Network Working Group
Request for Comments: 2673
Category: Standards Track

M. Crawford
Fermilab

August 1999

Binary Labels in the Domain Name System

Status of this Memo

Copyright Notice

1. Introduction and Terminology

This document defines a "Bit-String Label" which may appear within domain names. This new label type compactly represents a sequence of "One-Bit Labels" and enables resource records to be stored at any bit-boundary in a binary-named section of the domain name tree.

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 [KWORD].

2. Motivation

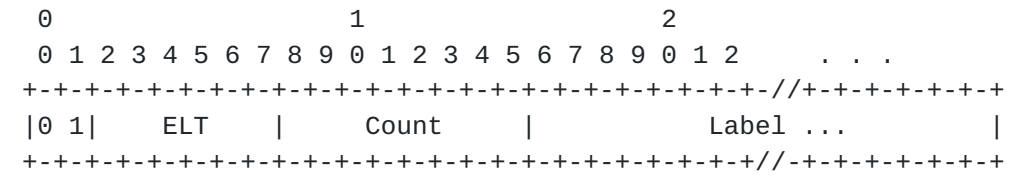
Binary labels are intended to efficiently solve the problem of storing data and delegating authority on arbitrary boundaries when the structure of underlying name space is most naturally represented in binary.

3. Label Format

Up to 256 One-Bit Labels can be grouped into a single Bit-String Label. Within a Bit-String Label the most significant or "highest level" bit appears first. This is unlike the ordering of DNS labels themselves, which has the least significant or "lowest level" label first. Nonetheless, this ordering seems to be the most natural and efficient for representing binary labels.

Among consecutive Bit-String Labels, the bits in the first-appearing label are less significant or "at a lower level" than the bits in subsequent Bit-String Labels, just as ASCII labels are ordered.

3.1. Encoding



(Each tic mark represents one bit.)

ELT 000001 binary, the six-bit extended label type [EDNS0] assigned to the Bit-String Label.

Count The number of significant bits in the Label field. A Count value of zero indicates that 256 bits are significant. (Thus the null label representing the DNS root cannot be represented as a Bit String Label.)

Label The bit string representing a sequence of One-Bit Labels, with the most significant bit first. That is, the One-Bit Label in position 17 in the diagram above represents a subdomain of the domain represented by the One-Bit Label in position 16, and so on.

The Label field is padded on the right with zero to seven pad bits to make the entire field occupy an integral number of octets. These pad bits MUST be zero on transmission and ignored on reception.

A sequence of bits may be split into two or more Bit-String Labels, but the division points have no significance and need not be preserved. An excessively clever server implementation might split Bit-String Labels so as to maximize the effectiveness of message compression [DNSIS]. A simpler server might divide Bit-String Labels at zone boundaries, if any zone boundaries happen to fall between One-Bit Labels.

3.2. Textual Representation

A Bit-String Label is represented in text -- in a zone file, for example -- as a <bit-spec> surrounded by the delimiters "\[" and "]". The <bit-spec> is either a dotted quad or a base indicator and a sequence of digits appropriate to that base, optionally followed by a

slash and a length. The base indicators are "b", "o" and "x", denoting base 2, 8 and 16 respectively. The length counts the significant bits and MUST be between 1 and 32, inclusive, after a dotted quad, or between 1 and 256, inclusive, after one of the other forms. If the length is omitted, the implicit length is 32 for a dotted quad or 1, 3 or 4 times the number of binary, octal or hexadecimal digits supplied, respectively, for the other forms.

In augmented Backus-Naur form [ABNF],

```
  bit-string-label =  "\[" bit-spec "]"

  bit-spec         =  bit-data [ "/" length ]
                   / dotted-quad [ "/" slength ]

  bit-data         =  "x" 1*64HEXDIG
                   / "o" 1*86OCTDIG
                   / "b" 1*256BIT

  dotted-quad      =  decbyte "." decbyte "." decbyte "." decbyte

  decbyte          =  1*3DIGIT

  length           =  NZDIGIT *2DIGIT

  slength          =  NZDIGIT [ DIGIT ]

  OCTDIG           =  %x30-37

  NZDIGIT          =  %x31-39
```

If a <length> is present, the number of digits in the <bit-data> MUST be just sufficient to contain the number of bits specified by the <length>. If there are insignificant bits in a final hexadecimal or octal digit, they MUST be zero. A <dotted-quad> always has all four parts even if the associated <slength> is less than 24, but, like the other forms, insignificant bits MUST be zero.

Each number represented by a <decbyte> must be between 0 and 255, inclusive.

The number represented by <length> must be between 1 and 256 inclusive.

The number represented by <slength> must be between 1 and 32 inclusive.

When the textual form of a Bit-String Label is generated by machine, the length SHOULD be explicit, not implicit.

3.2.1. Examples

The following four textual forms represent the same Bit-String Label.

```
\[b11010000011101]
\[o64072/14]
\[xd074/14]
\[208.116.0.0/14]
```

The following represents two consecutive Bit-String Labels which denote the same relative point in the DNS tree as any of the above single Bit-String Labels.

```
\[b11101].\[o640]
```

3.3. Canonical Representation and Sort Order

Both the wire form and the text form of binary labels have a degree of flexibility in their grouping into multiple consecutive Bit-String Labels. For generating and checking DNS signature records [DNSSEC] binary labels must be in a predictable form. This canonical form is defined as the form which has the fewest possible Bit-String Labels and in which all except possibly the first (least significant) label in any sequence of consecutive Bit-String Labels is of maximum length.

For example, the canonical form of any sequence of up to 256 One-Bit Labels has a single Bit-String Label, and the canonical form of a sequence of 513 to 768 One-Bit Labels has three Bit-String Labels of which the second and third contain 256 label bits.

The canonical sort order of domain names [DNSSEC] is extended to encompass binary labels as follows. Sorting is still label-by-label, from most to least significant, where a label may now be a One-Bit Label or a standard (code 00) label. Any One-Bit Label sorts before any standard label, and a 0 bit sorts before a 1 bit. The absence of a label sorts before any label, as specified in [DNSSEC].

For example, the following domain names are correctly sorted.

```
foo.example
\[b1].foo.example
\[b100].foo.example
\[b101].foo.example
bravo.\[b10].foo.example
alpha.foo.example
```

4. Processing Rules

A One-Bit Label never matches any other kind of label. In particular, the DNS labels represented by the single ASCII characters "0" and "1" do not match One-Bit Labels represented by the bit values 0 and 1.

5. Discussion

A Count of zero in the wire-form represents a 256-bit sequence, not to optimize that particular case, but to make it completely impossible to have a zero-bit label.

6. IANA Considerations

This document defines one Extended Label Type, termed the Bit-String Label, and requests registration of the code point 000001 binary in the space defined by [EDNS0].

7. Security Considerations

All security considerations which apply to traditional ASCII DNS labels apply equally to binary labels. he canonicalization and sorting rules of section 3.3 allow these to be addressed by DNS Security [DNSSEC].

8. References

[ABNF] Crocker, D. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", RFC 2234, November 1997.

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

[DNSSEC] Eastlake, D., 3rd, C. Kaufman, "Domain Name System Security Extensions", RFC 2065, January 1997

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

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

9. Author's Address

Matt Crawford
Fermilab MS 368
PO Box 500
Batavia, IL 60510
USA

Phone: +1 630 840-3461
EMail: crawdad@fnal.gov

10. Full Copyright Statement

Acknowledgement

Funding for the RFC Editor function is currently provided by the Internet Society.