Network Working Group
Request for Comments: 3123
Category: Experimental

P. Koch
Universitaet Bielefeld

June 2001

A DNS RR Type for Lists of Address Prefixes (APL RR)

Status of this Memo

This memo defines an Experimental Protocol for the Internet community. It does not specify an Internet standard of any kind. Discussion and suggestions for improvement are requested. Distribution of this memo is unlimited.

Copyright Notice

Abstract

The Domain Name System (DNS) is primarily used to translate domain names into IPv4 addresses using A RRs (Resource Records). Several approaches exist to describe networks or address ranges. This document specifies a new DNS RR type "APL" for address prefix lists.

1. Conventions used in this document

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 [RFC2119].

Domain names herein are for explanatory purposes only and should not be expected to lead to useful information in real life [RFC2606].

2. Background

The Domain Name System [RFC1034], [RFC1035] provides a mechanism to associate addresses and other Internet infrastructure elements with hierarchically built domain names. Various types of resource records have been defined, especially those for IPv4 and IPv6 [RFC2874] addresses. In [RFC1101] a method is described to publish information about the address space allocated to an organisation. In older BIND versions, a weak form of controlling access to zone data was implemented using TXT RRs describing address ranges.

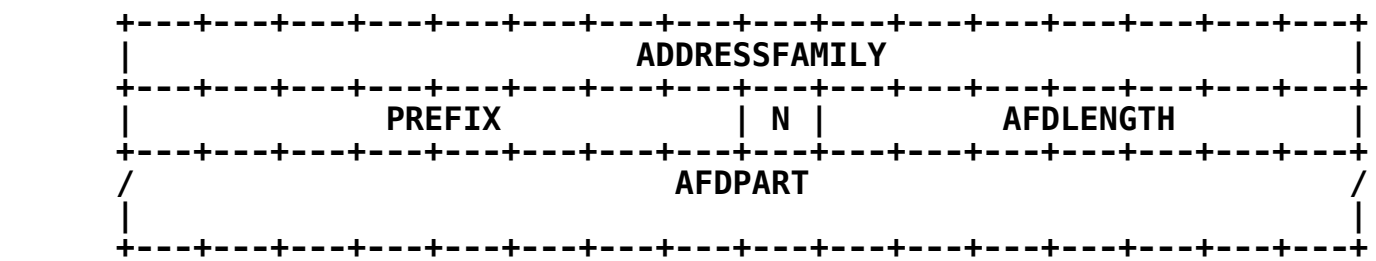
This document specifies a new RR type for address prefix lists.

3. APL RR Type

An APL record has the DNS type of "APL" and a numeric value of 42 [IANA]. The APL RR is defined in the IN class only. APL RRs cause no additional section processing.

4. APL RDATA format

The RDATA section consists of zero or more items (<apitem>) of the form



ADDRESSFAMILY	16 bit unsigned value as assigned by IANA (see IANA Considerations)
PREFIX	8 bit unsigned binary coded prefix length. Upper and lower bounds and interpretation of this value are address family specific.
N	negation flag, indicates the presence of the "!" character in the textual format. It has the value "1" if the "!" was given, "0" else.
AFDLENGTH	length in octets of the following address family dependent part (7 bit unsigned).
AFDPART	address family dependent part. See below.

This document defines the AFDPARTs for address families 1 (IPv4) and 2 (IPv6). Future revisions may deal with additional address families.

4.1. AFDPART for IPv4

The encoding of an IPv4 address (address family 1) follows the encoding specified for the A RR by [RFC1035], section 3.4.1.

PREFIX specifies the number of bits of the IPv4 address starting at the most significant bit. Legal values range from 0 to 32.

Trailing zero octets do not bear any information (e.g., there is no semantic difference between 10.0.0.0/16 and 10/16) in an address prefix, so the shortest possible AFDLENGTH can be used to encode it. However, for DNSSEC [RFC2535] a single wire encoding must be used by

all. Therefore the sender MUST NOT include trailing zero octets in the AFDPART regardless of the value of PREFIX. This includes cases in which AFDLENGTH times 8 results in a value less than PREFIX. The AFDPART is padded with zero bits to match a full octet boundary.

An IPv4 AFDPART has a variable length of 0 to 4 octets.

4.2. AFDPART for IPv6

The 128 bit IPv6 address (address family 2) is encoded in network byte order (high-order byte first).

PREFIX specifies the number of bits of the IPv6 address starting at the most significant bit. Legal values range from 0 to 128.

With the same reasoning as in 4.1 above, the sender MUST NOT include trailing zero octets in the AFDPART regardless of the value of PREFIX. This includes cases in which AFDLENGTH times 8 results in a value less than PREFIX. The AFDPART is padded with zero bits to match a full octet boundary.

An IPv6 AFDPART has a variable length of 0 to 16 octets.

5. Zone File Syntax

The textual representation of an APL RR in a DNS zone file is as follows:

<owner> IN <TTL> APL {[!]afi:address/prefix}*

The data consists of zero or more strings of the address family indicator <afi>, immediately followed by a colon ":", an address, immediately followed by the "/" character, immediately followed by a decimal numeric value for the prefix length. Any such string may be preceded by a "!" character. The strings are separated by whitespace. The <afi> is the decimal numeric value of that particular address family.

5.1. Textual Representation of IPv4 Addresses

An IPv4 address in the <address> part of an <apitem> is in dotted quad notation, just as in an A RR. The <prefix> has values from the interval 0..32 (decimal).

5.2. Textual Representation of IPv6 Addresses

The representation of an IPv6 address in the <address> part of an <apitem> follows [RFC2373], section 2.2. Legal values for <prefix> are from the interval 0..128 (decimal).

6. APL RR usage

An APL RR with empty RDATA is valid and implements an empty list. Multiple occurrences of the same <apitem> in a single APL RR are allowed and MUST NOT be merged by a DNS server or resolver. <apitems> MUST be kept in order and MUST NOT be rearranged or aggregated.

A single APL RR may contain <apitems> belonging to different address families. The maximum number of <apitems> is upper bounded by the available RDATA space.

RRSets consisting of more than one APL RR are legal but the interpretation is left to the particular application.

7. Applicability Statement

The APL RR defines a framework without specifying any particular meaning for the list of prefixes. It is expected that APL RRs will be used in different application scenarios which have to be documented separately. Those scenarios may be distinguished by characteristic prefixes placed in front of the DNS owner name.

An APL application specification MUST include information on

- the characteristic prefix, if any
- how to interpret APL RRSets consisting of more than one RR
- how to interpret an empty APL RR
- which address families are expected to appear in the APL RRs for that application
- how to deal with APL RR list elements which belong to other address families, including those not yet defined
- the exact semantics of list elements negated by the "!" character

Possible applications include the publication of address ranges similar to [RFC1101], description of zones built following [RFC2317] and in-band access control to limit general access or zone transfer (AXFR) availability for zone data held in DNS servers.

The specification of particular application scenarios is out of the scope of this document.

8. Examples

The following examples only illustrate some of the possible usages outlined in the previous section. None of those applications are hereby specified nor is it implied that any particular APL RR based application does exist now or will exist in the future.

```
; RFC 1101-like announcement of address ranges for foo.example
foo.example.              IN APL 1:192.168.32.0/21 !1:192.168.38.0/28

; CIDR blocks covered by classless delegation
42.168.192.IN-ADDR.ARPA. IN APL ( 1:192.168.42.0/26 1:192.168.42.64/26
                                  1:192.168.42.128/25 )

; Zone transfer restriction
_axfr.sbo.example.        IN APL 1:127.0.0.1/32 1:172.16.64.0/22

; List of address ranges for multicast
multicast.example.        IN APL 1:224.0.0.0/4  2:FF00:0:0:0:0:0:0:0/8
```

Note that since trailing zeroes are ignored in the first APL RR the AFDLENGTH of both <apitems> is three.

9. Security Considerations

Any information obtained from the DNS should be regarded as unsafe unless techniques specified in [RFC2535] or [RFC2845] were used. The definition of a new RR type does not introduce security problems into the DNS, but usage of information made available by APL RRs may compromise security. This includes disclosure of network topology information and in particular the use of APL RRs to construct access control lists.

10. IANA Considerations

This section is to be interpreted as following [RFC2434].

This document does not define any new namespaces. It uses the 16 bit identifiers for address families maintained by IANA in http://www.iana.org/numbers.html.

The IANA assigned numeric RR type value 42 for APL [IANA].

11. Acknowledgements

The author would like to thank Mark Andrews, Olafur Gudmundsson, Ed Lewis, Thomas Narten, Erik Nordmark, and Paul Vixie for their review and constructive comments.

12. References

[RFC1034] Mockapetris, P., "Domain Names - Concepts and Facilities", STD 13, RFC 1034, November 1987.

[RFC1035] Mockapetris, P., "Domain Names - Implementation and Specification", STD 13, RFC 1035, November 1987.

[RFC1101] Mockapetris, P., "DNS Encoding of Network Names and Other Types", RFC 1101, April 1989.

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

[RFC2181] Elz, R. and R. Bush, "Clarifications to the DNS Specification", RFC 2181, July 1997.

[RFC2317] Eidnes, H., de Groot, G. and P. Vixie, "Classless IN-ADDR.ARPA delegation", BCP 20, RFC 2317, March 1998.

[RFC2373] Hinden, R. and S. Deering, "IP Version 6 Addressing Architecture", RFC 2373, July 1998.

[RFC2434] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 2434, October 1998.

[RFC2535] Eastlake, D., "Domain Name System Security Extensions", RFC 2535, March 1999.

[RFC2606] Eastlake, D. and A. Panitz, "Reserved Top Level DNS Names", BCP 32, RFC 2606, June 1999.

[RFC2845] Vixie, P., Gudmundsson, O., Eastlake, D. and B. Wellington, "Secret Key Transaction Authentication for DNS (TSIG)", RFC 2845, May 2000.

[RFC2874] Crawford, M. and C. Huitema, "DNS Extensions to Support IPv6 Address Aggregation and Renumbering", RFC 2874, July 2000.

[IANA] http://www.iana.org/assignments/dns-parameters

13. Author's Address

Peter Koch
Universitaet Bielefeld
Technische Fakultaet
D-33594 Bielefeld
Germany

Phone: +49 521 106 2902
EMail: pk@TechFak.Uni-Bielefeld.DE

14. Full Copyright Statement

Acknowledgement

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