

Network Working Group
Internet-Draft
Updates: 3404, 3959
(if approved)
Intended status: Standards Track
Expires: April 14, 2011

P. Faltstrom
Cisco
O. Kolkman
NLNet
October 11, 2010

The Uniform Resource Identifier (URI) DNS Resource Record
draft-faltstrom-uri-06.txt

Abstract

This document defines a new DNS resource record, called the Uniform Resource Identifier (URI) RR, for publishing mappings from hostnames to URIs.

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 April 14, 2011.

Copyright Notice

Copyright (c) 2010 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	3
2. Applicability Statement	3
3. DNS considerations	4
4. The format of the URI RR	4
4.1. Ownername, class and type	4
4.2. Priority	5
4.3. Weight	5
4.4. Target	5
4.5. URI RDATA Wire Format	6
5. Definition of the flag 'D' for NAPTR records	6
6. Examples	7
6.1. Homepage at one domain, but two domains in use	7
7. Relation to S-NAPTR	7
8. Relation to U-NAPTR	7
9. Relation to SRV	8
10. IANA Considerations	8
10.1. Registration of the URI Resource Record Type	8
10.2. Registration of services	8
11. Security Considerations	8
12. Acknowledgements	9
Appendix A. RRTYPE Allocation Request	9
13. References	12
13.1. Normative References	12
13.2. Non-normative references	12
Authors' Addresses	13

1. Introduction

This document explains the use of the Domain Name System (DNS) for the storage of URIs, and how to resolve hostnames to such URIs that can be used by various applications. For resolution the application need to know both the hostname and the protocol that the URI is to be used for. The protocol is registered by IANA.

Currently, looking up URIs given a hostname uses the DDDS [RFC3401] application framework with the DNS as a database as specified in RFC 3404 [RFC3404]. This has a number of implications such as the inability to select what NAPTR records that match the query are interesting. The RRSset returned will always consist of all URIs "connected" with the domain in question.

The URI resource record specified in this document enables the querying party to select which ones of the NAPTR records one is interested in. This because data in the service field of the NAPTR record is included in the owner part of the URI resource record type.

Querying for URI resource records is not replacing querying for NAPTR resource records (or use of S-NAPTR [RFC3958]). Instead, the URI resource record type provides a complementary mechanism to use when one already knows what service field is interesting. With it, one can directly query for the specific subset of the otherwise possibly large RRSset given back when querying for NAPTR resource records.

This document updates RFC 3958 and RFC 3404 by adding the flag "D" to the list of defined terminal flags in section 2.2.3 of RFC 3958 and 4.3 of RFC 3404.

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 BCP 14, RFC 2119 [RFC2119].

2. Applicability Statement

In general, it is expected that URI records will be used by clients for applications where the relevant protocol to be used is known, but, for example, an extra abstraction is needed in order to separate a domain name from a point of service (as addressed by the URI). One example of such a situation is when an organisation has many domain names, but only one official web page.

Applications MUST know the specific service fields to prepend the hostname with. Using repetitive queries for URI records MUST NOT be

a replacement for querying for NAPTR records according to the NAPTR (DDDS) or S-NAPTR algorithms. NAPTR records serve the purpose to discover the various services and URIs for looking up access points for a given service. Those are two very different kinds of needs.

3. DNS considerations

Using prefix labels, such as underscored service tags, prevents the use of wildcards, as constructs as `_s2._s1.*.example.net.` are not possible in the DNS, see RFC 4592 [RFC4592]. Besides, underscored service tags used for the URI RR (based on the NAPTR service descriptions) may have slightly different semantics than service tags used for underscored prefix labels that are used in combination with other (yet unspecified) RR types. This may cause subtle management problems when delegation structure that has developed within the context of URI RRs is also to be used for other RR types. Since the service labels might be overloaded, applications should carefully check that the application level protocol is indeed the protocol they expect.

Subtle management issues may also arise when the delegations from service to sub service label involves several parties and different stake holders.

4. The format of the URI RR

This is the presentation format of the URI RR:

```
Ownername TTL Class URI Priority Weight Target
```

The URI RR does not cause any kind of Additional Section processing.

4.1. Ownername, class and type

The URI ownername is subject to special conventions.

Just like the SRV RR [RFC2782] the URI RR has service information encoded in its ownername. In order to encode the service for a specific owner name one uses service parameters. Valid service parameters used are those used for SRV resource records, or registered by IANA for Enumservice Registrations. The Enumservice Registration parameters are reversed (subtype(s) before type), prepended with an underscore (`_`) and prepended to the owner name in separate labels. The underscore is prepended to the service

parameters to avoid collisions with DNS labels that occur in nature, and the order is reversed to make it possible to do delegations, if needed, to different zones (and therefore providers of DNS).

It should be noted that the usage of a prefix must be described in detail in for example the Enumservice Registration documentation, or in a specific document that clarifies potential overload of parameters in the same URI. Specifically, registered URI schemes are not automatically acceptable as a service. With the HTTP scheme, one can for example have multiple methods (GET, PUT, etc), and this with the same URI.

For example, suppose we are looking for the URI for a service with Service Parameter "A:B:C" for host example.com.. Then we would query for (QNAME,QTYPE)=("_C._B._A.example.com","URI")

The type number for the URI record is TBD1 (to be assigned by IANA).

The URI resource record is class independent.

The URI RR has no special TTL requirements.

4.2. Priority

The priority of the target URI in this RR. Its range is 0-65535. A client MUST attempt to contact the URI with the lowest-numbered priority it can reach; URIs with the same priority SHOULD be tried in the order defined by the weight field.

4.3. Weight

A server selection mechanism. The weight field specifies a relative weight for entries with the same priority. Larger weights SHOULD be given a proportionately higher probability of being selected. The range of this number is 0-65535.

4.4. Target

The URI of the target, enclosed in double-quote characters (''). Resolution of the URI is according to the definitions for the Scheme of the URI.

The URI is encoded as one or more <character-string> RFC1035 section 3.3 [RFC1035].

4.5. URI RDATA Wire Format

The RDATA for a URI RR consists of a 2 octet Priority field, a two octet Weight field, and a variable length target field.

Priority and Weight are unsigned integers in network byte order.

The Target field contains the URI (without the enclosing double-quote characters used in the presentation format), encoded as a sequence of one or more <character-string> (as specified in section 3.3 of RFC 1035 [RFC1035]), where all but the last <character-string> are filled up to the maximum length of 255 octets.

The Target field can also contain an IRI, but with the additional requirements that it is in UTF-8 [RFC3629] and possible to convert to a URI according to section 3.1 of RFC 3987 [RFC3987] and back again to an IRI according to section 3.2. Other character sets than UTF-8 are not allowed. The domain name part of the IRI can be either an U-LABEL or A-LABEL as defined in RFC 5890 [RFC5890].

```

          1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 3 3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-----+-----+-----+-----+-----+-----+-----+-----+
|           Priority           |           Weight           |
+-----+-----+-----+-----+-----+-----+-----+-----+
/                               /
/                               /
/                               /
+-----+-----+-----+-----+-----+-----+-----+-----+

```

5. Definition of the flag 'D' for NAPTR records

This document specifies the flag "D" for use as a flag in NAPTR records. The flag indicate a terminal NAPTR record because it denotes the end of the DDDS/NAPTR processing rules. In the case of a "D" flag, the Replacement field in the NAPTR record, prepended with the service flags, is used as the Owner of a DNS query for URI records, and normal URI processing as defined in this document is applied.

The replacement field MUST NOT include any of the service parameters. Those are to be prepended (together with underscore) as described in other places in this document.

The Regexp field in the NAPTR record MUST be empty when the 'D' flag

is in use.

6. Examples

6.1. Homepage at one domain, but two domains in use

An organisation has the domain names example.com and example.net, but the official URI `http://www.example.com/`. Given the service type "web" and subtype "http" (from the IANA registry), the following URI Resource Records could be made available in the respective zones (example.com and example.net):

```
$ORIGIN example.com.  
_http._web      IN URI 10 1 "http://www.example.com/"  
  
$ORIGIN example.net.  
_http._web      IN URI 10 1 "http://www.example.com/"
```

7. Relation to S-NAPTR

The URI resource record type is not a replacement for the S-NAPTR. It is instead an extension and the second step of the S-NAPTR resolution can resolve a URI resource record instead of using SRV records and yet another algorithm for how to use SRV records for the specific protocol.

```
$ORIGIN example.com.  
;;      order pref flags  
  IN NAPTR 100    10    "s"      "EM:ProtA"          ( ; service  
                                     " "                ; regexp  
                                     _ProtA._tcp.example.com. ; replacement  
_ProtA._tcp IN URI "schemeA:service.example.com/example"
```

8. Relation to U-NAPTR

The URI Resource Record Type, together with S-NAPTR, can be viewed as a replacement for the U-NAPTR. The URI Resource Record Type is though only interesting when one know a base domain name, a protocol and service so that one can compose the record to look up. NAPTR records of any kind are used to look up what services exists for a certain domain, which is one step before the URI resource record is

used.

9. Relation to SRV

The URI Resource Record Type can be viewed as a replacement for the SRV record. This because it like the SRV record can only be looked up if one know the base domain, the protocol and the service. It has a similar functionality, but instead of returning a hostname and port number, the URI record return a full URI. As such, it can be viewed as a more powerful resource record than SRV.

10. IANA Considerations

10.1. Registration of the URI Resource Record Type

IANA has assigned Resource Record Type TBD1 for the URI Resource Record Type and added the line depicted below to the registry named Resource Record (RR) TYPEs and QTYPEs as defined in BCP 42 RFC 5395 [RFC5395] and located at <http://www.iana.org/assignments/dns-parameters>.

TYPE	Value and meaning	Reference
-----	-----	-----
URI	TBD1 a URI for a service (per the owner name)	[RFCXXXX]

10.2. Registration of services

No new registry is needed for the registration of services as the Enumservice Registrations registry is used also for the URI resource record type.

11. Security Considerations

The authors do not believe this resource record cause any new security problems. Deployment must though be done in a proper way as misconfiguration of this resource record might make it impossible to reach the service that was originally intended to be accessed.

Using the URI resource record together with security mechanisms that relies on verification of authentication of hostnames, like TLS, makes it important to choose the correct domain name when doing the comparison.

The basic mechanism works as follows:

1. Announce the fact example.com is hosted at example.org (with some URL) in DNS
2. Secure the URI resource record with DNSSEC.
3. Verify the TLS (for example) certificate for the connection to example.org matches, i.e. use the hostname in the URI and not the hostname used originally when looking up the URI resource record.
4. If needed, do application layer authentication etc over the then encrypted connection.

What also can happen is that the URI in the resource record type has errors in it. Applications using the URI resource record type for resolution should behave similarly as if the user typed (or copy and pasted) the URI. At least it must be clear to the user that the error is not due to any error from his side.

One SHOULD not include userinfo (see User Information, Section 3.2.1, in RFC 3986 [RFC3986]) in a URI that is used in a URI resource record as DNS data must be viewed as publicly available information.

12. Acknowledgements

Ideas on how to split the two different kind of queries "What services exists for this domain name" and "What is the URI for this service" came from Scott Bradner and Lawrence Conroy. Other people that have contributed to this document include Richard Barnes, Leslie Daigle, Olafur Gudmundsson, Ted Hardie, Peter Koch and Penn Pfautz.

Appendix A. RRTYPE Allocation Request

A. Submission Date:

May 23, 2009

B. Submission Type:

[X] New RRTYPE
[] Modification to existing RRTYPE

C. Contact Information for submitter:

Name: Patrik Faltstrom
Email Address: paf@cisco.com
International telephone number: +46-8-6859131

Other contact handles:

(Note: This information will be publicly posted.)

D. Motivation for the new RRTYPE application?

There is no easy way to get from a domain name to a URI (or IRI). Some mechanisms exist via use of the NAPTR [RFC3403] resource record. That implies quite complicated rules that are simplified via the S-NAPTR [RFC3958] specification. But, the ability to directly look up a URI still exists. This specification uses a prefix based naming mechanism originated in the definition of the SRV [RFC2782] resource record, and the RDATA is a URI, encoded as one text field.

See also above (Section 1).

E. Description of the proposed RR type.

The format of the URI resource record is as follows:

Ownername TTL Class URI Priority Weight Target

The URI RR has service information encoded in its ownername. In order to encode the service for a specific owner name one uses service parameters. Valid service parameters used are either Enumservice Registrations registered by IANA, or prefixes used for the SRV resource record.

The wire format of the RDATA is as follows:

```

          1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 3 3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|           Priority           |           Weight           |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
/                                                                    /
/                               Target                               /
/                                                                    /
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+

```

- F. What existing RRTYPE or RRTYPEs come closest to filling that need and why are they unsatisfactory?

The RRTYPE that come closest is the NAPTR resource record. It is for example used in the DDDS and S-NAPTR algorithms. The main problem with the NAPTR is that selection of what record (or records) one is interested in is based on data stored in the RDATA portion of the NAPTR resource record. This, as explained in RFC 5507 [RFC5507], is not optimal for DNS lookups. Further, most applications using NAPTR resource records uses regular expression based rewrite rules for creation of the URI, and that has shown be complicated to implement.

The second closest RRTYPE is the SRV record that given a prefixed based naming just like is suggested for the URI resource record, one get back a port number and domain name. This can also be used for creation of a URI, but, only URIs without path components.

- G. What mnemonic is requested for the new RRTYPE (optional)?

URI

- H. Does the requested RRTYPE make use of any existing IANA Registry or require the creation of a new IANA sub-registry in DNS Parameters?

Yes, partially.

One of the mechanisms to select a service is to use the Enumservice Registry managed by IANA. Another is to use services and protocols used for SRV records.

- I. Does the proposal require/expect any changes in DNS servers/resolvers that prevent the new type from being processed as an unknown RRTYPE (see [RFC3597])?

No

- J. Comments:

None

13. References

13.1. Normative References

- [E164] ITU-T, "The International Public Telecommunication Number Plan", Recommendation E.164, May 1997.
- [RFC1035] Mockapetris, P., "Domain names - implementation and specification", STD 13, RFC 1035, November 1987.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC3629] Yergeau, F., "UTF-8, a transformation format of ISO 10646", STD 63, RFC 3629, November 2003.
- [RFC3958] Daigle, L. and A. Newton, "Domain-Based Application Service Location Using SRV RRs and the Dynamic Delegation Discovery Service (DDDS)", RFC 3958, January 2005.
- [RFC3987] Duerst, M. and M. Suignard, "Internationalized Resource Identifiers (IRIs)", RFC 3987, January 2005.
- [RFC5395] Eastlake, D., "Domain Name System (DNS) IANA Considerations", BCP 42, RFC 5395, November 2008.
- [RFC5890] Klensin, J., "Internationalized Domain Names for Applications (IDNA): Definitions and Document Framework", RFC 5890, August 2010.

13.2. Non-normative references

- [RFC2782] Gulbrandsen, A., Vixie, P., and L. Esibov, "A DNS RR for specifying the location of services (DNS SRV)", RFC 2782, February 2000.
- [RFC3401] Mealling, M., "Dynamic Delegation Discovery System (DDDS) Part One: The Comprehensive DDDS", RFC 3401, October 2002.
- [RFC3403] Mealling, M., "Dynamic Delegation Discovery System (DDDS) Part Three: The Domain Name System (DNS) Database", RFC 3403, October 2002.
- [RFC3404] Mealling, M., "Dynamic Delegation Discovery System (DDDS) Part Four: The Uniform Resource Identifiers (URI)", RFC 3404, October 2002.
- [RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, January 2005.

- [RFC4592] Lewis, E., "The Role of Wildcards in the Domain Name System", RFC 4592, July 2006.
- [RFC4848] Daigle, L., "Domain-Based Application Service Location Using URIs and the Dynamic Delegation Discovery Service (DDDS)", RFC 4848, April 2007.
- [RFC5507] IAB, Faltstrom, P., Austein, R., and P. Koch, "Design Choices When Expanding the DNS", RFC 5507, April 2009.

Authors' Addresses

Patrik Faltstrom
Cisco Systems

Email: paf@cisco.com

Olaf Kolkman
NLnet Labs

Email: olaf@NLnetLabs.nl

DNS Extensions Working Group
Internet-Draft
Updates: 2536, 2539, 3110, 4034,
4398, 5155, 5702, 5933
(if approved)
Intended status: Standards Track
Expires: February 12, 2011

S. Rose
NIST
August 11, 2010

Applicability Statement: DNS Security (DNSSEC) DNSKEY Algorithm IANA
Registry
draft-ietf-dnsext-dnssec-registry-fixes-06

Abstract

The DNS Security Extensions (DNSSEC) requires the use of cryptographic algorithm suites for generating digital signatures over DNS data. There is currently an IANA registry for these algorithms that is incomplete in that it lacks the implementation status of each algorithm. This document provides an applicability statement on algorithm status for DNSSEC implementations. This document replaces that registry table with a new IANA registry table for Domain Name System Security (DNSSEC) Algorithm Numbers which lists each algorithm's status based on the current reference. If that status is not defined in the original specification, this document assigns a status.

Status of This Memo

This Internet-Draft is submitted to IETF in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

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."

The list of current Internet-Drafts can be accessed at
<http://www.ietf.org/ietf/lid-abstracts.txt>.

The list of Internet-Draft Shadow Directories can be accessed at
<http://www.ietf.org/shadow.html>.

This Internet-Draft will expire on February 12, 2011.

Copyright Notice

Copyright (c) 2010 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the BSD License.

Table of Contents

1. Introduction	3
1.1. Requirements Language	3
2. The DNS Security Algorithm Number Subregistry	3
2.1. Individual Changes	3
2.2. Domain Name System (DNS) Security Algorithm Number Registry Table	5
2.3. Specifying New Algorithms and Updating Status of Existing Entries	6
3. IANA Considerations	6
4. Security Considerations	6
5. References	6
5.1. Normative References	6
5.2. Informative References	8

1. Introduction

The Domain Name System (DNS) Security Extensions (DNSSEC) [RFC4033], [RFC4034], and [RFC4035] uses digital signatures over DNS data to provide source authentication and integrity protection. DNSSEC uses an IANA registry to allocate codes for digital signature algorithms (consisting of a cryptographic algorithm and one-way hash function).

The original list of algorithm status is found in [RFC4034]. Other DNSSEC documents have added new algorithms or changed the status of algorithms in the registry. However, currently implementors must read through all the documents in order to discover the current status of each algorithm in the registry.

This document replaces the current IANA registry for Domain Name System Security (DNSSEC) Algorithm Numbers with a newly defined registry table. This new table (Section 2.2 below) contains a column that will list the current status of each digital signature algorithm in the registry at the time of writing and assigns status for some algorithms used with DNSSEC that did not have an identified status in their specification. This document updates the following: [RFC2536], [RFC2539], [RFC3110], [RFC4034], [RFC4398], [RFC5155], [RFC5702], and [RFC5933].

1.1. Requirements Language

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].

2. The DNS Security Algorithm Number Subregistry

The DNS Security Algorithm Number subregistry (part of the Domain Name System (DNS) Security Number registry) will be replaced with the table below. This table contains a column that contains the current implementation requirements of the given algorithm.

There are additional differences to entries that are described in sub-section 2.1. The overall new registry table is in sub-section 2.2. The values for the status were obtained from [RFC4034] with updates for algorithms specified after the original DNSSEC specification. If no status was listed in the original specification, this document assigns one.

2.1. Individual Changes

This document changes three entries in the Domain Name System Security (DNSSEC) Algorithm Registry. They are:

The description for assignment number 4 is changed to "Reserved until 2020".

The description for assignment number 9 is changed to "Reserved until 2020".

The description for assignment number 11 is changed to "Reserved until 2020".

Registry entries 13-251 remains Unassigned.

The status of RSASHA1-NSEC3-SHA1 and DSA-NSEC3-SHA1 are set to RECOMMENDED and OPTIONAL respectively. The difference is due to the fact that RSA/SHA-1 is REQUIRED and DSA/SHA-1 is only OPTIONAL. The status of RSA/SHA-256 and RSA/SHA-512 are set to RECOMMENDED as it is believed that these algorithms will replace older algorithms (e.g. RSA/SHA-1) that have a perceived weakness in their hash algorithm (SHA-1).

2.2. Domain Name System (DNS) Security Algorithm Number Registry Table

The Domain Name System (DNS) Security Algorithm Number registry is hereby specified as follows:

Number	Description	Mnemonic	Zone Sign	Transaction Sign	Status	Reference
0	Reserved					[RFC4398]
1	RSA/MD5	RSAMD5	N	Y	MUST NOT IMPLEMENT	[RFC4034], [RFC3110] (this memo)
2	Diffie-Hellman	DH	N	Y		[RFC2539] (this memo)
3	DSA/SHA-1	DSASHA1	Y	Y		[RFC2536], [RFC4034], FIPS 186-3, FIPS 180-3 (this memo) (this memo)
4	Reserved until 2020	ECC				
5	RSA/SHA-1	RSASHA1	Y	Y	REQUIRED	[RFC4034] (this memo)
6	DSA-NSEC3-SHA1	DSA-NSEC3 -SHA1	Y	Y		[RFC5155] (this memo)
7	RSASHA1-NSEC3 -SHA1	RSASHA1- NSEC3- SHA1	Y	Y	RECOMMENDED	[RFC5155] (this memo)
8	RSA/SHA-256	RSASHA256	Y	*	RECOMMENDED	[RFC5702] (this memo) (this memo)
9	Reserved until 2020					
10	RSA/SHA-512	RSASHA512	Y	*	RECOMMENDED	[RFC5702] (this memo) (this memo)
11	Reserved until 2020					
12	GOST R 34.10-2001	GOST-ECC	Y	*		[RFC5933] (this memo)
13-251	Unassigned					
252	Reserved for Indirect keys	INDIRECT	N	N		[RFC4034] (this memo)
253	private algorithm	PRIVATE	Y	Y		[RFC4034] (this memo)
254	private algorithm OID	PRIVATEOID	Y	Y		[RFC4034] (this memo)
255	Reserved					

2.3. Specifying New Algorithms and Updating Status of Existing Entries

[I-D.ietf-dnsexp-dnssec-alg-allocation] establishes a parallel procedure for obtaining an algorithm number for new algorithms other than a standards track document. Algorithms entered into the registry using that procedure do not have a listed status. Specifications that follow this path do not need to obsolete or update this document.

Adding a newly specified algorithm to the registry with a status SHALL entail obsoleting this document and replacing the registry table (with the new algorithm entry). Altering the status column value of any existing algorithm in the registry SHALL entail obsoleting this document and replacing the registry table.

This document cannot be updated, only made obsolete and replaced by a successor document.

3. IANA Considerations

This document replaces the Domain Name System (DNS) Security Algorithm Numbers registry. The new registry table is in Section 2.2.

The original Domain Name System (DNS) Security Algorithm Number registry is available at <http://www.iana.org/assignments/dns-sec-alg-numbers/dns-sec-alg-numbers.xhtml>.

4. Security Considerations

This document replaces the Domain Name System (DNS) Security Algorithm Numbers registry. It is not meant to be a discussion on algorithm superiority. No new security considerations are raised in this document.

5. References

5.1. Normative References

- | | |
|---|---|
| [I-D.ietf-dnsexp-dnssec-alg-allocation] | Hoffman, P., "Cryptographic Algorithm Identifier Allocation for DNSSEC", draft-ietf-dnsexp-dnssec-alg-allocation-03 (work in progress), March 2010. |
| [RFC2119] | Bradner, S., "Key words for use in RFCs to Indicate |

- Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC2536] Eastlake, D., "DSA KEYS and SIGs in the Domain Name System (DNS)", RFC 2536, March 1999.
- [RFC2539] Eastlake, D., "Storage of Diffie-Hellman Keys in the Domain Name System (DNS)", RFC 2539, March 1999.
- [RFC3110] Eastlake, D., "RSA/SHA-1 SIGs and RSA KEYS in the Domain Name System (DNS)", RFC 3110, May 2001.
- [RFC4033] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "DNS Security Introduction and Requirements", RFC 4033, March 2005.
- [RFC4034] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "Resource Records for the DNS Security Extensions", RFC 4034, March 2005.
- [RFC4035] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "Protocol Modifications for the DNS Security Extensions", RFC 4035, March 2005.
- [RFC4398] Josefsson, S., "Storing Certificates in the Domain Name System (DNS)", RFC 4398, March 2006.
- [RFC5155] Laurie, B., Sisson, G., Arends, R., and D. Blacka, "DNS Security (DNSSEC) Hashed Authenticated Denial

of Existence", RFC 5155,
March 2008.

[RFC5702]

Jansen, J., "Use of SHA-2
Algorithms with RSA in
DNSKEY and RRSIG Resource
Records for DNSSEC",
RFC 5702, October 2009.

[RFC5933]

Dolmatov, V., Chuprina, A.,
and I. Ustinov, "Use of GOST
Signature Algorithms in
DNSKEY and RRSIG Resource
Records for DNSSEC",
RFC 5933, July 2010.

5.2. Informative References

[FIPS.180-3.2008]

National Institute of
Standards and Technology,
"Secure Hash Standard",
FIPS PUB 180-3,
October 2008, <[http://
csrc.nist.gov/publications/
fips/fips180-3/
fips180-3.pdf](http://csrc.nist.gov/publications/fips/fips180-3/fips180-3.pdf)>.

[FIPS.186-3.2009]

National Institute of
Standards and Technology,
"Digital Signature
Standard", FIPS PUB 186-3,
June 2009, <[http://
csrc.nist.gov/publications/
fips/fips186-3/
fips_186-3.pdf](http://csrc.nist.gov/publications/fips/fips186-3/fips_186-3.pdf)>.

Author's Address

Scott Rose
NIST
100 Bureau Dr.
Gaithersburg, MD 20899
USA

Phone: +1-301-975-8439
EMail: scottr.nist@gmail.com

DNS Extensions Working Group
Internet-Draft
Obsoletes: 2672 (if approved)
Updates: 3363,4294
(if approved)
Intended status: Standards Track
Expires: April 17, 2011

S. Rose
NIST
W. Wijngaards
NLnet Labs
October 14, 2010

Update to DNAME Redirection in the DNS
draft-ietf-dnsext-rfc2672bis-dname-20

Abstract

The DNAME record provides redirection for a sub-tree of the domain name tree in the DNS system. That is, all names that end with a particular suffix are redirected to another part of the DNS. This is a revision of the original specification in RFC 2672, also aligning RFC 3363 and RFC 4294 with this revision.

Requirements Language

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].

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 April 17, 2011.

Copyright Notice

Copyright (c) 2010 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

This document may contain material from IETF Documents or IETF Contributions published or made publicly available before November 10, 2008. The person(s) controlling the copyright in some of this material may not have granted the IETF Trust the right to allow modifications of such material outside the IETF Standards Process. Without obtaining an adequate license from the person(s) controlling the copyright in such materials, this document may not be modified outside the IETF Standards Process, and derivative works of it may not be created outside the IETF Standards Process, except to format it for publication as an RFC or to translate it into languages other than English.

Table of Contents

1. Introduction	4
2. The DNAME Resource Record	4
2.1. Format	4
2.2. The DNAME Substitution	5
2.3. DNAME Owner Name Matching the QNAME	7
2.4. Names Next to and Below a DNAME Record	7
2.5. Compression of the DNAME record.	7
3. Processing	8
3.1. CNAME synthesis	8
3.2. Server algorithm	9
3.3. Wildcards	11
3.4. Acceptance and Intermediate Storage	11
4. DNAME Discussions in Other Documents	11
5. Other Issues with DNAME	13
5.1. Canonical hostnames cannot be below DNAME owners	13
5.2. Dynamic Update and DNAME	13
5.3. DNSSEC and DNAME	13
5.3.1. Signed DNAME, Unsigned Synthesized CNAME	13
5.3.2. DNAME Bit in NSEC Type Map	14
5.3.3. DNAME Chains as Strong as the Weakest Link	14
5.3.4. Validators Must Understand DNAME	14
5.3.4.1. DNAME in Bitmap Causes Invalid Name Error	14
5.3.4.2. Valid Name Error Response Involving DNAME in Bitmap	15
5.3.4.3. Response With Synthesized CNAME	15
6. IANA Considerations	15
7. Security Considerations	15
8. Acknowledgments	16
9. References	16
9.1. Normative References	16
9.2. Informative References	17

1. Introduction

DNAME is a DNS Resource Record type originally defined in RFC 2672 [RFC2672]. DNAME provides redirection from a part of the DNS name tree to another part of the DNS name tree.

The DNAME RR and the CNAME RR [RFC1034] cause a lookup to (potentially) return data corresponding to a domain name different from the queried domain name. The difference between the two resource records is that the CNAME RR directs the lookup of data at its owner to another single name, a DNAME RR directs lookups for data at descendants of its owner's name to corresponding names under a different (single) node of the tree.

Take for example, looking through a zone (see RFC 1034 [RFC1034], section 4.3.2, step 3) for the domain name "foo.example.com" and a DNAME resource record is found at "example.com" indicating that all queries under "example.com" be directed to "example.net". The lookup process will return to step 1 with the new query name of "foo.example.net". Had the query name been "www.foo.example.com" the new query name would be "www.foo.example.net".

This document is a revision of the original specification of DNAME in RFC 2672 [RFC2672]. DNAME was conceived to help with the problem of maintaining address-to-name mappings in a context of network renumbering. With a careful set-up, a renumbering event in the network causes no change to the authoritative server that has the address-to-name mappings. Examples in practice are classless reverse address space delegations.

Another usage of DNAME lies in aliasing of name spaces. For example, a zone administrator may want sub-trees of the DNS to contain the same information. Examples include punycode alternates for domain spaces.

This revision to DNAME does not change the wire format or the handling of DNAME Resource Records. Discussion is added on problems that may be encountered when using DNAME.

2. The DNAME Resource Record

2.1. Format

The DNAME RR has mnemonic DNAME and type code 39 (decimal). It is not class-sensitive.

Its RDATA is comprised of a single field, <target>, which contains a fully qualified domain name that must be sent in uncompressed form [RFC1035], [RFC3597]. The <target> field MUST be present. The presentation format of <target> is that of a domain name [RFC1035].

<owner> <ttl> <class> DNAME <target>

The effect of the DNAME RR is the substitution of the record's <target> for its owner name, as a suffix of a domain name. This substitution is to be applied for all names below the owner name of the DNAME RR. This substitution has to be applied for every DNAME RR found in the resolution process, which allows fairly lengthy valid chains of DNAME RRs.

Details of the substitution process, methods to avoid conflicting resource records, and rules for specific corner cases are given in the following subsections.

2.2. The DNAME Substitution

When following RFC 1034 [RFC1034], section 4.3.2's algorithm's third step, "start matching down, label by label, in the zone" and a node is found to own a DNAME resource record a DNAME substitution occurs. The name being sought may be the original query name or a name that is the result of a CNAME resource record being followed or a previously encountered DNAME. As in the case when finding a CNAME resource record or NS resource record set, the processing of a DNAME will happen prior to finding the desired domain name.

A DNAME substitution is performed by replacing the suffix labels of the name being sought matching the owner name of the DNAME resource record with the string of labels in the RDATA field. The matching labels end with the root label in all cases. Only whole labels are replaced. See the table of examples for common cases and corner cases.

In the table below, the QNAME refers to the query name. The owner is the DNAME owner domain name, and the target refers to the target of the DNAME record. The result is the resulting name after performing the DNAME substitution on the query name. "no match" means that the query did not match the DNAME and thus no substitution is performed and a possible error message is returned (if no other result is possible). Thus every line contains one example substitution. In the examples below, 'cyc' and 'shortloop' contain loops.

QNAME	owner	DNAME	target	result
-----	-----	-----	-----	-----
com.	example.com.	example.net.	<no match>	
example.com.	example.com.	example.net.	[0]	
a.example.com.	example.com.	example.net.	a.example.net.	
a.b.example.com.	example.com.	example.net.	a.b.example.net.	
ab.example.com.	b.example.com.	example.net.	<no match>	
foo.example.com.	example.com.	example.net.	foo.example.net.	
a.x.example.com.	x.example.com.	example.net.	a.example.net.	
a.example.com.	example.com.	y.example.net.	a.y.example.net.	
cyc.example.com.	example.com.	example.com.	cyc.example.com.	
cyc.example.com.	example.com.	c.example.com.	cyc.c.example.com.	
shortloop.x.x.	x.	.	shortloop.x.	
shortloop.x.	x.	.	shortloop.	

[0] The result depends on the QTYPE. If the QTYPE = DNAME, then the result is "example.com." else "<no match>"

Table 1. DNAME Substitution Examples.

It is possible for DNAMEs to form loops, just as CNAMEs can form loops. DNAMEs and CNAMEs can chain together to form loops. A single corner case DNAME can form a loop. Resolvers and servers should be cautious in devoting resources to a query, but be aware that fairly long chains of DNAMEs may be valid. Zone content administrators should take care to insure that there are no loops that could occur when using DNAME or DNAME/CNAME redirection.

The domain name can get too long during substitution. For example, suppose the target name of the DNAME RR is 250 octets in length (multiple labels), if an incoming QNAME that has a first label over 5 octets in length, the result would be a name over 255 octets. If this occurs the server returns an RCODE of YXDOMAIN [RFC2136]. The DNAME record and its signature (if the zone is signed) are included in the answer as proof for the YXDOMAIN (value 6) RCODE.

2.3. DNAME Owner Name Matching the QNAME

Unlike a CNAME RR, a DNAME RR redirects DNS names subordinate to its owner name; the owner name of a DNAME is not redirected itself. The domain name that owns a DNAME record is allowed to have other resource record types at that domain name, except DNAMEs, CNAMEs or other types that have restrictions on what they can co-exist with. When there is a match of the QTYPE to a type (or types) also owned by the owner name the response is sourced from the owner name. E.g., a QTYPE of ANY would return the (available) types at the owner name, not the target name.

DNAME RRs MUST NOT appear at the same owner name as an NS RR unless the owner name is the zone apex as this would constitute data below a zone cut.

If a DNAME record is present at the zone apex, there is still a need to have the customary SOA and NS resource records there as well. Such a DNAME cannot be used to mirror a zone completely, as it does not mirror the zone apex.

These rules also allow DNAME records to be queried through RFC 1034 [RFC1034] compliant, DNAME-unaware caches.

2.4. Names Next to and Below a DNAME Record

Resource records MUST NOT exist at any sub-domain of the owner of a DNAME RR. To get the contents for names subordinate to that owner name, the DNAME redirection must be invoked and the resulting target queried. A server MAY refuse to load a zone that has data at a sub-domain of a domain name owning a DNAME RR. If the server does load the zone, those names below the DNAME RR will be occluded as described in RFC 2136 [RFC2136], section 7.18. Also a server SHOULD refuse to load a zone subordinate to the owner of a DNAME record in the ancestor zone. See Section 5.2 for further discussion related to dynamic update.

DNAME is a singleton type, meaning only one DNAME is allowed per name. The owner name of a DNAME can only have one DNAME RR, and no CNAME RRs can exist at that name. These rules make sure that for a single domain name only one redirection exists, and thus no confusion which one to follow. A server SHOULD refuse to load a zone that violates these rules.

2.5. Compression of the DNAME record.

The DNAME owner name can be compressed like any other owner name. The DNAME RDATA target name MUST NOT be sent out in compressed form,

so that a DNAME RR can be treated as an unknown type [RFC3597].

Although the previous DNAME specification [RFC2672] (that is obsoleted by this specification) talked about signaling to allow compression of the target name, such signaling has never been specified and this document also does not specify this signaling behavior.

RFC 2672 (obsoleted by this document) stated that the EDNS version had a meaning for understanding of DNAME and DNAME target name compression. This document revises RFC 2672, in that there is no EDNS version signaling for DNAME.

3. Processing

The DNAME RR causes type NS additional section processing. This refers to action at step 6 of the server algorithm outlined in section 3.2.

3.1. CNAME synthesis

When preparing a response, a server performing a DNAME substitution will in all cases include the relevant DNAME RR in the answer section. Relevant includes the following cases:

1. The DNAME is being employed as a substitution instruction.
2. The DNAME itself matches the QTYPE and the owner name matches QNAME.

When the owner name matches the QNAME and the QTYPE matches another type owned there, the DNAME is not included in the answer.

A CNAME RR with TTL equal to the corresponding DNAME RR is synthesized and included in the answer section when the DNAME is employed as a substitution instruction. The owner name of the CNAME is the QNAME of the query. The DNSSEC specification [RFC4033], [RFC4034], [RFC4035] says that the synthesized CNAME does not have to be signed. The DNAME has an RRSIG and a validating resolver can check the CNAME against the DNAME record and validate the signature over the DNAME RR.

Servers MUST be able to answer a query for a synthesized CNAME. Like other query types this invokes the DNAME, and synthesizes the CNAME into the answer. If the server in question is a cache, the synthesized CNAME's TTL SHOULD be equal to the decremented TTL of the cached DNAME.

Resolvers MUST be able to handle a synthesized CNAME TTL of zero or equal to the TTL of the corresponding DNAME record (as some older authoritative server implementations set the TTL of synthesized CNAMEs to zero). A TTL of zero means that the CNAME can be discarded immediately after processing the answer.

3.2. Server algorithm

Below is the server algorithm, which appeared in RFC 2672 Section 4.1.

1. Set or clear the value of recursion available in the response depending on whether the name server is willing to provide recursive service. If recursive service is available and requested via the RD bit in the query, go to step 5, otherwise step 2.
2. Search the available zones for the zone which is the nearest ancestor to QNAME. If such a zone is found, go to step 3, otherwise step 4.
3. Start matching down, label by label, in the zone. The matching process can terminate several ways:
 - A. If the whole of QNAME is matched, we have found the node.

If the data at the node is a CNAME, and QTYPE does not match CNAME, copy the CNAME RR into the answer section of the response, change QNAME to the canonical name in the CNAME RR, and go back to step 1.

Otherwise, copy all RRs which match QTYPE into the answer section and go to step 6.
 - B. If a match would take us out of the authoritative data, we have a referral. This happens when we encounter a node with NS RRs marking cuts along the bottom of a zone.

Copy the NS RRs for the sub-zone into the authority section of the reply. Put whatever addresses are available into the additional section, using glue RRs if the addresses are not available from authoritative data or the cache. Go to step 4.

- C. If at some label, a match is impossible (i.e., the corresponding label does not exist), look to see whether the last label matched has a DNAME record.

If a DNAME record exists at that point, copy that record into the answer section. If substitution of its <target> for its <owner> in QNAME would overflow the legal size for a <domain-name>, set RCODE to YXDOMAIN [RFC2136] and exit; otherwise perform the substitution and continue. The server MUST synthesize a CNAME record as described above and include it in the answer section. Go back to step 1.

If there was no DNAME record, look to see if the "*" label exists.

If the "*" label does not exist, check whether the name we are looking for is the original QNAME in the query or a name we have followed due to a CNAME or DNAME. If the name is original, set an authoritative name error in the response and exit. Otherwise just exit.

If the "*" label does exist, match RRs at that node against QTYPE. If any match, copy them into the answer section, but set the owner of the RR to be QNAME, and not the node with the "*" label. If the data at the node with the "*" label is a CNAME, and QTYPE doesn't match CNAME, copy the CNAME RR into the answer section of the response changing the owner name to the QNAME, change QNAME to the canonical name in the CNAME RR, and go back to step 1. Otherwise, Go to step 6.

4. Start matching down in the cache. If QNAME is found in the cache, copy all RRs attached to it that match QTYPE into the answer section. If QNAME is not found in the cache but a DNAME record is present at an ancestor of QNAME, copy that DNAME record into the answer section. If there was no delegation from authoritative data, look for the best one from the cache, and put it in the authority section. Go to step 6.
5. Use the local resolver or a copy of its algorithm to answer the query. Store the results, including any intermediate CNAMEs and DNAMEs, in the answer section of the response.
6. Using local data only, attempt to add other RRs which may be useful to the additional section of the query. Exit.

Note that there will be at most one ancestor with a DNAME as described in step 4 unless some zone's data is in violation of the no-descendants limitation in section 3. An implementation might take advantage of this limitation by stopping the search of step 3c or step 4 when a DNAME record is encountered.

3.3. Wildcards

The use of DNAME in conjunction with wildcards is discouraged [RFC4592]. Thus records of the form "*.example.com DNAME example.net" SHOULD NOT be used.

The interaction between the expansion of the wildcard and the redirection of the DNAME is non-deterministic. Because the processing is non-deterministic, DNSSEC validating resolvers may not be able to validate a wildcarded DNAME.

A server MAY give a warning that the behavior is unspecified if such a wildcarded DNAME is loaded. The server MAY refuse it, refuse to load the zone or refuse dynamic updates.

3.4. Acceptance and Intermediate Storage

Recursive caching name servers can encounter data at names below the owner name of a DNAME RR, due to a change at the authoritative server where data from before and after the change resides in the cache. This conflict situation is a transitional phase that ends when the old data times out. The caching name server can opt to store both old and new data and treat each as if the other did not exist, or drop the old data, or drop the longer domain name. In any approach, consistency returns after the older data TTL times out.

Recursive caching name servers MUST perform CNAME synthesis on behalf of clients.

If a recursive caching name server encounters a DNAME RR which contradicts information already in the cache (excluding CNAME records), it SHOULD NOT cache the DNAME RR, but it MAY cache the CNAME record received along with it, subject to the rules for CNAME.

4. DNAME Discussions in Other Documents

In [RFC2181], in Section 10.3., the discussion on MX and NS records touches on redirection by CNAMEs, but this also holds for DNAMEs.

Excerpt from 10.3. MX and NS records (in RFC 2181).

The domain name used as the value of a NS resource record, or part of the value of a MX resource record must not be an alias. Not only is the specification clear on this point, but using an alias in either of these positions neither works as well as might be hoped, nor well fulfills the ambition that may have led to this approach. This domain name must have as its value one or more address records. Currently those will be A records, however in the future other record types giving addressing information may be acceptable. It can also have other RRs, but never a CNAME RR.

The DNAME RR is discussed in RFC 3363, section 4, on A6 and DNAME. The opening premise of this section is demonstrably wrong, and so the conclusion based on that premise is wrong. In particular, [RFC3363] deprecates the use of DNAME in the IPv6 reverse tree, which is then carried forward as a recommendation in [RFC4294]. Based on the experience gained in the meantime, [RFC3363] should be revised, dropping all constraints on having DNAME RRs in these zones. This would greatly improve the manageability of the IPv6 reverse tree. These changes are made explicit below.

In [RFC3363], the paragraph

"The issues for DNAME in the reverse mapping tree appears to be closely tied to the need to use fragmented A6 in the main tree: if one is necessary, so is the other, and if one isn't necessary, the other isn't either. Therefore, in moving RFC 2874 to experimental, the intent of this document is that use of DNAME RRs in the reverse tree be deprecated."

is to be replaced with the word "DELETED".

In [RFC4294], the reference to DNAME was left in as an editorial oversight. The paragraph

"Those nodes are NOT RECOMMENDED to support the experimental A6 and DNAME Resource Records [RFC3363]."

is to be replaced by

"Those nodes are NOT RECOMMENDED to support the experimental A6 Resource Record [RFC3363]."

5. Other Issues with DNAME

There are several issues to be aware of about the use of DNAME.

5.1. Canonical hostnames cannot be below DNAME owners

The names listed as target names of MX, NS, PTR and SRV [RFC2782] records must be canonical hostnames. This means no CNAME or DNAME redirection may be present during DNS lookup of the address records for the host. This is discussed in RFC 2181 [RFC2181], section 10.3, and RFC 1912 [RFC1912], section 2.4. For SRV see RFC 2782 [RFC2782] page 4.

The upshot of this is that although the lookup of a PTR record can involve DNAMEs, the name listed in the PTR record can not fall under a DNAME. The same holds for NS, SRV and MX records. For example, when punycode alternates for a zone use DNAME then the NS, MX, SRV and PTR records that point to that zone must use names without punycode in their RDATA. What must be done then is to have the domain names with DNAME substitution already applied to it as the MX, NS, PTR, SRV data. These are valid canonical hostnames.

5.2. Dynamic Update and DNAME

DNAME records can be added, changed and removed in a zone using dynamic update transactions. Adding a DNAME RR to a zone occludes any domain names that may exist under the added DNAME.

A server MUST ignore a dynamic update message that attempts to add a non-DNAME/CNAME RR at a name that already has a DNAME RR associated with that name. Otherwise, replace the DNAME RR with the DNAME (or CNAME) update RR. This is similar behavior to dynamic updates to an owner name of a CNAME RR [RFC2136].

5.3. DNSSEC and DNAME

The following subsections specify the behavior of implementations that understand both DNSSEC and DNAME (synthesis).

5.3.1. Signed DNAME, Unsigned Synthesized CNAME

In any response, a signed DNAME RR indicates a non-terminal redirection of the query. There might or might not be a server synthesized CNAME in the answer section; if there is, the CNAME will never be signed. For a DNSSEC validator, verification of the DNAME RR and then checking that the CNAME was properly synthesized is sufficient proof.

5.3.2. DNAME Bit in NSEC Type Map

In any negative response, the NSEC or NSEC3 [RFC5155] record type bit map SHOULD be checked to see that there was no DNAME that could have been applied. If the DNAME bit in the type bit map is set and the query name is a sub-domain of the closest encloser that is asserted, then DNAME substitution should have been done, but the substitution has not been done as specified.

5.3.3. DNAME Chains as Strong as the Weakest Link

A response can contain a chain of DNAME and CNAME redirections. That chain can end in a positive answer or a negative (no name error or no data error) reply. Each step in that chain results in resource records added to the answer or authority section of the response. Only if all steps are secure can the AD bit be set for the response. If one of the steps is bogus, the result is bogus.

5.3.4. Validators Must Understand DNAME

Below are examples of why DNSSEC validators MUST understand DNAME. In the examples below, SOA records, wildcard denial NSECs and other material not under discussion has been omitted or shortened.

5.3.4.1. DNAME in Bitmap Causes Invalid Name Error

```
;; Header: QR AA RCODE=3(NXDOMAIN)
;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags: do; udp: 4096

;; Question
foo.bar.example.com. IN A
;; Authority
bar.example.com. NSEC dub.example.com. A DNAME
bar.example.com. RRSIG NSEC [valid signature]
```

If this is the received response, then only by understanding that the DNAME bit in the NSEC bitmap means that foo.bar.example.com needed to have been redirected by the DNAME, the validator can see that it is a BOGUS reply from an attacker that collated existing records from the DNS to create a confusing reply.

If the DNAME bit had not been set in the NSEC record above then the answer would have validated as a correct name error response.

5.3.4.2. Valid Name Error Response Involving DNAME in Bitmap

```
;; Header: QR AA RCODE=3(NXDOMAIN)
;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags: do; udp: 4096

;; Question
cee.example.com. IN A
;; Authority
bar.example.com. NSEC dub.example.com. A DNAME
bar.example.com. RRSIG NSEC [valid signature]
```

This response has the same NSEC records as the example above, but with this query name (cee.example.com), the answer is validated, because 'cee' does not get redirected by the DNAME at 'bar'.

5.3.4.3. Response With Synthesized CNAME

```
;; Header: QR AA RCODE=0(NOERROR)
;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags: do; udp: 4096

;; Question
foo.bar.example.com. IN A
;; Answer
bar.example.com. DNAME bar.example.net.
bar.example.com. RRSIG DNAME [valid signature]
foo.bar.example.com. CNAME foo.bar.example.net.
```

The response shown above has the synthesized CNAME included. However, the CNAME has no signature, since the server does not sign online. So this response cannot be trusted. It could be altered by an attacker to be foo.bar.example.com CNAME bla.bla.example. The DNAME record does have its signature included, since it does not change. The validator must verify the DNAME signature and then recursively resolve further to query for the foo.bar.example.net A record.

6. IANA Considerations

The DNAME Resource Record type code 39 (decimal) originally has been registered by [RFC2672]. IANA should update the DNS resource record registry to point to this document for RR type 39.

7. Security Considerations

DNAME redirects queries elsewhere, which may impact security based on policy and the security status of the zone with the DNAME and the

redirection zone's security status. For validating resolvers, the lowest security status of the links in the chain of CNAME and DNAME redirections is applied to the result.

If a validating resolver accepts wildcarded DNAMEs, this creates security issues. Since the processing of a wildcarded DNAME is non-deterministic and the CNAME that was substituted by the server has no signature, the resolver may choose a different result than what the server meant, and consequently end up at the wrong destination. Use of wildcarded DNAMEs is discouraged in any case [RFC4592].

A validating resolver MUST understand DNAME, according to [RFC4034]. The examples in Section 5.3.4 illustrate this need.

8. Acknowledgments

The authors of this draft would like to acknowledge Matt Larson for beginning this effort to address the issues related to the DNAME RR type. The authors would also like to acknowledge Paul Vixie, Ed Lewis, Mark Andrews, Mike StJohns, Niall O'Reilly, Sam Weiler, Alfred Hoenes and Kevin Darcy for their review and comments on this document.

9. References

9.1. Normative 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.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC2136] Vixie, P., Thomson, S., Rekhter, Y., and J. Bound, "Dynamic Updates in the Domain Name System (DNS UPDATE)", RFC 2136, April 1997.
- [RFC2181] Elz, R. and R. Bush, "Clarifications to the DNS Specification", RFC 2181, July 1997.
- [RFC2782] Gulbrandsen, A., Vixie, P., and L. Esibov, "A DNS RR for specifying the location of services (DNS SRV)", RFC 2782, February 2000.
- [RFC3597] Gustafsson, A., "Handling of Unknown DNS Resource Record

(RR) Types", RFC 3597, September 2003.

- [RFC4033] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "DNS Security Introduction and Requirements", RFC 4033, March 2005.
- [RFC4034] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "Resource Records for the DNS Security Extensions", RFC 4034, March 2005.
- [RFC4035] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "Protocol Modifications for the DNS Security Extensions", RFC 4035, March 2005.
- [RFC4592] Lewis, E., "The Role of Wildcards in the Domain Name System", RFC 4592, July 2006.
- [RFC5155] Laurie, B., Sisson, G., Arends, R., and D. Blacka, "DNS Security (DNSSEC) Hashed Authenticated Denial of Existence", RFC 5155, March 2008.

9.2. Informative References

- [RFC1912] Barr, D., "Common DNS Operational and Configuration Errors", RFC 1912, February 1996.
- [RFC2672] Crawford, M., "Non-Terminal DNS Name Redirection", RFC 2672, 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.
- [RFC4294] Loughney, J., "IPv6 Node Requirements", RFC 4294, April 2006.

Authors' Addresses

Scott Rose
NIST
100 Bureau Dr.
Gaithersburg, MD 20899
USA

Phone: +1-301-975-8439
Fax: +1-301-975-6238
EMail: scottr.nist@gmail.com

Wouter Wijngaards
NLnet Labs
Science Park 140
Amsterdam 1098 XG
The Netherlands

Phone: +31-20-888-4551
EMail: wouter@nlnetlabs.nl

Internet Draft
<draft-kitamura-ipv6-simple-dns-query-00.txt>

H. Kitamura
NEC Corporation
S. Ata
Osaka City University
October 18, 2010

Expires March 2011

Simplified DNS Query under IPv4/IPv6 Mixed Environment
<draft-kitamura-ipv6-simple-dns-query-00.txt>

Status of this Memo

This Internet-Draft is submitted to IETF in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

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."

The list of current Internet-Drafts can be accessed at
<http://www.ietf.org/ietf/lid-abstracts.txt>.

The list of Internet-Draft Shadow Directories can be accessed at
<http://www.ietf.org/shadow.html>.

This Internet-Draft will expire on January 2010.

Copyright Notice

Copyright (c) 2009 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document.

Abstract

This document discusses a simplified regular DNS query (resolving from a domain name to IP address(es)) method under IPv4/IPv6 mixed environment.

Under IPv4/IPv6 mixed environment, in order to obtain IPv4 and IPv6 addresses of the node with its one domain name argument by the DNS query(ies), it requires for a client to issue two times of DNS queries transaction (one for A(IPv4) record query, the other for AAAA(IPv6) record query). In shortly to say the current DNS query method: "Two DNS queries transaction is required for One domain name resolving."

Two DNS queries transaction method is complicated, inefficient and problematic. It is clear that this two DNS queries method is not suitable and not optimized for current IPv4/IPv6 mixed environment, and this method will never last to the future IPv6 fully deployed environment.

Goals of this document are:

1. to clarify the problems of current regular DNS query method
2. to propose a simplified regular DNS query method
("One DNS query transaction for One domain name resolving")

1. Introduction

This document discusses a simplified regular DNS query (resolving from a domain name to IP address(es)) method under IPv4/IPv6 mixed environment.

It is becoming common to set both IPv4 and IPv6 addresses to one node in current IPv4/IPv6 mixed environment. In typical or normal situation, the both IPv4 and IPv6 addresses information of the node are registered to the DNS DB. In other words; for the node (one domain name), both A (IPv4) record entry and AAAA (IPv6) record entry exist in the DNS DB.

Under such IPv4/IPv6 mixed environment, in order to obtain IPv4 and IPv6 addresses of the node with its one domain name argument by the DNS query(ies), it requires for a client to issue two times of DNS queries transaction (one for A(IPv4) record query, the other for AAAA(IPv6) record query). In shortly to say the current DNS query method: "Two DNS queries transaction is required for One domain name resolving."

There are several historical reasons why we use this two DNS queries transaction method. However, it has proved until now that this two DNS queries method is complicated, inefficient and problematic. It is clear that this two DNS queries method is not suitable and not optimized for current IPv4/IPv6 mixed environment, and this method will never last to the future IPv6 fully deployed environment.

Goals of this document are the followings:

1. to clarify the problems of current regular DNS query method
2. to propose a simplified regular DNS query method
("One DNS query transaction for One domain name resolving")

2. Analysis of current regular DNS query method

Hereafter, we will discuss the following typical example case on one node under IPv4/IPv6 mixed environment.

```
Domain Name:      "hostX"
IPv4 addresses:  p, q
IPv6 addresses:  s, t
```

All together four IP addresses are set to one node whose domain name is "hostX" and these addresses and domain name information is registered.

Fig. 1 shows how these IP addresses and domain name of this node are registered in the DNS DB (example of zone file configuration).

```
hostX IN A      p(IPv4 address)
hostX IN A      q(IPv4 address)
hostX IN AAAA   s(IPv6 address)
hostX IN AAAA   t(IPv6 address)
```

Fig. 1 DNS DB situation: both IPv4 and IPv6 addresses are registered as A and AAAA record entries respectively.

Fig. 2 shows typical current regular DNS query method. Two times of DNS queries (one for A (IPv4) record entries, the other for AAAA (IPv6) record entries) are issued from a client in order to resolve One domain name "hostX".

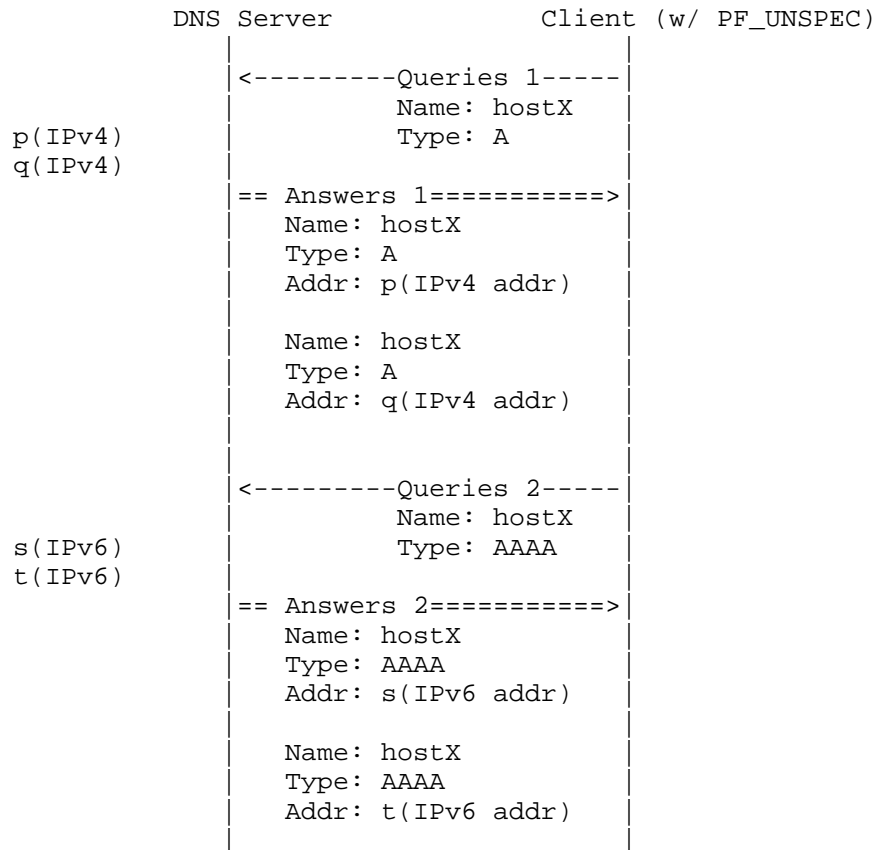


Fig. 2 "Two DNS queries transaction" for One domain name resolving.

An idea of the "Two DNS queries" transaction method is invented from an idea (keep the existing DNS query for A record as it is, and add a new DNS query for AAAA record.).

Fig. 2 only shows one type of typical DNS query methods;
 order of DNS queries: [1st for A record, 2nd for AAAA record],
 queries issued style: [series]

With some OSes, DNS queries implementation methods are different. Table 1 shows the categorization of known implementation method types.

Table 1 current "Two DNS queries transaction" method types

Type	1st query	2nd query	style
A-AAAA Series	for A record	for AAAA record	Series
A-AAAA Parallel	for A record	for AAAA record	Parallel
AAAA-A Series	for AAAA record	for A record	Series
AAAA-A Parallel	for AAAA record	for A record	Parallel

Some OSes choose more complicated or advanced method (whether 2nd query is issued or not depends on the result of 1st query or configuration status of the node.)

3. Problems of the current Two DNS queries transaction method

Two DNS queries transaction method has the following problems.

Compared to the "One" DNS query transaction method, "Two" DNS queries transaction method has the following problems.

1. Complicated (not Simple enough)
2. Inefficient
3. Takes Long time to get result
4. Issues Twice large traffics

As above section shows that combination of Two queries is many and complicated. In order to obtain the resolving results, we have to wait for two answers for two queries. It takes much time to wait for two answers. If either one of two answers is lost, too complicated recover procedures are requires.

It is clear the Two DNS queries transaction method is problematic and not recommendable. This method will never last to the future IPv6 fully deployed environment. Current Two DNS queries transaction method should be improved.

One DNS queries transaction for One domain name resolving method solves all of above problems and we propose it in the next section.

4. Proposed three types of "One DNS query transaction for One domain name resolving" methods.

We can design the following three types of DNS query methods to achieve the goal "One DNS query transaction for One domain name resolving".

1. Two record types (AAAA and A) set Queries
2. One special new record type (e.g., AAAA+A) set Queries
3. One record type (AAAA) set Queries
 with transformation of A (IPv4) record entries

4.1 Two record types (AAAA and A) set Queries

Fig. 3 shows Two record types (AAAA and A) set Queries method.

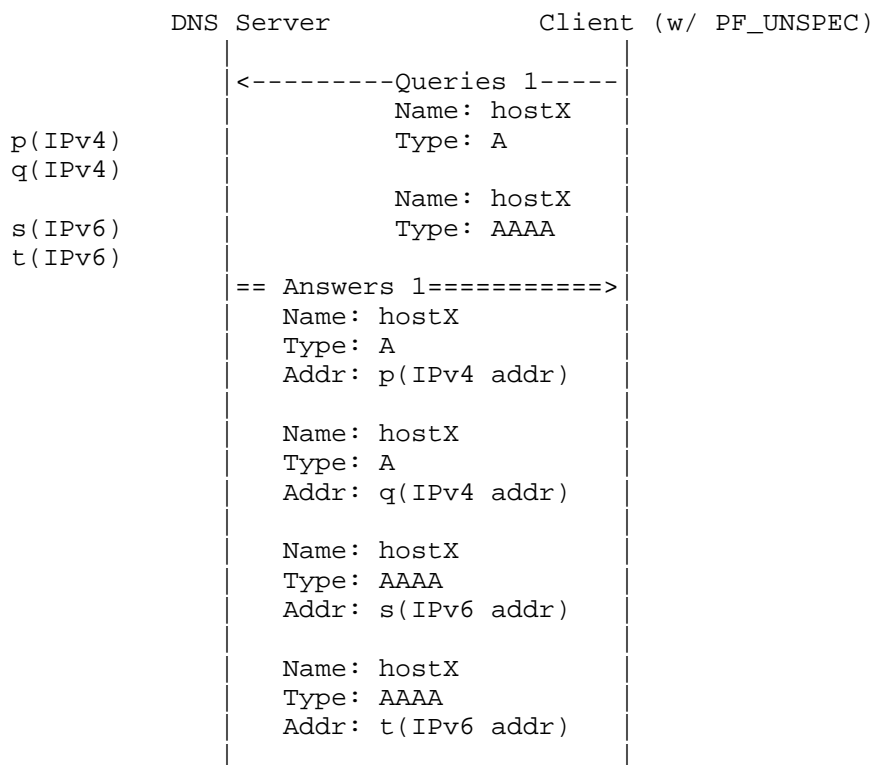


Fig. 3 Two record types (AAAA and A) set Queries

Advantages:

Natural extension method

Disadvantages:

Client side implementation modifications are necessary.

Non standard DNS query usage. (Two record types set Queries DNS

query examples are not known till now.)

4.2 One special new record type (e.g., AAAA+A) set Queries

Fig. 4 shows One special new record type (e.g., AAAA+A) set Queries method. (This function may became possible by introducing new "pseudo" Resource Record type instead of new record type.)

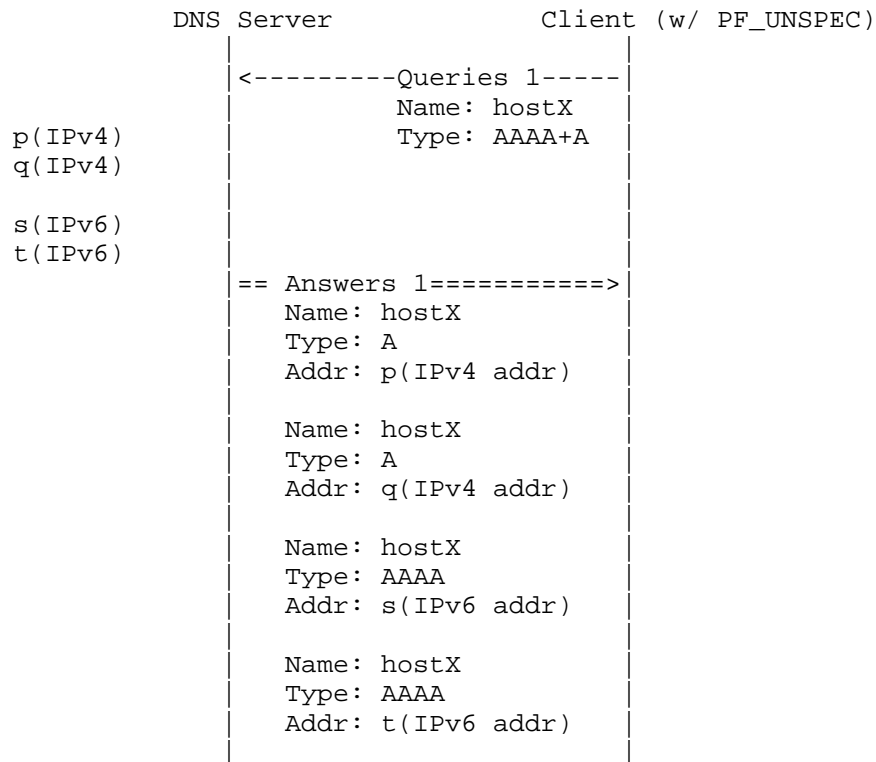


Fig. 4 One special new record type (AAAA+A) set Queries

Advantages:

Standard DNS query usage. (One record type set Queries)

Disadvantages:

Client side implementation modifications are necessary.

Special record type (e.g., AAAA+A) should be newly introduced.

4.3 One record type (AAAA) set Queries with transformation of A (IPv4) record entries

Fig. 5 shows One record type (AAAA) set Queries with transformation of A (IPv4) record entries method. This idea is completely new.

All of existing A (IPv4) record entries are assumed to be AAAA (IPv6) entries by using "IPv4 mapped IPv6 address" transformation.

If "IPv4 mapped IPv6 address" function is installed and deployed on Client, it is not necessary to install additional reverse transformation function on the Client side.

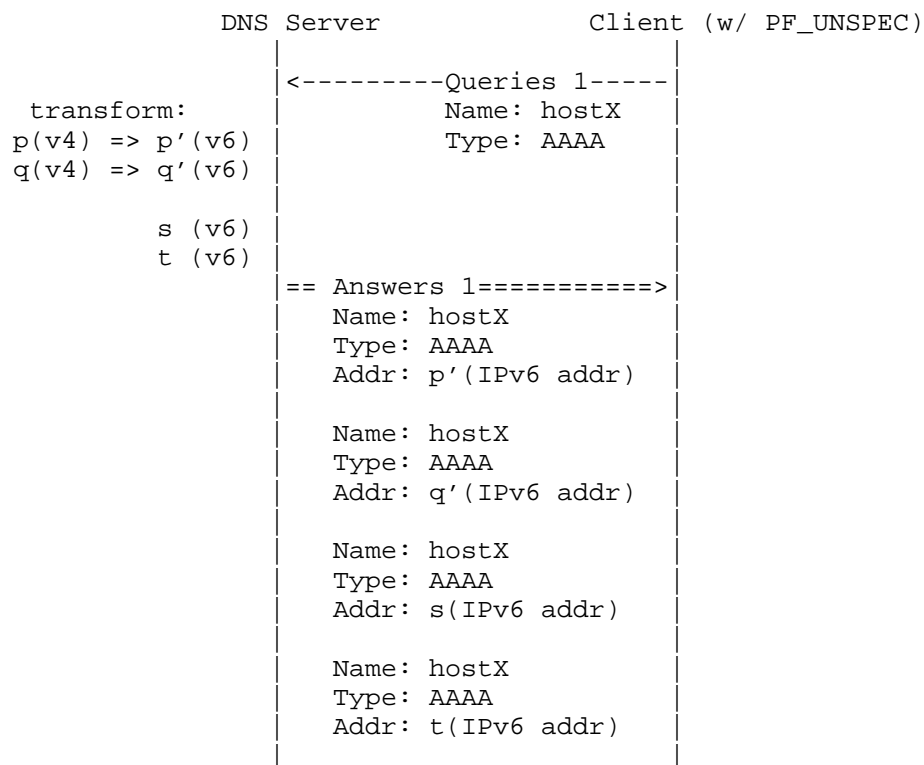


Fig. 5 One record type (AAAA) set Queries with transformation of A (IPv4)

Advantages:

NO client side implementation modifications are necessary.

Disadvantages:

"IPv4 mapped IPv6 address" function should be fully deployed on the Client side.

5. Security Considerations

The proposed DNS query methods in this document is based on simplification or natural extension of the current DNS query method. Security considerations of the proposed methods are as same as those of the current normal DNS query method. There are no newly occurred security considerations in this document.

6. IANA Considerations

This document describes a new special DNS record type in one of the proposed DNS query methods. So, the value of this new special DNS record type should be managed by the IANA.

References

Normative References

- [RFC4291] R. Hinden and S. Deering, "IP Version 6 Addressing Architecture", RFC 4291, February 2006
- [RFC4038] M-K. Shin, et al., "Application Aspects of IPv6 Transition", RFC 4038, March 2005
- [RFC1034] P. Mockapetris, "Domain names - concepts and facilities", RFC 1034, November 1987
- [RFC1035] P. Mockapetris, "Domain names - implementation and specification", RFC 1035, November 1987
- [RFC2874] M. Crawford and C. Huitema, "DNS Extensions to Support IPv6 Address Aggregation and Renumbering", RFC 2874, July 2000
- [RFC3596] S. Thomson, C. Huitema, V. Ksinant and M. Souissi, "DNS Extensions to Support IP Version 6", RFC 3596, October 2003
- [RFC4038] M-K. Shin, et al., "Application Aspects of IPv6 Transition", RFC 4038, March 2005
- [RFC4291] R. Hinden and S. Deering, "IP Version 6 Addressing Architecture", RFC 4291, February 2006
- [RFC4472] A. Durand, et al., "Operational Considerations and Issues with IPv6 DNS", RFC 4472, April 2006

Authors' Addresses

Hiroshi Kitamura
Service Platform Research Laboratories, NEC Corporation
(SC building 12F)1753, Shimonumabe, Nakahara-Ku, Kawasaki,
Kanagawa 211-8666, JAPAN
Graduate School of Information Systems,
University of Electro-Communications
5-1 Chofugaoka 1-Chome, Chofu-shi, Tokyo 182-8585, JAPAN
Phone: +81 44 431 7686
Fax: +81 44 431 7680
Email: kitamura@da.jp.nec.com

Shingo Ata
Graduate School of Engineering, Osaka City University
3-3-138, Sugimoto, Sumiyoshi-Ku, Osaka 558-8585, JAPAN
Phone: +81 6 6605 2191
Fax: +81 6 6605 2191
Email: ata@info.eng.osaka-cu.ac.jp

DNSEXT Working Group
INTERNET-DRAFT
<draft-vixie-dnsext-resimprove-00.txt>
Intended Status: For Your Information

P. Vixie, ISC
R. Joffe, Centergate
F. Neves, Registro
June 22, 2010

Improvements to DNS Resolvers
for Resiliency, Robustness, and Responsiveness

Status of this Memo

This Internet-Draft is submitted to IETF in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

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."

The list of current Internet-Drafts can be accessed at
<http://www.ietf.org/lid-abstracts.txt>.

The list of Internet-Draft Shadow Directories can be accessed at
<http://www.ietf.org/shadow.html>

This Internet-Draft will expire on December 31, 2010.

Copyright (c) 2010 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Abstract

This document describes several mechanisms which can be employed by iterative caching DNS resolvers to improve resiliency, robustness, and responsiveness. These improvements are optional and they require no changes to the protocol, or to authority servers, or to DNS stub resolver clients.

1. Introduction

1.1. Iterative caching DNS resolvers can be both compliant and interoperable without also being optimal. Indeed a wide range of optimizations, mechanisms, and outright "tricks" can be employed without affecting correctness or interoperability. Such optimizations could be recommended but never required.

1.2. This document describes several practices which can improve the resilience, robustness, and responsiveness of iterative caching DNS resolvers. These practices are not required for correctness or interoperability, and no other DNS protocol agent need be modified to gain the prospective benefits of implementing these practices.

1.3. Three practices are described here:

- A. Revalidating a delegation when a parent NS RRset TTL expires.
- B. Stopping a downward cache search when an NXDOMAIN is encountered.
- C. Upgrading the credibility of NS RRsets upon delegation events.

These practices are described in detail in later sections of this document. While they are described together this single document for editorial convenience, and are known to work well together, they are in no way interdependent.

2. Delegation Revalidation Upon NS RRSet Expiry

2.1. Because the delegating NS RRset at the bottom of the parent zone and the apex NS RRset in the child zone are unsynchronized, the TTL of the parent's delegating NS RRset is meaningless. A child zone's apex NS RRset is authoritative and thus has a higher cache credibility than the parent's delegating NS RRset, so, the NS RRset "below the cut" immediately replaces the parent's delegating NS RRset in cache when an iterative caching DNS resolver crosses a zone cut.

2.2. The lowest TTL found in a parent zone's delegating NS RRset should be stored in the cache and used to trigger delegation

revalidation as follows. Whenever a cached RRset is being considered for use in a response, the cache should be walked upward toward the root, looking for expired delegations. At the first expired delegation encountered while walking upward toward the root, revalidation should be triggered, putting the processing of dependent queries on hold until validation is complete.

2.3. To revalidate a delegation, the iterative caching DNS resolver will forward the query that triggered revalidation to the nameservers at the closest enclosing zone cut above the revalidation point. While searching for these nameservers, additional revalidations may occur, perhaps placing an entire chain of dependent queries on hold, unwinding in downward order as revalidations closer to the root must be complete before revalidations further from the root can begin.

2.4. If a delegation can be revalidated at the same node, then the old apex NS RRset should be deleted from cache and then the new delegating NS RRset should be stored in cache. The minimum TTL from the new delegating NS RRset should also be stored in cache to facilitate future revalidations. This order of operations ensures that the RRset credibility rules do not prevent the new delegating NS RRset from entering the cache. It is expected that the child's apex NS RRset will rapidly replace the parent's delegating NS RRset as soon as iteration restarts after the revalidation event.

2.5. If the new delegating NS RRset cannot be found (RCODE=NXDOMAIN) or if there is a new zone cut at some different level of the hierarchy (insertion or deletion of a delegation point above the revalidation point) or if the new RRset shares no nameserver names in common with the old one (indicating some kind of redelegation, which is rare) then the cache should be purged of all names and RRsets at or below the revalidation point. This facilitates redelegation or revocation of a zone by a parent zone administrator, and also conserves cache storage by deleting unreachable data.

2.6. To make the timing of a revalidation event unpredictable from the point of view of a potential cache-spoof attacker, the parent's delegating NS RRset TTL should be reduced by a random fraction of its value before being stored for use in revalidation activities.

3. Stopping Downward Cache Search on NXDOMAIN

3.1. In virtually all existing resolvers, a cached NXDOMAIN is not considered "proof" that there can be no child domains underneath. This is due to an ambiguity in RFC 1034 that failed to distinguish

empty nonterminal domain names from nonexistent names. For DNSSEC, the IETF had to distinguish this case, but the implication on non-DNSSEC resolvers wasn't fully realized.

3.2. When searching downward in its cache, an iterative caching DNS resolver should stop searching if it encounters a cached NXDOMAIN. The response to the triggering query should be NXDOMAIN.

3.3. When an iterative caching DNS resolver stores an NXDOMAIN in its cache, all names and RRsets at or below that node should be deleted since they will have become unreachable.

3.4. By implication, a stream of queries FOO.TLD, BAR.FOO.TLD where FOO.TLD does not exist would normally cause both queries to be forwarded to TLD's nameservers. Following this recommended practice, the second query and indeed any other query for names at or below FOO.TLD would not be forwarded.

4. Upgrading NS RRset Credibility Upon Delegation Events

4.1. Noting that a parent's delegating NS RRset is nonauthoritative "glue" whereas a child's apex NS RRset is authoritative, the latter will replace the former in cache whenever the latter is encountered. However, it is not mandatory for the child zone's nameservers to include the apex NS RRset in responses, thus it is possible for an iterative caching DNS resolver to never learn the authoritative NS RRset for a zone.

4.2. When a delegation response is received during iteration, a validation query should be sent in parallel with the forwarding of the triggering query to the delegated nameservers for the newly discovered zone cut. The response to the triggering query should be delayed until both the forwarded query and the validation query have been answered.

4.3. A validation query consists of a query for the child's apex NS RRset, sent to the newly discovered delegation's nameservers. Normal iterative logic applies to the processing of responses to validation queries, including storing the results in cache, propagating NXDOMAIN back to the triggering query, trying the next server on SERVFAIL or timeout, and so on.

4.4. If there are no nameserver names in common between the child's apex NS RRset and the parent's delegation NS RRset, then the responses received from forwarding the triggering query to the

parent's delegated nameservers should be discarded after validation, and this query should be forwarded again to the child's apex nameservers.

5. Security Considerations

5.1. A successful cache exploit which inserted a fake NXDOMAIN can deny more service from an iterative caching DNS resolver that implements the recommendation to stop downward searches when a cached NXDOMAIN is encountered.

5.2. A perfectly timed cache exploit which inserted a fake NS RRset during a delegation validation event could cause one fewer good response to be heard (per TTL expiry interval) from an iterative caching DNS resolver that implements the recommendation to validate delegations.

IANA Considerations

None.

Normative References

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

Authors' Addresses

Paul Vixie

Internet Systems Consortium
950 Charter Street
Redwood City, CA, USA
EMail: vixie@isc.org

Rodney Joffe

Centergate Research Group, LLC
420 S Smith Rd
Tempe, AZ 85281 USA
EMail: rjoffe@centergate.com

Frederico A. C. Neves

NIC.br / Registro.br
Av. das Nacoes Unidas, 11541, 7
Sao Paulo, SP 04578-000 BR
EMail: fneves@registro.br

Network Working Group
Internet-Draft
Intended status: Informational
Expires: April 28, 2011

S. Woolf
Internet Systems Consortium, Inc.

J. Yao

X. Lee

CNNIC

October 25, 2010

Problem Statement: DNS Resolution of Aliased Names
draft-yao-dnsex-identical-resolution-02.txt

Abstract

This document attempts to describe a set of issues that arises from the desire to treat a set or group of names as "aliases" of each other, "bundled," "variants," or "the same," which is problematic in terms of corresponding behavior for DNS labels.

With the emergence of internationalized domain names, among other potential use cases, two or more names that users will regard as having identical meaning will sometimes require corresponding behavior in the DNS. It's not clear how to accommodate these requirements for behavior of such names in DNS resolution; in particular, it's not clear when they are best accommodated in registry practices for generating names for lookup in the DNS, existing DNS protocol elements and behavior, or some set of protocol elements or behavior not yet defined. This document attempts to describe some of these cases and the behavior of some of the possible solutions discussed to date.

NOTE: Even more than usual, version -02 of this document is a "work in progress". Additional updates may be expected between the date of this document and the DNSEXT meeting in Beijing, and can be found at <http://users.isc.org/~woolf>.

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 April 28, 2011.

Copyright Notice

Copyright (c) 2010 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

This document may contain material from IETF Documents or IETF Contributions published or made publicly available before November 10, 2008. The person(s) controlling the copyright in some of this material may not have granted the IETF Trust the right to allow modifications of such material outside the IETF Standards Process. Without obtaining an adequate license from the person(s) controlling the copyright in such materials, this document may not be modified outside the IETF Standards Process, and derivative works of it may not be created outside the IETF Standards Process, except to format it for publication as an RFC or to translate it into languages other than English.

Table of Contents

1. Introduction	4
1.1. What this document does	5
1.2. What this document does not do	5
1.3. Terminology	5
2. Problem Statement	6
2.1. Registration of Domain Name Variants	7
2.2. Identical DNS Resolution for Bundled DNS Names	7
2.3. Character Variants	8
2.3.1. An example: Simplified and Traditional Chinese	8
2.3.2. An example: Greek	9
2.3.3. An Example: Arabic	9
3. Possible Solutions	9
3.1. Mapping or Redirection of Domain Names	10
3.1.1. Mapping itself	10
3.1.2. Mapping its descendants	10
3.1.3. Mapping itself and its descendants	10
3.2. Zone Clone	11
4. IANA Considerations	12
5. Security Considerations	12
6. Acknowledgements	13
7. Change History	13
7.1. draft-yao-dnsex-identical-resolution: Version 00	13
7.2. draft-yao-dnsex-identical-resolution: Version 01	13
8. References	13
8.1. Normative References	13
8.2. Informative References	14
Authors' Addresses	15

1. Introduction

As the Internet and the DNS have evolved beyond their original realms of use, a set of needs and expectations has appeared about how DNS labels behave that is informed significantly by common human assumptions about how "names" or "words" work. One aspect of this is the notion or expectation that multiple sets of names may be similar to a human user, and expected to behave "the same" or as "aliases" of one another. The DNS was designed with the implicit expectation that names would be based on ASCII characters, and the "similarity" or "sameness" property doesn't seem to arise terribly often in the names people originally wanted to use in the DNS; thus the requirements of identical resolution of "aliased" or "bundled" names hasn't figured prominently as an attribute that needed to be accommodated in the generation or lookup of DNS names. However, with the standardization of internationalized domain names protocols (ref: IDNA and IDNAbis), more and more internationalized domain name labels [RFC3490] are appearing in DNS zones. In some cases, these labels [RFC3743] are accompanied by the expectation that they are "equivalent" or should behave "the same," often because these labels are derived from names or strings that users consider "the same" in some languages. Accordingly, Internet users hope for such labels to behave in DNS contexts as they expect the corresponding human constructs to behave.

The general issues of what "the same" means, or of defining "variants" in human scripts as codified in Unicode (or anywhere else) are well outside the scope of the DNS or the expertise of most of the people who work on it. They are matters for philosophers and applications developers, respectively. However, to the extent that these issues can be specified as involving the resolution of names in the DNS, it's reasonable to describe those expectations and attempt to accomodate them.

There is some existing technology defined in the DNS for behavior that can be described as one name behaving "the same" as another. For a single node in the DNS tree, CNAME can be used to map one name as an "alias" to another, "canonical" name. If there is a need to map a subtree of the DNS-- a zone, or a domain and its subdomains-- to another domain, DNAME has been defined to allow this behavior. However, there is no way currently defined to do both, as CNAME is required to be the record at its node in the tree. Behavior that combines their characteristics is not currently defined in the DNS.

If existing protocol does not meet the zone administrator's need to be able to treat one label, name, or zone as "the same" as another, there are also administrative mechanisms available for manipulating databases underlying the generation and resolution of DNS names. Registry operators have many mechanisms for working around DNS

protocol in order to get behavior they want for names in DNS zones, and management of "aliases" is no exception. However, it is not clear how much of the user and operator requirements for "aliases" can be met by mechanisms for provisioning DNS zones, at acceptable cost. Concerns have been raised about this approach particularly at large scales.

1.1. What this document does

Attempts to think about "aliases" or similar concepts as applied to the DNS have been difficult, both because use cases have been unclear and because terminology for describing and distinguishing them has not been readily available. This document attempts to provide both brief descriptions of identified use cases, and a rough organization for how to think about behavior in the DNS that might correspond to the requirements derived from them as a way of evaluating proposed solutions. This includes existing and additional possible solutions.

As a departure point, we attempt to be rigorous about distinguishing DNS "labels" from "words" (a human construct) and "strings" (which we use here as machine-readable constructs that nonetheless may not conform to DNS label constraints, such as IDNA U-labels).

We also review existing constructs (CNAME, DNAME) and proposed new ones ("BNAME," "zone clones").

1.2. What this document does not do

This document makes no attempt to solve or even describe "translation" of one name into another in the DNS, which is likely to be impossible. "Translation" in general, or even the particular problem of determining when or why two DNS labels (or even FQDNs) should be considered "the same", are simply not in scope for the DNS protocol. We pre-suppose those decisions are made elsewhere and that the DNS needs to deliver behavior in conformance with that external decision.

Accordingly, this document makes no comment on policy regarding when two names are "the same," what restrictions should be placed on their generation or use outside those imposed by the DNS protocol, or the ability of one approach over another to instantiate what a given user regards as "the same" for a language, script, culture, or community.

1.3. Terminology

All the basic terms used in this specification are defined in the documents [RFC1034], [RFC1035], [RFC2672] and [RFC3490].

We also note that there is a wide variety of terminology in use to describe the issues we attempt to treat in this document, and no consensus on which apply under what conditions. Terms for "a set of domain names that somehow need to be treated as similar" include "bundle," "variants," or "clones". As uniformity of terms is one of the goals of any work on this topic in the DNS, we try not to add to the confusion in the problem statement but can't claim to have finalized a recommendation in early versions of this document.

2. Problem Statement

From the point of view of the DNS, a number of attributes suggest themselves as important dimensions for evaluating what "the same" might mean.

One question is exactly what it is that's to be defined as "the same"? Are the end results to be identical, and if so from what perspective: that of the recursive resolver? The application? The human consumer of content? Is it enough that lookups on the FQDN portion of an email address result in the same A or AAAA records, or does some intermediate mapping need to be maintained between MX records in the resolution chain? What about the FQDN portion of a URL handed back to an application, or in resolution processes that include multiple lookups of records that may include FQDNs? Do there need to be general rules specified for the handling of FQDNs in RDATA of present and future RRtypes?

Another question is the behavior of multiple names with respect to one another: is it enough to define one as "canonical" or "preferred," with the others considered as "variants" that are transformed to the "preferred" form? Or is there a real need for multiple names to be "equivalent", interchangeable, with none considered "preferred" over the others? (We note here that no requirement for complete interchangeability or identity has been articulated, except anecdotally, and such equivalence would be extremely difficult to define in the DNS.)

In addition, the tree structure of the DNS requires that we consider the behavior of "identical" names across multiple zones in the hierarchy. Are mappings to be maintained in names more than a level, or two, deep? If so, with what characteristics, and what characteristics are required for scalability?

A further question arises with respect to how applications should interact with alias-specific DNS behavior. A basic requirement would seem to be "First, do no harm," or in other words, any extensions to DNS protocol in support of the desired "alias" behavior should not

interfere with applications that expect to do such interpretation on their own. This concern is based in the expectation that DNS is simple and predictable, operating strictly as infrastructure under the process of creating "the user experience," not as part of it.

2.1. Registration of Domain Name Variants

The introduction of IDN has provided a forcing function for defining how "variants" might behave as DNS names. It's generally conceded that recognition and careful management of cases where multiple names are associated together as "variants" in the expectation or preference of users are important; without such management of grouped domain names, security risks may be increased and the quality of user experience may be compromised. [RFC3743] developed by JET (Joint Engineering Team) gives one possible solution of how to manage registration of a domain name, intended to be applied to the script and usage common across Chinese, Japanese, and Korean users. [RFC3743] proposed an algorithm which will allocate a group of names, consisting of a domain name and its variants, to the same domain holder. It means that the domain holder will get control of the domain name and its variants. [RFC4290] suggests the practice in [RFC3743] to be used in registrations of internationalized domain names. But [RFC3743] and [RFC4290] do not define how, exactly, these bundles of names are to be treated by the registry or the DNS in order to obtain the desired "identical" behavior. [RFC4690] said that the "variant" model introduced in [RFC3743] and [RFC4290] can be used by a registry to prevent the most negative consequences of possible confusion, by ensuring either that both names are registered to the same party in a given domain or that one of them is completely prohibited. The principles described in [RFC3743], [RFC4290] and [RFC4690] have been accepted by many registries. But the technical details of how to guarantee that a bundle of domain names are "identical" in the DNS remain unspecified.

2.2. Identical DNS Resolution for Bundled DNS Names

To some extent, the desired behavior can be described: "identical DNS resolution" means that the process of resolving two domain names will end with the same result, in most cases the same IP address. In the history of DNS protocol development, there have been two attempts to specify such "identical resolution" behavior: CNAME[RFC1034] which maps or redirects itself, and DNAME[RFC2672] which maps or redirects its descendants. In the case of bundles or groups of names, however, some operators have asserted they need identical DNS resolution at all levels' domain names, including the domain name itself and its descendants. As alluded to above, registries are left with ad hoc provisioning and database management mechanisms for managing variant names, with some help from existing DNS protocol mechanisms for

mapping labels or FQDNs to each other. However, some are finding the existing mechanisms to have unsatisfactory limitations; they are seeking more guidance on the use of existing mechanisms, and perhaps the addition of new ones in the DNS protocol.

2.3. Character Variants

Many defined scripts as used in many different languages have "character variants" included. There is no uniform definition of variants, and in fact their characteristics differ widely, but it's possible to define some. For example, the definition of variant characters in the JET Guidelines [RFC3743], intended for use with the CJK language/script communities, is roughly this: One conceptual character can be identified with several different Code Points in character sets for computer use. In UNICODE definitions of some scripts, including Han (chinese), some characters can be identified as "compatibility variants" of another character, which usually implies that the first can be remapped to the second without the loss of any meaning. In this document, variant characters are two or more characters that may be similar in appearance or identical in meaning (similarity in appearance is not required by the definition but often occurs).

With the introduction of IDNs in the DNS, perhaps most prominently in the root zone, decisions about how to deal with IDN variants is a significant challenge ahead of us. We describe here a couple of examples, Chinese and Greek; [STW: ...and additional cases (Arabic, Cyrillic, etc.) should be described in future versions by experts in those scripts and languages].

2.3.1. An example: Simplified and Traditional Chinese

For example, if the IDN TLD "China" (U+4E2D U+56FD) and its variant (U+4E2D U+570B) are put into the root, the first one (U+4E2D U+56FD) can be considered the "original" IDN TLD and the second one (U+4E2D U+570B) can be considered the IDN TLD "variant". Ideally, it should be possible to treat the original IDN TLD and its IDN TLD variant as "identical" for purposes of DNS resolution, in a way similar to the case mapping most DNS users take for granted, in which the uppercase "A" is the variant of lowercase "a". For example, the string ".COM" can be considered a variant of ".com", and the corresponding DNS labels are treated as identical. However, for the historical reasons already discussed, and technical reasons having to do with the underpinnings of the IDNabis protocol (ref: IDNabis rationale), there's no generalization of the "case" mapping available for situations where it might be useful for IDNs. In addition, it's perilous because DNS rules around "case insensitivity" and "case preservation" are not intuitively consistent.

At this writing, four Chinese script IDN TLDs are in the root, including two pairs comprising a Traditional Chinese name and its Simplified counterpart, which should allow future versions of this document to include those operators' experience of managing maintenance of those names as "the same."

2.3.2. An example: Greek

In Greek, almost every word has the "tonos" accent sign, but where it is placed (on which character) can vary. Further, some words end in a final sigma, which is represented differently to sigma appearing elsewhere in the word. If a registry wishes to be able to enforce the association among all of the domain names that correspond to a "word" in Greek, with all its possible Unicode strings, some mechanism must be used to enumerate the "variant" names and tie them together. This makes sense from the human factors perspective, as depending on how the user types something, results may include a different domain to what was expected, although the user may have the firm belief that "the same word" was input in multiple cases.

As an example, the domain names "xn--0xadhj4a.gr" and "xn--0xaafjl.gr" appear to a native speaker/reader of Greek to represent "the same word," in a sense very much like the case insensitivity that native users of Latin script take for granted in the DNS.

2.3.3. An Example: Arabic

[STW: [to be added]

3. Possible Solutions

Currently, there are several possible mechanisms to support identical DNS resolution of "bundled" or "variant" names as "aliases" in the DNS. Existing mechanisms in the DNS include CNAME and DNAME. In addition, as described briefly above, registry operators have a great many techniques for applying policy to what names can be registered, and provisioning technology to how they are instantiated in the DNS, in support of keeping "variant" names behaving similarly to each other, or in preventing the use of such variants as might be considered confusing or dangerous.

In addition, there are new proposals for DNS protocol to support "aliases" in the DNS as part of the desired behavior of "variant" names: Names direction[BNAME], and "Zone clone".

All of the solutions have their advantages and disadvantages. In particular, there are a couple of limitations they all share. Every

mechanism existing or proposed to support "aliases" in the DNS requires that one name be designated as the "canonical" name ("preferred" in the terminology of the JET variant mechanism) and any others bundled with it are to be considered "variants" or "aliases". The only known way to enforce a symmetrical or equivalent association is via careful registry provisioning within and across domains. In addition, the different "alias" mechanisms differ in subtle ways that have to be carefully reviewed against the desired behavior of the DNS in support of different types of "variants".

3.1. Mapping or Redirection of Domain Names

3.1.1. Mapping itself

It was recognized as part of the original specification of the DNS that a host can have many names; in fact this expectation predates the DNS, referring to the earlier specification of host names. In the simplest case for "aliases", Internet users need these multiple names to be resolved to the same IP address by a DNS server. The CNAME record [RFC1034], where "CNAME" is an abbreviation for "Canonical Name", is a way to designate aliases of the "real" or canonical name of a host. In some cases, CNAME can be used to produce the necessary association a bundle of variant domain names. But the CNAME only maps itself, not its descendants; in fact it is defined to not have descendants, as it is the only name at a node in the DNS tree and can't exist at the same name as delegation. In the case of IDN variants, however, it is often desirable that the name map both itself and its descendants.

3.1.2. Mapping its descendants

In order to maintain the address-to-name mappings in a context of network renumbering, a DNAME record or Delegation Name record defined by [RFC2672] was invented to create an alias for all its subdomains. In contrast, the CNAME record creates an alias only of a single name (and not of its subdomains). As with the CNAME record, the DNS lookup will continue by retrying the lookup with the new name. If a DNS resolver sends a query without EDNS[EDNS0], or with EDNS version 0, then a name server synthesizes a CNAME record to simulate the semantics of the DNAME record. A DNAME record is very much like the CNAME record, but while the CNAME record only applies for one name, with a DNAME record one can create aliases for all the records for its subdomain.

3.1.3. Mapping itself and its descendants

Bundling of "variant" strings or names as domain names, possibly along with other use cases not yet identified, require the ability to

map a whole tree of the domain space to another domain. The current DNS protocols do not support this function. A new DNS resource record [BNAME] has been proposed to deal with this problem.

The advantage of BNAME is that it would enable a class of "aliasing" behavior that some operators find desirable, particularly in preference to some of the provisioning overhead they describe having to deploy to support potentially large numbers of "bundles" of variants at multiple levels of the DNS tree. The disadvantage is that it may not provide the behavior people really want while requiring the time and resources to code and deploy any new DNS facility.

Alternatively, a proposal has been made that would leave CNAME as already specified, but eliminating the constraint that a CNAME must be alone at a node in the DNS tree. This would avoid any coding and deployment overhead associated with new RRtypes, while obtaining the desired behavior. Concerns expressed about it, however, include the possible (but not yet specified) effort required for backwards compatibility to avoid harm to implementations that expect, and use, the old behavior.

3.2. Zone Clone

The proposal of "zone clone" or "dns shadow", is an alternative solution for a higher level of support than the DNS currently provides for "alias" behavior across zones. In this scheme, a new RRtype, SHADOW, is specified; it can exist at a zone apex and can be used to define "clones" or "shadows" of the zone content so that records in the zone are reachable via lookups from multiple delegations. This mechanism varies fundamentally from CNAME/DNAME/BNAME in that it creates a local copy on each cooperating authoritative server that has the original zone, reachable by the names specified in the SHADOW RR. Its scope, then, is the zone as maintained by an authoritative server rather than a single RRset (even one corresponding to a delegation).

This scheme has the advantage that it allows a SHADOW zone to be used in all the same contexts as the canonical or underlying zone, including contexts where a CNAME or DNAME (or, presumably, a BNAME) cannot appear, such as in the RDATA of certain RRtypes. Of the proposed DNS protocol mechanisms, it probably comes closest to the behavior some have requested as "equivalence," where none of the bundled or SHADOW names is canonical or preferred over the others. It does implicate an unknown level of effort to implement and support.

4. IANA Considerations

There are no obvious IANA considerations in this memo; we reiterate that the determination of which names are to be considered "the same" is explicitly out of scope.

5. Security Considerations

[STW: Looking for examples for this section.]

Unsolved issues that will have to be considered in the definition of what "the same" means for the DNS include the implications for DNSSEC, and whether "identical" resolution includes DNSSEC validation in the expected "identical" behavior.

Another area of possible peril includes SSL certificates, "Host" headers as seen by web servers, and other security-relevant data often associated with domain names. It will have to be considered whether, and how, the "sameness" property maps into the expected behavior of security-related protocols that use domain names, particularly given that it's unlikely that all operators will ever use the same set of constructs (whether in the DNS or elsewhere) to signal whether different "names" are "the same" for purposes of the function of a particular application or protocol.

In addition, there is a large cluster of security risks at the user and application levels that motivate significant portions of the interest in what it means to treat a set of names as "aliases" of each other. One set of issues is around the expectation that two strings are seen as "different" by the user in some obvious way (such as visually) but need to be treated as "the same". The potential for user confusion and subversion is not hard to imagine in cases where two visually distinct strings are nonetheless likely to be expected by the user to behave "the same" in some functional way. This is the case we have attempted to address here.

There is a separate but complementary set of issues that arise around cases where strings that look "the same" should nonetheless be treated as different-- the so-called "confusing visual similarity" problem. The easy example is substituting the Unicode codepoint for a character in one script, or a string of them, for the Unicode codepoints for similar-looking characters in an altogether different script. This has a different set of potential risks to users, and has not been discussed here. It's often closely related to the "alias" issue we have attempted to deal with, however, which poses risks of its own to analysis of the either subject.

6. Acknowledgements

Most of the ideas here and much of the text is taken from discussions on the DNSEXT and DNSOP WG mailing lists. Particular help is acknowledged from the authors of the proposed solutions drafts, and from the many contributors to the IDNabis work and its underpinnings. Special thanks at the intersection of DNS and IDNabis is owed to Patrik Faltstrom, Cary Karp, John Klensin, Vaggelis Segredakis, and Andrew Sullivan for their patient explanations.

7. Change History

[[anchor20: RFC Editor: Please remove this section.]]

7.1. draft-yao-dnsext-identical-resolution: Version 00

- o Domain Name Identical Resolution Problem Statement (initial attempt)

7.2. draft-yao-dnsext-identical-resolution: Version 01

- o Expanded introduction
- o Added Greek example
- o Added some detail to descriptions of proposed solutions

8. References

8.1. Normative References

- [ASCII] American National Standards Institute (formerly United States of America Standards Institute), "USA Code for Information Interchange", ANSI X3.4-1968, 1968.
- [EDNS0] Vixie, P., "Extension Mechanisms for DNS (EDNS0)", RFC 2671, August 1999.
- [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.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC2136] Vixie, P., Thomson, S., Rekhter, Y., and J. Bound,

"Dynamic Updates in the Domain Name System (DNS UPDATE)",
RFC 2136, April 1997.

- [RFC2672] Crawford, M., "Non-Terminal DNS Name Redirection",
RFC 2672, August 1999.
- [RFC3490] Faltstrom, P., Hoffman, P., and A. Costello,
"Internationalizing Domain Names in Applications (IDNA)",
RFC 3490, March 2003.
- [RFC3597] Gustafsson, A., "Handling of Unknown DNS Resource Record
(RR) Types", RFC 3597, September 2003.
- [RFC3629] Yergeau, F., "UTF-8, a transformation format of ISO
10646", RFC 3629, November 2003.
- [RFC3743] Konishi, K., Huang, K., Qian, H., and Y. Ko, "Joint
Engineering Team (JET) Guidelines for Internationalized
Domain Names (IDN) Registration and Administration for
Chinese, Japanese, and Korean", RFC 3743, April 2004.
- [RFC4033] Arends, R., Austein, R., Larson, M., Massey, D., and S.
Rose, "DNS Security Introduction and Requirements",
RFC 4033, March 2005.
- [RFC4034] Arends, R., Austein, R., Larson, M., Massey, D., and S.
Rose, "Resource Records for the DNS Security Extensions",
RFC 4034, March 2005.
- [RFC4035] Arends, R., Austein, R., Larson, M., Massey, D., and S.
Rose, "Protocol Modifications for the DNS Security
Extensions", RFC 4035, March 2005.
- [RFC4290] Klensin, J., "Suggested Practices for Registration of
Internationalized Domain Names (IDN)", RFC 4290,
December 2005.
- [RFC4690] Klensin, J., Faltstrom, P., Karp, C., and IAB, "Review and
Recommendations for Internationalized Domain Names
(IDNs)", RFC 4690, September 2006.

8.2. Informative References

- [BNAME] Yao, J., Lee, X., and P. Vixie, "Bundle DNS Name
Redirection", draft-yao-dnsext-bname-01.txt (work in
progress), 12 2009.
- [CNAME-DNAME]

Sury, O., "CNAME+DNAME Name Redirection",
draft-sury-dnsex-cname-dname-00.txt (work in progress),
4 2010.

[IDN-TLD-Variants]

Yao, J. and X. Lee, "IDN TLD Variants Implementation
Guideline", draft-yao-dnsop-idntld-implementation-01.txt
(work in progress), 11 2009.

[RFC2672bis]

Rose, S. and W. Wijngaards, "Update to DNAME Redirection
in the DNS", Internet-Draft ietf-dnsex-rfc2672bis-dname-
17.txt, 6 2009.

[SHADOW]

Vixie, P., "Use of DNS to Carry Configuration Metadata
Concerning Automatic Replication of Zones",
draft-vixie-dnsex-dnshadow-00.txt (work in progress),
2 2010.

Authors' Addresses

Suzanne Woolf
Internet Systems Consortium, Inc.
950 Charter St.
Redwood City, CA 94063

Phone: +1 650 423 1333
Email: woolf@isc.org

Jiankang YAO
CNNIC
No.4 South 4th Street, Zhongguancun
Beijing

Phone: +86 10 58813007
Email: yaojk@cnnic.cn

Xiaodong LEE
CNNIC
No.4 South 4th Street, Zhongguancun
Beijing

Phone: +86 10 58813020
Email: lee@cnnic.cn

