                      DNAME Redirection in the DNS

Abstract

   The DNAME record provides redirection for a subtree of the domain
   name tree in the DNS. That is, all names that end with a particular
   suffix are redirected to another part of the DNS. This document
   obsoletes the original specification in RFC 2672 as well as updates
   the document on representing IPv6 addresses in DNS (RFC 3363).

Status of This Memo

   This is an Internet Standards Track document.

   This document is a product of the Internet Engineering Task Force
   (IETF). It represents the consensus of the IETF community. It has
   received public review and has been approved for publication by the
   Internet Engineering Steering Group (IESG). Further information on
   Internet Standards is available in Section 2 of RFC 5741.

   Information about the current status of this document, any errata,
   and how to provide feedback on it may be obtained at
   http://www.rfc-editor.org/info/rfc6672.

Copyright Notice

Table of Contents

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, whereas 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.

   For example, take 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 setup, 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 subtrees of the DNS to contain the same
   information. Examples include punycode [RFC3492] alternates for
   domain spaces.

   This revision of the DNAME specification 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.

1.1. Requirements Language

   The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT",
   "SHOULD", "SHOULD NOT", "RECOMMENDED" "NOT RECOMMENDED", "MAY", and
   "OPTIONAL" in this document are to be interpreted as described in RFC
   2119 [RFC2119].

2. The DNAME Resource Record

2.1. Format

   The DNAME RR has mnemonic DNAME and type code 39 (decimal). It is
   CLASS-insensitive.

   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].
   The presentation format of the RR is as follows:

           <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 step 3 of the algorithm in RFC 1034 [RFC1034], Section
   4.3.2, "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 ensure 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 coexist 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. For
example, 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; if it is not the zone apex, then the
NS RR signifies a delegation point, and the DNAME RR must in that
case appear below the zone cut at the zone apex of the child zone.

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 caches
that are RFC 1034 [RFC1034] compliant and are DNAME unaware.

2.4. Names next to and below a DNAME Record

Resource records MUST NOT exist at any subdomain 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
subdomain 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 ought
to 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; thus, there's no
confusion about which one to follow. A server ought to 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
and MUST be downcased for DNS Security Extensions (DNSSEC)
validation.

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, nor is it specified in this document.

RFC 2672 (obsoleted by this document) states that the Extended DNS
(EDNS) version has a means for understanding DNAME and DNAME target
name compression. This document revises RFC 2672, in that there is
no EDNS version signaling for DNAME.

3. Processing

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 cases includes the following:

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 Time to Live (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 signed 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 then the server
synthesizes the CNAME and places it into the answer section. 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 a
value 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 revised version of the server algorithm, which appears
   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 that 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. Due to the fact that
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 DNSSEC validated
DNAME RR that contradicts information already in the cache (excluding
CNAME records), it SHOULD cache the DNAME RR, but it MAY cache the
CNAME record received along with it, subject to the rules for CNAME.
If the DNAME RR cannot be validated via DNSSEC (i.e., not BOGUS, but
not able to validate), the recursive caching server SHOULD NOT cache
the DNAME RR but MAY cache the CNAME record received along with it,
subject to the rules for CNAME.

3.4.1. Resolver Algorithm

Below is the revised version of the resolver algorithm, which appears
in RFC 2672, Section 4.2.

1. See if the answer is in local information or can be synthesized
    from a cached DNAME; if so, return it to the client.

2. Find the best servers to ask.

3. Send queries until one returns a response.

4. Analyze the response, either:

   A. If the response answers the question or contains a name
       error, cache the data as well as return it back to the
       client.

   B. If the response contains a better delegation to other
       servers, cache the delegation information, and go to step 2.

   C. If the response shows a CNAME and that is not the answer
       itself, cache the CNAME, change the SNAME to the canonical
       name in the CNAME RR, and go to step 1.

   D. If the response shows a DNAME and that is not the answer
       itself, cache the DNAME (upon successful DNSSEC validation if
       the client is a validating resolver). If substitution of the
       DNAME's target name for its owner name in the SNAME would
       overflow the legal size for a domain name, return an
       implementation-dependent error to the application; otherwise,
       perform the substitution and go to step 1.

   E. If the response shows a server failure or other bizarre
       contents, delete the server from the SLIST and go back to
       step 3.

4. DNAME Discussions in Other Documents

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

   Section 10.3 ("MX and NS records") of [RFC2181] states:

        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. Based on the

experience gained in the meantime, [RFC3363] is revised, dropping all
constraints on having DNAME RRs in these zones [RFC6434]. This would
greatly improve the manageability of the IPv6 reverse tree. These
changes are made explicit below.

In [RFC3363], the following paragraph is updated by this document,
and the use of DNAME RRs in the reverse tree is no longer deprecated.

   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.

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 cannot fall under a
   DNAME. The same holds for NS, SRV, and MX records. For example,
   when punycode [RFC3492] alternates for a zone use DNAME, then the NS,
   MX, SRV, and PTR records that point to that zone must use names that
   are not aliases in their RDATA. Then, what must be done is to have
   the domain names with DNAME substitution already applied to it as the
   MX, NS, PTR, and 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.

   If a dynamic update message attempts to add a DNAME with a given
   owner name, but a CNAME is associated with that name, then the server
   MUST ignore the DNAME. If a DNAME is already associated with that
   name, then it is replaced with the new DNAME. Otherwise, add the
   DNAME. If a CNAME is added with a given owner name, but a DNAME is

associated with that name, then the CNAME MUST be ignored. Similar
behavior occurs for 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 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
bitmap SHOULD be checked to see that there was no DNAME that could
have been applied. If the DNAME bit in the type bitmap is set and
the query name is a subdomain 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 reply (no name error
or no data error). Each step in that chain results in resource
records being added to the answer or authority section of the
response. Only if all steps are secure can the AD (Authentic Data)
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, SOA records, wildcard denial NSECs, and other
material not under discussion have been omitted or shortened.

5.3.4.1. Invalid Name Error Response Caused by DNAME in Bitmap

```
;; 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 in order to query for the
   foo.bar.example.net A record.

6. Examples of DNAME Use in a Zone

   Below are some examples of the use of DNAME in a zone. These
   examples are by no means exhaustive.

6.1. Organizational Renaming

   If an organization with domain name FROBOZZ.EXAMPLE.NET became part
   of an organization with domain name ACME.EXAMPLE.COM, it might ease
   transition by placing information such as this in its old zone.

```
    frobozz.example.net.  DNAME    frobozz-division.acme.example.com.
                          MX       10      mailhub.acme.example.com.
```

   The response to an extended recursive query for
   www.frobozz.example.net would contain, in the answer section, the
   DNAME record shown above and the relevant RRs for www.frobozz-
   division.acme.example.com.

   If an organization wants to have aliases for names, for a different
   spelling or language, the same example applies. Note that the MX RR
   at the zone apex is not redirected and has to be repeated in the
   target zone. Also note that the services at mailhub or www.frobozz-
   division.acme.example.com. have to recognize and handle the aliases.

6.2. Classless Delegation of Shorter Prefixes

   The classless scheme for in-addr.arpa delegation [RFC2317] can be
   extended to prefixes shorter than 24 bits by use of the DNAME record.
   For example, the prefix 192.0.8.0/22 can be delegated by the
   following records.

```
        $ORIGIN 0.192.in-addr.arpa.
        8/22    NS        ns.slash-22-holder.example.com.
        8       DNAME     8.8/22
        9       DNAME     9.8/22
        10      DNAME     10.8/22
        11      DNAME     11.8/22
```

   A typical entry in the resulting reverse zone for some host with
   address 192.0.9.33 might be as follows:

```
         $ORIGIN 8/22.0.192.in-addr.arpa.
         33.9    PTR       somehost.slash-22-holder.example.com.
```

   The advisory remarks in [RFC2317] concerning the choice of the "/"
   character apply here as well.

6.3. Network Renumbering Support

   If IPv4 network renumbering were common, maintenance of address space
   delegation could be simplified by using DNAME records instead of NS
   records to delegate.

```
        $ORIGIN new-style.in-addr.arpa.
        189.190           DNAME     in-addr.example.net.

        $ORIGIN in-addr.example.net.
        188               DNAME     in-addr.customer.example.com.

        $ORIGIN in-addr.customer.example.
        1                 PTR       www.customer.example.com
        2                 PTR       mailhub.customer.example.com.
        ; etc ...
```

   This would allow the address space 190.189.0.0/16 assigned to the ISP
   "example.net" to be changed without having to alter the zone data
   describing the use of that space by the ISP and its customers.

Renumbering IPv4 networks is currently so arduous a task that
updating the DNS is only a small part of the labor, so this scheme
may have a low value. But it is hoped that in IPv6 the renumbering
task will be quite different, and the DNAME mechanism may play a
useful part.

7. IANA Considerations

The DNAME resource record type code 39 (decimal) originally was
registered by [RFC2672] in the DNS Resource Record (RR) Types
registry table at http://www.iana.org/assignments/dns-parameters.
IANA has updated the DNS resource record registry to point to this
document for RR type 39.

8. 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.

9. Acknowledgments

The authors of this document 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 reviews and comments on this
document.

10. References

10.1. 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.

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

   [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.

10.2. 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.

   [RFC3492] Costello, A., "Punycode: A Bootstring encoding of Unicode
              for Internationalized Domain Names in Applications
              (IDNA)", RFC 3492, March 2003.

   [RFC6434] Jankiewicz, E., Loughney, J., and T. Narten, "IPv6 Node
              Requirements", RFC 6434, December 2011.

Appendix A. Changes from RFC 2672

A.1. Changes to Server Behavior

   Major changes to server behavior from the original DNAME
   specification are summarized below:

   o The rules for DNAME substitution have been clarified in
      Section 2.2.

   o The EDNS option to signal DNAME understanding and compression has
      never been specified, and this document clarifies that there is no
      signaling method (Section 2.5).

   o The TTL for synthesized CNAME RRs is now set to the TTL of the
      DNAME, not zero (Section 3.1).

   o Recursive caching servers MUST perform CNAME synthesis on behalf
      of clients (Section 3.4).

   o The revised server algorithm is detailed in Section 3.2.

   o Rules for dynamic update messages adding a DNAME or CNAME RR to a
      zone where a CNAME or DNAME already exists are detailed in
      Section 5.2.

A.2. Changes to Client Behavior

   Major changes to client behavior from the original DNAME
   specification are summarized below:

   o Clients MUST be able to accept synthesized CNAME RR's with a TTL
      of either zero or the TTL of the DNAME RR that accompanies the
      CNAME RR.

   o DNSSEC-aware clients SHOULD cache DNAME RRs and MAY cache
      synthesized CNAME RRs they receive in the same response. DNSSEC-
      aware clients SHOULD also check the NSEC/NSEC3 type bitmap to
      verify that DNAME redirection is to be done. DNSSEC validators
      MUST understand DNAME (Section 5.3).

   o The revised client algorithm is detailed in Section 3.4.1.

Authors' Addresses

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

   Phone: +1-301-975-8439
   Fax: +1-301-975-6238
   EMail: scott.rose@nist.gov

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

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