Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: March 8, 2017

J. Levine
Taughannock Networks
P. Vixie

September 4, 2016

An Extension Language for the DNS
draft-levine-dnsextlang-08

Abstract

Adding new RRTYPEs to the DNS has required that DNS servers and provisioning software be upgraded to support each new RRTYPE in Master files. This document defines a DNS extension language intended to allow most new RRTYPEs to be supported by adding entries to configuration data read by the DNS software, with no software changes needed for each RRTYPE.

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 March 8, 2017.

Copyright Notice

Table of Contents

1. Introduction

The Domain Name System[RFC1034] [RFC1035] is designed to be extensible, with new record types, known as RRTYPEs, added as needed. While it is straightforward in principle to add a new RRTYPE, in practice it can be difficult due to the software changes needed to add the new RRTYPE to the master file format read by many authoritative DNS servers, and to the provisioning software used to create and update the master files or the local equivalent.

While some new RRTYPEs, notably those for DNSSEC [RFC4033], require that DNS servers do new special purpose processing, most new RRTYPEs are, from the point of view of the DNS, just static data to return to queries, perhaps with some additional section records if the record includes another domain name. This document defines an extension language to describe any RRTYPEs, so that provisioning software can parse master file records for the RRTYPEs. DNS servers can use the

extension language to implement RRTYPEs that do not require special purpose processing.

2. Typical usage

The extension language is written as strings of UTF-8 text that describe new RR types, intended to be stored in the DNS itself. (They may also be stored in a local file with a well-known name, for debugging and local overrides, but this usage is optional.) All of the DNS software that needs to handle master file records fetches records from the DNS as needed. To support a new RRTYPE, one would add suitable records to the DNS zone where the descriptions are located, or to the local file.

DNS servers can use the extension language to parse new RRTYPE records in master files, and to translate them to the binary representation. Servers that create ASCII master files from zone data retrieved via AXFR can use the extension language to create master file records for new RRTYPEs.

Provisioning software can use the extension language to create templates for users to fill in, to create new RRTYPE records in master files to be passed to DNS servers, and to syntax check records entered by users. The extension language includes natural language field descriptions intended to be used as prompts in fill-in templates, and can handle versions of prompts in multiple languages.

Provisioning software could create TYPEnn master records if the local DNS server doesn't implement the extension language, although it would be less confusing if both provisioning and server software both accept the same master record syntax.

Some DNS servers store records in ways other than master files, such as SQL databases. The extension language could be used to create new schema entries to handle new RRTYPEs, although the details are too specific to particular varieties of DNS server software for this document to try to describe the details.

The extension language can describe all existing RRTYPEs, which may be useful in table driven provisioning software.

3. Extension language syntax

3.1. Lexical structure

The extension language consists of "stanzas", each of which defines an RRTYPE. In the DNS, a stanza is stored as a multi-string TXT record, with each string conceptually being a line in the stanza. In

a file, it is stored as a series of lines. The first line of a stanza defines the symbolic RRTYPE name. Subsequent lines, which must start with white space, each define a field in the record. Blank lines and comment lines where the first nonblank character is "#" are ignored.

The following ABNF imports ALPHA, DIGIT, and WSP from [RFC5234].

```
ldh = ALPHA 0*(ALPHA | DIGIT | "-")

dnsextfile = 1*stanza

stanza = rrtypeline 1*fieldline

rrtypeline = ldh ":" 1*DIGIT 0*1(":" 1*ALPHA) 0*1(WSP freetext)

fieldline = ftype 0*1qualifiers 0*1(":" ldh ) 0*1(WSP freetext)

ftype = "I1" | "I2" | "I4" | "A" | "AA" | "AAAA" | "N" | "S" |
  "B32" | "B64" | "X" | "T" | "T6"

qualifiers = "[" qual 0*(, qual) "]"

qual = ldh "=" 1*DIGIT | "C" | "A" | "L" | "M" | "X"

freetext = 0*(%x20-%xfe)
```

3.2. Storage in the DNS

Each extension language stanza stored in the DNS is stored as two identical TXT records, one with a name based on the numeric RR type, one with a name based on the text name. (One record may be aliased to the other using a CNAME.) The numeric names are located at RRTYPE.ARPA, and the text names are located at RRNAME.ARPA.

The first two strings in the TXT record are the identification tag "RRTYPE=1" to identify the record as an RRTYPE definition, and a language tag [RFC5646] that identifies the language in which the descriptive text is written. Each line of the stanza is a string in the TXT records. The leading spaces used in the file format (described below) are not used. For example, if the FOO record type were type 999, the two records for an English language description would be:

```
999.RRTYPE.ARPA. TXT "RRTYPE=1" "EN" "FOO:999 Foo record" "I2:count Count"
"..."
FOO.RRNAME.ARPA. TXT "RRTYPE=1" "EN" "FOO:999 Foo record" "I2:count Count"
"..."
```

If there are descriptions in multiple languages, they are all stored at the same name, and applications can choose the most suitable one.

```
999.RRTYPE.ARPA. TXT "RRTYPE=1" "EN" "FOO:999 Foo record" "I2:count Count"
"..."
999.RRTYPE.ARPA. TXT "RRTYPE=1" "FR" "FOO:999 Dossier foo" "I2:count Compte"
"..."
FOO.RRNAME.ARPA. TXT "RRTYPE=1" "EN" "FOO:999 Foo record" "I2:count Count"
"..."
FOO.RRNAME.ARPA. TXT "RRTYPE=1" "FR" "FOO:999 Dossier foo" "I2:count Compte"
"..."
```

3.3. Storage in a file

All the extension language stanzas stored in a file are stored as lines of ASCII text. The name of the RR type starts in the first position of the first line in the stanza. Subsequent lines in the stanza start with white space. A line that is blank or where the first nonblank character is a # is a comment and is ignored.

Descriptions in different languages are stored in separate files.

3.4. Stanza structure

Each stanza starts with a line containing the name of the RRTYPE, a colon, and the numeric RRTYPE. The name of the RRTYPE must start in the first position on the line. When stored in a file, the RRTYPE name should not be the same as an existing RRTYPE or DNS class name (IN or CH) or bad things will happen.

The RRTYPE may be followed a colon and letters, to indicate options for the RRTYPE. The only currently used letter is "X" which means that implementing the RRTYPE requires extra processing by DNS servers, e.g., the extra processing for DNAME or DNSSEC records. The intention of the option is to allow DNS servers to report an error if a zone contains a record defined with "X" for which the server does not implement the extra processing.

That can be followed by white space and a descriptive comment intended to be displayed to human users, but not interpreted by DNS software. Provisioning software might use the comments as prompts or labels to help a user select the desired RRTYPE.

The rest of the lines in the stanza describe the fields in the record. Each field is one or more octets long, and fields are stored sequentially in the record:

```
FOO:999 Foo record
   field description
   field:tag description
   field[qual,qual] description
   field[qual,qual]:tag description
   field ...
```

Some fields may be followed by a comma-separated list of qualifiers in square brackets. The qualifiers further define the field, e.g., in a numeric field, the qualifiers may define symbolic names for field values or bit masks. That can be followed by an colon and an ldh string. The string is intended to be used as the name of the field in software applications that create data structures for an RRTYPE. Applications will often have to change the punctuation to match the syntax of the programming language, such as replacing hyphens with underscores. If two fields in an RRTYPE have the same name, the result is undefined.

The field and optional qualifiers and name may be followed by white space and a description of the field. The description is intended to be displayed to human users, and is not interpreted by DNS software. Provisioning software might use the comments as prompts or labels for templates into which users enter RR data.

3.5. Field types

Each field type is defined by a token name consisting of letters and digits, starting with a letter.

3.5.1. Integer fields

Integer fields are defined by I1, I2, and I4 tokens, for fields one, two, or four octets long. The corresponding value in a master record is an unsigned integer number. A field may be followed by qualifiers defining symbolic field values.

A symbolic field value is represented as NAME=NN where NAME is the symbol and NN is the numeric value to be placed in the field. The corresponding value in a master record is the symbol. The symbol can contain letters, digits, and hypens. For example, to define the type field in a CERT record [RFC4398]:

```
I2[PKIX=1,SPKI=2,PGP=2,IPKIX=4,ISPKI=5,IPGP=6,ACPKIX=7,\
 IACPKIX=8,URI=253,OID=254]:type Certificate type
```

RRTYPE fields are defined by R tokens, for a two octet field containing an RRTYPE. The corresponding value in a master record is a symbolic RRTYPE or TYPEnnn for types without names.

3.5.2. IP address and partial address fields

IP address fields are defined by A or AAAA tokens, for four-octet IPv4 addresses or 16-octet IPv6 addresses. The corresponding value in a master record is an IP address written in the usual way. There are no qualifiers.

The AA token defines a 64 bit field written like half of an IPv6 address, with up to four colon separated groups of up to four hex digits.

3.5.3. Domain name fields

Domain name fields are defined by N tokens. The qualifier C means the name is compressed. The qualifier M, which can only appear on the last field in a record, means there can be an arbitrary number of domain names. The qualifier A means that the domain name represents a mailbox, with the first component being the local part of the mailbox. The qualifier L means that the domain name is converted to lower case before DNSSEC validation.

The corresponding value in a master record is a domain name or list of domain names, written in the usual way, with \. meaning a literal dot in a record.

Names are absolute if they end with a dot, otherwise relative to $ORIGIN, the convention for master files.

3.5.4. String fields

String fields are defined by S tokens. The qualifier M means that there may be multiple strings, each stored as a string in the record. A string field with the M qualifier must be the last field in the record.

The corresponding value in a master record is a string enclosed in single or double quotes, or multiple strings if the M qualifier is present. Embedded quotes may be escaped with a backslash, and a double backslash represents a backslash. If a non-null string contains no white space, quote characters, or backslashes, the quotes may be omitted.

A string with the X qualifier is a raw string, stored without any length bytes. It must be the last field in the record.

3.5.5. Base-32 and Base-64 fields

A base32 or base64 field is defined by a B32 or B64 token. The qualifier C means that the field is stored in the record as a string with a preceding length byte. The qualifier S means that the field is stored in the record as a string with a preceding two-byte length field. A base32 or base64 field without a C or S qualifier must be the last field in the record.

The corresponding value in a master record is a string represented as base64 [RFC3548]. The value of a base64 field without a C qualifier may include embedded spaces for readability, which are ignored.

3.5.6. Hex fields

A hex field is defined by an X token. The qualifier C means that the field is stored in the record as a string with a preceding length byte. The qualifier S means that the field is stored in the record as a string with a preceding two-byte length field. An unqualified hex field must be the last field in the record.

The corresponding value in a master record is a string represented as an even number of hexadecimal digits. The value may include embedded spaces for readability, which are ignored.

EUI48 and EUI64 fields are defined by X6 and X8 tokens, respectively. The corresponding fields in master records are six or eight pairs of hex digits separated by hyphens.

3.5.7. Time stamp fields

A 32-bit timestamp field is defined by a T token. The corresponding value in a master record is a fourteen digit value in the form YYYYMMDDHHmmSS indicating a UTC timestamp. The field is stored in the record as a Unix timestamp, the unsigned number of seconds since January 1, 1970 00:00:00 UTC.

A 48-bit timestamp field is defined by a T6 token. The corresponding value in a master record is an integer value representing the number of seconds since January 1, 1970 00:00:00 UTC. The field is stored in the record as a six octet binary version of that value.

4. Examples

If a DNS server didn't already have support for MX records, they could be defined as:

```
 MX:15 Mail exchanger
   I2 Priority (lower values are higher priority)
   N[A,C] Host name
```

The name is MX, the RRTYPE is 15, and the data includes a two-octet number and a compressed domain name, with additional section records for the domain name.

The SRV record [RFC2782] could be defined as:

```
 SRV:33 Service location
   I2 Priority
   I2 Weight
   I2 Port
   N[A] Target host name
```

The name is SRV, the RRTYPE is 33. The record contains three two-octet fields for the priority, weight, and port, and a domain name. The domain name is not compressed, but the DNS server should include additional section records for it.

5. Security considerations

The extension language makes it possible to create master files that represent arbitrary DNS records. Since most DNS servers already provide ways to represent arbitrary data, this doesn't introduce any new security issues to the DNS and DNS servers, although it may create security issues in provisioning software if the provisioning system is intended to limit the kinds of records its users can define.

Extension language files with accidentally or deliberately invalid field definitions could provoke odd bugs in server or provisioning software that doesn't check the syntax before using it.

When extension language data are imported from the DNS, a hostile party might use DNS spoofing techniques to modify the records imported. Methods to defend against DNS spoofing include DNSSEC.

6. IANA considerations

This document requests that IANA create the RRTYPE.ARPA and RRNAME.ARPA zones. Their initial contents are as follows: [list of description of existing RRs here]

When new RR types are defined, the defining documents SHOULD request IANA to add appropriate records to RRTYPE.ARPA and RRNAME.ARPA.

This document requests that IANA create a registry of DNS Extension Language Field Types. Its initial contents are as follows

TYPE	REFERENCE	EXTLANG VERSION
I1	(this document)	1
I2	(this document)	1
I4	(this document)	1
A	(this document)	1
AA	(this document)	1
AAAA	(this document)	1
N	(this document)	1
S	(this document)	1
B32	(this document)	1
B64	(this document)	1
X	(this document)	1
T	(this document)	1
T6	(this document)	1

Table 1: DNS Extension Language Field Types Registry Initial Values

7. References

7.1. References - Normative

[RFC1034] Mockapetris, P., "Domain names - concepts and facilities", STD 13, RFC 1034, DOI 10.17487/RFC1034, November 1987, <http://www.rfc-editor.org/info/rfc1034>.

[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>.

[RFC3548] Josefsson, S., Ed., "The Base16, Base32, and Base64 Data Encodings", RFC 3548, DOI 10.17487/RFC3548, July 2003, <http://www.rfc-editor.org/info/rfc3548>.

[RFC5234] Crocker, D., Ed. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, DOI 10.17487/RFC5234, January 2008, <http://www.rfc-editor.org/info/rfc5234>.

[RFC5646] Phillips, A., Ed. and M. Davis, Ed., "Tags for Identifying Languages", BCP 47, RFC 5646, DOI 10.17487/RFC5646, September 2009, <http://www.rfc-editor.org/info/rfc5646>.

7.2. References - Informative

[RFC2782] Gulbrandsen, A., Vixie, P., and L. Esibov, "A DNS RR for specifying the location of services (DNS SRV)", RFC 2782, DOI 10.17487/RFC2782, February 2000, <http://www.rfc-editor.org/info/rfc2782>.

[RFC4033] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "DNS Security Introduction and Requirements", RFC 4033, DOI 10.17487/RFC4033, March 2005, <http://www.rfc-editor.org/info/rfc4033>.

[RFC4398] Josefsson, S., "Storing Certificates in the Domain Name System (DNS)", RFC 4398, DOI 10.17487/RFC4398, March 2006, <http://www.rfc-editor.org/info/rfc4398>.

Appendix A. Change Log

NOTE TO RFC EDITOR: This section may be removed upon publication of this document as an RFC.

A.1. Changes from -07 to -08

Add counted hex and raw strings and other new types. Added language tags. Added field names.

A.2. Changes from -06 to -07

Add RRTYPE=1 tag in TXT records.

Allow digits and hyphens in qualifier tags, for names like SHA-1.

A.3. Changes from -05 to -06

Fix formatting problems.

Add RRTYPE option "X".

A.4. Changes from -04 to -05

DNS publication in RRYPE.ARPA and RRNAME.ARPA.

A.5. Changes from -03 to -04

More use cases.

Fix up BNF

A.6. Changes from -02 to -03

First stab at BNF

Note $ORIGIN matters

A.7. Changes from -01 to -02

Editorial nits

A.8. Changes from -00 to -01

Switch to multi-line format. Add comments for provisioning.

Authors' Addresses

John Levine
Taughannock Networks
PO Box 727
Trumansburg, NY 14886

Phone: +1 831 480 2300
Email: standards@taugh.com
URI: http://jl.ly

Paul Vixie
950 Charter Street
Redwood City, CA

Email: vixie@fsi.io