended to advertise an architectural limit as a maximum UDP payload size. Even on system stacks capable of reassembling 64KB datagrams, memory usage at low levels in the system will be a concern. A good compromise may be the use of a EDNS maximum payload size of 4096 octets as a starting point.

A requestor MAY choose to implement a fallback to smaller advertised sizes to work around firewall or other network limitations. A requestor SHOULD choose to use a fallback mechanism which begins with a large size, such as 4096. If that fails, a fallback around the 1280-1410 byte range SHOULD be tried, as it has a reasonable chance to fit within a single Ethernet frame. Failing that, a requestor MAY choose a 512 byte packet, which with large answers may cause a TCP retry.

Values of less than 512 bytes MUST be treated as equal to 512 bytes.

6.2.6. Support in MiddleBoxes

In a network that carries DNS traffic there could be active equipment other than that participating directly in the DNS resolution process (stub and caching resolvers, authoritative servers) that affect the transmission of DNS messages (e.g. firewalls, load balancers, proxies, etc) referred to here as MiddleBoxes.

Conformant MiddleBoxes MUST NOT limit DNS messages over UDP to 512 bytes.

MiddleBoxes which simply forward requests to a recursive resolver MUST NOT modify and MUST NOT delete the OPT record contents in either direction.

MiddleBoxes which have additional functionality, such as answering queries or acting as intelligent forwarders, SHOULD be able to process the OPT record and act based on its contents. These boxes MUST consider the incoming request and any outgoing requests as separate transactions if the characteristics of the messages are different.

A more in depth discussion of this type of equipment and other considerations regarding their interaction with DNS traffic is found

in [RFC5625]

7. Transport Considerations

The presence of an OPT pseudo-RR in a request should be taken as an indication that the requestor fully implements the given version of EDNS, and can correctly understand any response that conforms to that feature's specification.

Lack of presence of an OPT record in a request MUST be taken as an indication that the requestor does not implement any part of this specification and that the responder MUST NOT include an OPT record in its response.

Extended agents MUST be prepared for handling the interactions with unextended clients in the face of new protocol elements, and fall back gracefully to unextended DNS when needed.

Responders which choose not to implement the protocol extensions defined in this document MUST respond with a return code (RCODE) of FORMERR to messages containing an OPT RR in the additional section and MUST NOT include an OPT record in the response.

If there is a problem with processing the OPT record itself, such as an option value that is badly formatted or includes out of range values, a FORMERR MUST be returned. If this occurs the response MUST include an OPT record. This is intended to allow the requestor to distinguish between servers which do not implement EDNS and format errors within EDNS.

The minimal response MUST be the DNS header, question section, and an OPT record. This MUST also occur when an truncated response (using the DNS header's TC bit) is returned.

8. Security Considerations

Requestor-side specification of the maximum buffer size may open a DNS denial of service attack if responders can be made to send messages which are too large for intermediate gateways to forward, thus leading to potential ICMP storms between gateways and responders.

Announcing very large UDP buffer sizes may result in dropping by middleboxes (see Section 6.2.6). This could cause retransmissions with no hope of success. Some devices have been found to reject fragmented UDP packets.

Announcing too small UDP buffer sizes may result in fallback to TCP with a corresponding load impact on DNS servers. This is especially important with DNSSEC, where answers are much larger.

9. IANA Considerations

The IANA has assigned RR type code 41 for OPT.

[RFC2671] specified a number of IANA sub-registries within "DOMAIN NAME SYSTEM PARAMETERS:"

- DNS EDNS(0) Options
- EDNS Version Number
- EDNS Header Flags

Additionally, two entries were generated in existing registries:

EDNS Extended Label Type in the DNS Label Types Registry

Bad OPT Version in the DNS RCODES registry

IANA is advised to update references to [RFC2671] in these entries and sub-registries to this document.

[RFC2671] created the "EDNS Extended Label Type Registry". We request that this registry be closed.

This document assigns option code 65535 in the "EDNS Option Codes" registry to "Reserved for future expansion."

[RFC2671] expands the RCODE space from 4 bits to 12 bits. This allows more than the 16 distinct RCODE values allowed in [RFC1035]. IETF Standards Action is required to add a new RCODE.

This document assigns EDNS Extended RCODE 16 to "BADVERS" in the DNS RCODES registry.

[RFC2671] called for the recording of assignment of extended label types 0bxx111111 as "Reserved for future extended label types". This request implied a request to open a new registry for Extended Label Types but due to the possibility of ambiguity new text registrations were instead made within the general "DNS Label Types" registry which also registers entries originally defined by [RFC1035].

This document requests IANA to close registration of further Extended

Label Types in the "DNS Label Types" Registry.

IETF Standards Action is required for assignments of new EDNS(0) flags. Flags SHOULD be used only when necessary for DNS resolution to function. For many uses, a EDNS Option Code may be preferred.

IETF Standards Action is required to create new entries in the EDNS Version Number registry. Within the EDNS Option Code space Expert Review is required for allocation of an EDNS Option Code. IANA is requested to keep a registry for the EDNS Option Code space.

9.1. OPT Option Code Allocation Procedure

OPT Option Codes are assigned by expert review.

Assignment of Option Codes should be liberal, but duplicate functionality is to be avoided.

10. References

10.1. Normative References

[RFC1035] Mockapetris, P., "Domain names - implementation and specification", STD 13, RFC 1035, November 1987.

[RFC2671] Vixie, P., "Extension Mechanisms for DNS (EDNS0)", RFC 2671, August 1999.

[RFC3225] Conrad, D., "Indicating Resolver Support of DNSSEC", RFC 3225, December 2001.

10.2. Informative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC2673] Crawford, M., "Binary Labels in the Domain Name System", RFC 2673, August 1999.

[RFC3363] Bush, R., Durand, A., Fink, B., Gudmundsson, O., and T. Hain, "Representing Internet Protocol version 6 (IPv6) Addresses in the Domain Name System (DNS)", RFC 3363, August 2002.

[RFC3364] Austein, R., "Tradeoffs in Domain Name System (DNS) Support for Internet Protocol version 6 (IPv6)", RFC 3364, August 2002.

[RFC5625] Bellis, R., "DNS Proxy Implementation Guidelines", BCP 152, RFC 5625, August 2009.

Appendix A. Document Editing History

Following is a list of high-level changes made to the original RFC2671.

A.1. Changes since RFC2671

- Support for the OPT record is now mandatory.
- Extended label types obsoleted and the registry is closed.
- The bitstring label type, which was already moved from draft to experimental, is requested to be moved to historical.
- Changes in how EDNS buffer sizes are selected, with recommendations on how to select them.
- Front material (IPR notice and such) was updated to current requirements.

A.2. Changes since -02

- Specified the method for allocation of constants.
- Cleaned up a lot of wording, along with quite a bit of document structure changes.

Authors' Addresses

Joao Damas
Bond Internet Systems
Av Albufera 14
S.S. Reyes, Madrid 28701
ES

Phone: +1 650.423.1312
Email: joao@bondis.org

Michael Graff

Email: explorer@flame.org

Paul Vixie
Internet Systems Consortium
950 Charter Street
Redwood City, California 94063
US

Phone: +1 650.423.1301
Email: vixie@isc.org