DNSIND Working Group
INTERNET-DRAFT
<draft-ietf-dnsind-edns-03.txt>

Paul Vixie
ISC

August, 1998

Extensions to DNS (EDNS)

Status of this Memo

Abstract

The Domain Name System's wire protocol includes a number of fixed fields whose range has been or soon will be exhausted, does not allow clients to advertise their capabilities to servers, and does not support the use of multiple questions per request. This document describes backward compatible mechanisms for allowing the protocol to grow.

1 - Rationale and Scope

1.1. DNS (see [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 is fixed. Many of DNS's protocol limits are too small for uses which are or which are desired to become common. There is no way for clients to advertise their capabilities to servers, and it is not possible to ask multiple questions in a single request.

1.2. Existing clients will not know how to interpret the protocol extensions detailed here. In practice, these clients will be upgraded when they have need of a new feature, and only new features will make use of the extensions. We must however take account of client behaviour in the face of extra fields, and design a fallback scheme for interoperability with these clients.

2 - Affected Protocol Elements

2.1. The DNS Message Header's (see [RFC1035 4.1.1]) second full 16-bit word is divided into a 4-bit OPCODE, a 4-bit RCODE, and a number of 1-bit flags. The original reserved Z bits have been allocated to various purposes, and most of the RCODE values are now in use. More types and more possible RCODEs are needed.

2.2. The first two bits of a wire format domain label are used to denote the type of the label. [RFC1035 4.1.4] allocates two of the four possible types and reserves the other two. Proposals for use of the remaining types far outnumber those available. More label types are needed.

2.3. Compression pointers are 14 bits in size and are relative to the start of the DNS Message, which can be 64KB in length. 14 bits restrict pointers to the first 16KB of the message, which makes labels introduced in the last 48KB of the message unreachable by compression pointers. A longer pointer format is needed.

2.4. DNS Messages are limited to 512 octets in size when sent over UDP. While the minimum maximum reassembly buffer size is still 512 bytes, most of the hosts now connected to the Internet are able to reassemble larger datagrams. Some mechanism must be created to allow requestors to advertise larger buffer sizes to responders.

2.5. DNS Messages are limited to 65535 octets in size when sent over TCP. This acts as an effective maximum on RRset size, since multiple TCP messages are only possible in the case of zone transfers. Some mechanism must be created to allow normal DNS responses (other than zone transfers) to span multiple DNS Messages when TCP is used.

2.6. Multiple queries in a question section have not been supported in DNS due the applicability of some DNS Message Header flags (such as AA) and of the RCODE field only to a single QNAME, QTYPE, and QCLASS. Multiple questions per request are desirable, and some way of asking them must be made available.

3 - Extended Label Types

3.1. The ``1 0'' label type will now indicate an extended label type, whose value is encoded in the lower six bits of the first octet of a label. All subsequently developed label types should be encoded using an extended label type.

3.2. The ``0 0 0 0 0 0'' extended label type will indicate an extended compression pointer, such that the following two octets comprise a 16-bit compression pointer in network byte order. Like the normal compression pointer, this pointer is relative to the start of the DNS Message.

3.3. The ``0 0 0 0 0 1'' extended label type will indicate a counted bit string label with interior longest-match query matching semantics as described in [CRAW98].

3.4. The ``0 0 0 0 1 0'' extended label type will indicate a ``long local compression pointer'' as described in [KOCH98].

3.5. The ``1 1 1 1 1 1'' extended label type will be reserved for future expansion of the extended label type code space.

4 - OPT pseudo-RR

4.1. The OPT pseudo-RR can be added to the additional data section of either a request or a response. An OPT is called a pseudo-RR because it pertains to a particular transport level message and not to any actual DNS data. OPT RRs shall never be cached, forwarded, or stored in or loaded from master files.

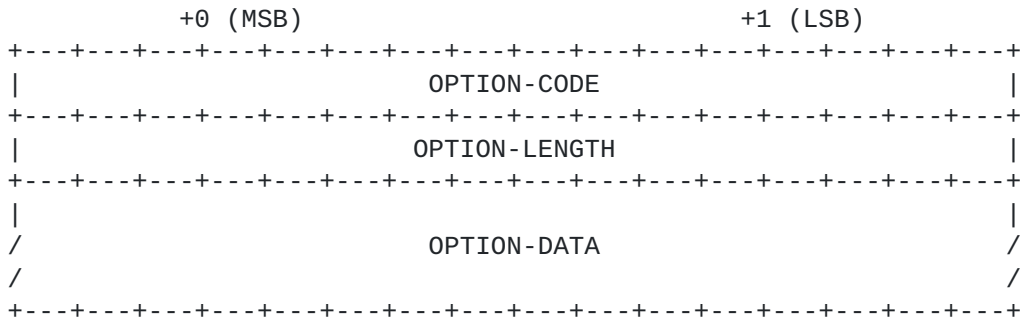
4.2. An OPT RR has a fixed part and a variable set of options expressed as {attribute, value} pairs. The fixed part holds some DNS meta data and also a small collection of new protocol elements which we expect to be so popular that it would be a waste of wire space to encode them as {attribute, value}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

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

Field Name	Field Type	Description
NAME	domain name	empty (root domain)
TYPE	u_int16_t	OPT (XXX IANA)
CLASS	u_int16_t	sender's UDP buffer size

TTL	u_int32_t	extended RCODE and flags
RDLEN	u_int16_t	describes RDATA
RDATA	octet stream	{attribute,value} pairs

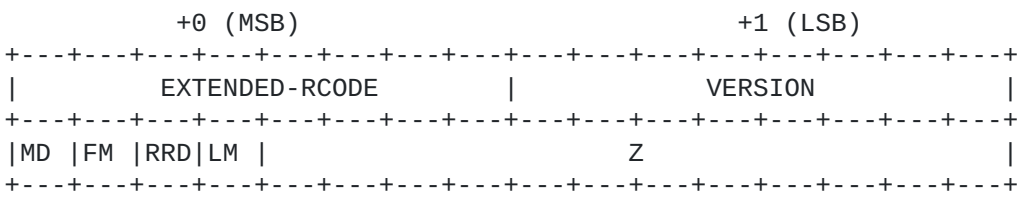
4.4. The variable part of an OPT RR is encoded in its RDATA and is structured as zero or more of the following:



OPTION-CODE	Assigned by the IANA. Value 65535 is reserved for future expansion.
OPTION-LENGTH	Size (in octets) of OPTION-DATA.
OPTION-DATA	Varies per OPTION-CODE.

4.5. The sender's UDP buffer size is the number of octets of the largest UDP payload that can be reassembled and delivered in the sender's network stack. Note that path MTU, with or without fragmentation, may be smaller than this. Also note that a 512-octet UDP payload requires a 576-octet IP reassembly buffer. Choosing 1436 on an Ethernet connected requestor would be reasonable. The consequence of choosing too large a value may be an ICMP message from an intermediate gateway, or even a silent drop of the response message. Requestors are advised to retry in such cases. Both requestors and responders are advised to take account of the path's MTU when considering message sizes.

4.6. The extended RCODE and flags are structured as follows:



EXTENDED-RCODE Forms upper 8 bits of extended 12-bit RCODE. (Meaningless in requests.)

VERSION Indicates the implementation level of whoever sets it. Full conformance with the draft standard version of this specification is version ``0.'' Note that both requestors and responders should set this to the highest level they implement, that responders should send back RCODE=BADVERS (XXX IANA) and that requestors should be prepared to probe using lower version numbers if they receive an RCODE=BADVERS.

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

MD ``More data'' flag. Valid only in TCP streams where message ordering and reliability are guaranteed. This flag indicates that the current message is not the complete request or response, and should be aggregated with the following message(s) before being considered complete. Such messages are called ``segmented.'' It is an error for the RCODE (including the EXTENDED-RCODE), AA flag, or DNS Message ID to differ among segments of a segmented message. It is an error for TC to be set on any message of a segmented message. Any given RR must fit completely within a message, and all messages will both begin and end on RR boundaries.

FM ``First match'' flag. Notable only when multiple questions are present. If set in a request, questions will be processed in wire order and the first question whose answer would be NOERROR AND ANCOUNT>0 is treated as if it were the only question in the query message. Otherwise, questions can be processed in any order and all possible answer records will be included in the

response. FM should be set to zero in responses and ignored by requestors.

RRD ``Recursion really desired'' flag. Notable only when a request is processed by an intermediate name server (``forwarder'') who is not authoritative for the zone containing QNAME, and where QTYPE=ANY or QDCOUNT>1. If set in a request, the intermediate name server can only answer using unexpired cached answers (either positive or negative) which were atomically acquired using the same QTYPE or set of QTYPEs present in the current question and where all such answers had the same TTL when first cached.

LM ``Longest match'' flag. If this flag is present in a query message, then for any question whose QNAME is not fully matched by zone or cache data, the longest trailing suffix of the QNAME for which zone or cache data is present will be eligible for use as an answer. Any of: QTYPE=ANY, or QCLASS=ANY, or QCOUNT>1, shall be considered an error if the LM flag is set.

5 - Multiple Questions for QUERY

5.1. If QDCOUNT>1, multiple questions are present. All questions must be for the same QNAME and QCLASS; only the QTYPE is allowed to vary. It is an error for QDCOUNT>1 and any QTYPE=ANY or QCLASS=ANY.

5.2. RCODE and AA apply to all RRs in the answer section having the QNAME that is shared by all questions in the question section. AA applies to all matching answers, and will not be set unless the exact original request was processed by an authoritative server and the response forwarded in its entirety if at all, or unless iterative requests are used as described in [5.4] below.

5.3. If a multiple question request is processed by an intermediate server and the authority server does not support multiple questions, the intermediate server must generate an answer iteratively by making multiple requests of the authority server. In this case, AA must never be set in the final answer due to lack of atomicity of the contributing authoritative responses.

5.4. If iteratively processing a multiple question request using an authority server which can only process single question requests, if any contributing request generates a SERVFAIL response, then the final

response's RCODE should be SERVFAIL.

6 - Transport Considerations

6.1. The presence of an OPT pseudo-RR, or any new label type, or QDCOUNT>1 in a request should be taken as an indication that the requestor fully implements that feature of this specification, and can correctly understand any response that conforms to that feature's specification. If a new label type or QDCOUNT>1 is used in a message that does not have an OPT RR, a VERSION of ``0'' shall be imputed, for the purpose of either interpreting the request or defining conformance of the response.

6.2. Lack of use of these features in a request must be taken as an indication that the requestor does not implement any part of this specification and that the responder may make no use of any protocol extension described here in its response.

6.3. Responders who do not understand these protocol extensions are expected to send a respose with RCODE NOTIMPL, FORMERR, or SERVFAIL. Therefore use of extensions should be ``probed'' such that a responder who isn't known to support them be allowed a retry with no extensions if it responds with one of the above mentioned RCODEs. If a responder's capability is cached by requestors, a new probe should be sent periodically to test for upgrades to responder capability.

7 - Security Considerations

Requestor-side specification of the maximum buffer size may open a new 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.

8 - Acknowledgements

Paul Mockapetris, Mark Andrews, Robert Elz, Don Lewis, Bob Halley, and Donald Eastlake were each instrumental in creating this specification.

9 - References

[RFC1035] P. Mockapetris, ``Domain Names - Implementation and Specification,'' RFC 1035, USC/Information Sciences Institute, November 1987.

[CRAW98] M. Crawford, ``Binary Labels in the Domain Name System,'' Draft draft-ietf-dnsind-binary-labels-XX, IETF DNSIND, March 1998.

[KOCH98] P. Koch, ``A New Scheme for the Compression of Domain Names,'' Draft draft-ietf-dnsind-local-compression-XX.txt. IETF DNSIND, March 1998.

10 - Author's Address

Paul Vixie
Vixie Enterprises
950 Charter Street
Redwood City, CA 94063
+1 650 779 7001
<paul@vix.com>