DNSOP Working Group   L. Peltan
Internet-Draft   CZ.NIC
Intended status: Standards Track   O. Sury
Expires: September 10, 2020   Internet Systems Consortium
                                                            W. Toorop
                                                           NLnet Labs
                                                    L. Vandewoestijne
                                                        March 9, 2020

DNS Catalog Zones
draft-toorop-dnsop-dns-catalog-zones-00

Abstract

   This document describes a method for automatic DNS zone provisioning
   among DNS primary and secondary nameservers by storing and
   transferring the catalog of zones to be provisioned as one or more
   regular DNS zones.

Status of This Memo

Copyright Notice

Table of Contents

[1](#). Introduction

This is a simple resubmit of "[draft-muks-dnsop-dns-catalog-zones-04](#)"
with the authors replaced. The new authors are representatives of
Open Source DNS implementations and a DNS Operator who want to work
and cooperate on progressing this draft to produce a cross-
implementation DNS Catalog Zones standard.

The data in a DNS zone is synchronized amongst its primary and
secondary nameservers using AXFR and IXFR. However, the list of
zones served by the primary (called a catalog in [[RFC1035](#)]) is not
automatically synchronized with the secondaries. To add or remove a
zone, the administrator of a DNS nameserver farm not only has to add
or remove the zone from the primary, they must also add/remove the
zone from all secondaries, either manually or via an external
application. This can be both inconvenient and error-prone; it will
also be dependent on the nameserver implementation.

This document describes a method in which the catalog is represented
as a regular DNS zone (called a "catalog zone" here), and transferred
using DNS zone transfers. As zones are added to or removed from the
catalog zone, the changes are propagated to the secondary nameservers
in the normal way. The secondary nameservers then add/remove/modify
the zones they serve in accordance with the changes to the zone.

The contents and representation of catalog zones are described in
[Section 3](#). Nameserver behavior is described in [Section 6](#).

[2](#). Terminology

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
[BCP 14](#) [[RFC2119](#)] [[RFC8174](#)] when, and only when, they appear in all
capitals, as shown here.

Catalog zone
   A DNS zone containing a DNS catalog, that is, a list of DNS zones
   and associated zone configuration.

Member zone
   A DNS zone whose configuration is published inside a catalog zone.

Zone property
   A configuration parameter of a zone, sometimes also called a zone
   option, represented as a key/value pair.

$CATZ

Used in examples as a placeholder to represent the domain name of
the catalog zone itself (c.f. $ORIGIN).

3. Description

A catalog zone is a specially crafted DNS zone that contains, as DNS
zone data:

o A list of DNS zones (called "member zones").

o Default zone configuration information common to all member zones.

o Zone-specific configuration information.

An implementation of catalog zones MAY allow the catalog to contain
other catalog zones as member zones, but default zone configuration
present in a catalog zone only applies to its immediate member zones.

Although the contents of a catalog zone are interpreted and acted
upon by nameservers, a catalog zone is a regular DNS zone and so must
adhere to the standards for such zones.

A catalog zone is primarily intended for the management of a farm of
authoritative nameservers. It is not expected that the content of
catalog zones will be accessible from any recursive nameserver.

4. Catalog Zone Structure

4.1. SOA and NS Records

As with any other DNS zone, a catalog zone MUST have a syntactically
correct SOA record and one or more NS records at its apex.

The SOA record's SERIAL, REFRESH, RETRY and EXPIRE fields [RFC1035]
are used during zone transfer. A catalog zone's SOA SERIAL field
MUST increase when an update is made to the catalog zone's contents
as per serial number arithmetic defined in [RFC1982]. Otherwise,
secondary nameservers might not notice updates to the catalog zone's
contents.

Should the zone be made available for querying, the SOA record's
MINIMUM field's value is the negative cache time (as defined in
[RFC2308]). Since recursive nameservers are not expected to be able
to access (and subsequently cache) entries from a catalog zone a
value of zero (0) is RECOMMENDED.

Since there is no requirement to be able to query the catalog zone
via recursive namservers the NS records at the apex will not be used

and no parent delegation is required. However, they are still
required so that catalog zones are syntactically correct DNS zones.
Any valid DNS name can be used in the NSDNAME field of such NS
records [RFC1035] and they MUST be ignored. A single NS RR with an
NSDNAME field containing the absolute name "invalid." is RECOMMENDED
[RFC2606].

4.2. Zone Data

A catalog zone contains a set of key/value pairs, where each key is
encapsulated within the owner name of a DNS RR and the corresponding
value is stored in the RR's RDATA. The specific owner name depends
on whether the property relates to the catalog zone itself, a member
zone thereof, or to default zone properties described in Section 4.3.
The owner names are case insensitive.

4.2.1. Resource Record Format

Each key/value pair has a defined data type, and each data type
accordingly uses a particular RR TYPE to represent its possible
values, as specified in Section 5.

The general form of a catalog zone record is as follows:

        [<unique-id>.]<key>.<path>.$CATZ 0 IN <RRTYPE> <value>

where "<path>" is a sequence of labels with values depending on the
purpose (and hence position) of the record within the catalog zone
(see Section 4.3) and where the "<unique-id>" prefix is only present
for multi-valued properties (see Section 4.2.2).

NB: Catalog zones use some RR TYPEs (such as PTR) with alternate
semantics to those originally defined for them. Although this may be
controversial, the situation is similar to other similar zone-based
representations such as response-policy zones [RPZ].

The CLASS field of every RR in a catalog zone MUST be IN (1). This
is because some RR TYPEs such as APL used by catalog zones are
defined only for the IN class.

The TTL field's value is not specially defined by this memo. Catalog
zones are for authoritative nameserver management only and are not
intended for general querying via recursive resolvers and therefore a
value of zero (0) is RECOMMENDED.

It is an error for any single owner name within a catalog zone (other
than the apex of the zone itself) to have more than one RR associated
with it.

4.2.2. Multi-valued Properties

Some properties do not represent single values but instead represent
a collection of values. The specification for each property
describes whether it is single-valued or multi-valued. A multi-
valued property is encoded as multiple RRs where the owner name of
each individual RR contains a unique (user specified) DNS label.

So, while a single-valued key might be represented like this:

```
                  <key>.<path>.$CATZ IN TXT "value"
```

a multi-valued key would be represented like this:

```
        <unique-id-1>.<key>.<path>.$CATZ IN TXT "value 1"
        <unique-id-2>.<key>.<path>.$CATZ IN TXT "value 2"
        ...
```

NB: a property that is specified to be multi-valued MUST be encoded
using the unique prefixed key syntax even if there is only one value
present.

The specification of any multi-valued property MUST document whether
the collection represents either an ordered or unordered list. In
the former case the ordering of the prefixes according to the usual
DNS canonical name ordering will determine the sort order.

4.2.3. Vendor-specific Properties

TBD: Prepare a list of zone configuration properties that are common
to DNS implementations. This is so that a company may manage a
catalog zone using a Windows DNS server as the primary, and a
secondary nameserver hosting service may pick up the common
properties and may use a different nameserver implementation such as
BIND or NSD on a POSIX operating system to serve it.

TBD: We may specify that unrecognized zone property names must be
ignored, or that nameserver specific properties must be specified
using the "x-" prefix similar to MIME type naming.

TBD: Any list of zone properties is ideally maintained as a registry
rather than within this memo.

4.3. Zone Structure

4.3.1. List of Member Zones

   The list of member zones is specified as a multi-valued collection of
   domain names under the owner name "zones" where "zones" is a direct
   child domain of the catalog zone.

   The names of member zones are represented on the RDATA side (instead
   of as a part of owner names) so that all valid domain names may be
   represented regardless of their length [RFC1035].

   For example, if a catalog zone lists three zones "example.com.",
   "example.net." and "example.org.", the RRs would appear as follows:

                <m-unique-1>.zones.$CATZ 0 IN PTR example.com.
                <m-unique-2>.zones.$CATZ 0 IN PTR example.net.
                <m-unique-3>.zones.$CATZ 0 IN PTR example.org.

   where "<m-unique-N>" is a label that uniquely tags each record in the
   collection, as described in Section 4.2.2.

   Although any legal label could be used for "<m-unique-N>" it is
   RECOMMENDED that it be a value deterministically derived from the
   fully-qualified member zone name. The BIND9 implementation uses the
   40 character hexadecimal representation of the SHA-1 digest
   [FIPS.180-4.2015] of the lower-cased member zone name as encoded in
   uncompressed wire format.

4.3.2. Catalog Zone Schema Version

   The catalog zone schema version is specified by an unsigned integer
   property with the property name "version". All catalog zones MUST
   have this property present. Primary and secondary nameservers MUST
   NOT use catalog zones with an unexpected value in this property, but
   they may be transferred as ordinary zones. For this memo, the
   "version" property value MUST be set to 2, i.e.

                      version.$CATZ 0 IN TXT "2"

   NB: Version 1 was used in a draft version of this memo and reflected
   the implementation first found in BIND 9.11.

4.3.3. Default Zone Configuration

   Default zone configuration comprises a set of properties that are
   applied to all member zones listed in the catalog zone unless
   overridden my member zone-specific information.

   All such properties are stored as child nodes of the owner name
   "defaults" itself a direct child node of the catalog zone, e.g.:

           example-prop.defaults.$CATZ 0 IN TXT "Example"

4.3.4. Zone Properties Specific to a Member Zone

   Default zone properties can be overridden on a per-zone basis by
   specifying the property under the the sub-domain associated with the
   member zone in the list of zones, e.g.:

         example-prop.<m-unique>.zones.$CATZ 0 IN TXT "Example"

   where "m-unique" is the label that uniquely identifies the member
   zone name as described in Section 4.3.1.

   NB: when a zone-specific property is multi-valued the owner name will
   contain two unique identifiers, the left-most tagging being
   associated with the individual value ("<unique-id-N>") and the other
   ("<m-unique>") associated with the member zone itself, e.g.:

              $ORIGIN <m-unique>.zones.$CATZ
              <unique-id-1>.example-prop 0 IN TXT "Value 1"
              <unique-id-2>.example-prop 0 IN TXT "Value 2"
              ...

5. Data Types

   This section lists the various data types defined for use within
   catalog zones.

5.1. String

   A key with a string value is represented with a TXT RR [RFC1035],
   e.g.:

         example-prop.<m-unique>.zones.$CATZ 0 IN TXT "Example"

   If the RDATA is split into multiple "<character-string>" elements the
   MUST be directly concatenated without any separating character.

5.2. Booleans

   A key with a boolean value is represented with a TXT RR containing a
   single "<character-string>" with a value of "true" for true condition
   and "false" for false condition, e.g:

         example-prop.<m-unique>.zones.$CATZ 0 IN TXT "false"

The RDATA is case-insensitive.

5.3. Integers

A key with an integer value is specified using a TXT RR containing a
single "<character-string>".

A signed integer's TXT RDATA uses the representation of an unsuffixed
"integer constant" as defined in the C programming language standard
[ISO.9899.1990] (of the type matching a 64-bit signed integer on that
platform), with an optional minus prefix.

An unsigned integer's TXT RDATA uses the representation of an
unsuffixed "integer constant" as defined in the C programming
language standard [ISO.9899.1990] (of the type matching a 64-bit
unsigned integer on that platform).

For example, a property with an unsigned integer value of 300 would
appear as follows:

        example-prop.<m-unique>.zones.$CATZ 0 IN TXT "300"

5.4. Floating-Point Values

A key with a floating-point value is specified using a TXT RR
containing a single "<character-string>".

A floating-point value's TXT RDATA uses the representation of an
unsuffixed "floating constant" as defined in the C programming
language standard [ISO.9899.1990].

For example, a property with an unsigned integer value of 0.15 may
appear as follows:

        example-prop.<m-unique>.zones.$CATZ 0 IN TXT "15e-2"

5.5. Domain Name

A key whose value is a domain name is specified using a PTR RR
[RFC1035], e.g.:

        example-prop.defaults.$CATZ 0 IN PTR ns1.example.com.

5.6. IP Prefix

A property whose value is an IP network prefix is specified using an
APL RR [RFC3123]. The negation flag ("!" in presentation format) may

be used to indicate all addresses not included within that prefix,
e.g. for use in Access Control Lists, e.g.:

Although a single APL record is capable of containing multiple
prefixes, for consistency of representation lists of prefixes MUST
use the multi-valued property syntax as documented in [Section 4.2.2](#),
e.g.:

```
    $ORIGIN <m-unique>.zones.$CATZ
    <unique-id-1>.example-prop 0 IN APL ( 1:192.0.2.0/24 )
    <unique-id-2>.example-prop 0 IN APL ( !1:0.0.0.0/0 )
```

Implementations MUST accept only the first prefix within each APL
record and MUST ignore any subsequent prefixes found therein.

5.7. Single Host Address

A single host address is represented using either an A or AAAA record
as appropriate, e.g.:

```
    example-prop1.<m-unique>.zones.$CATZ 0 IN A 192.0.2.1
    example-prop2.<m-unique>.zones.$CATZ 0 IN AAAA 2001:db8::1
```

6. Nameserver Behavior

6.1. General Requirements

As it is a regular DNS zone, a catalog zone can be transferred using
DNS zone transfers among nameservers.

Although they are regular DNS zones, catalog zones contain only
information for the management of a set of authoritative nameservers.
For this reason, operators may want to limit the systems able to
query these zones. It may be inconvenient to serve some contents of
catalog zones via DNS queries anyway due to the nature of their
representation. A separate method of querying entries inside the
catalog zone may be made available by nameserver implementations (see
[Section 7.1](#)).

Catalog updates should be automatic, i.e., when a nameserver that
supports catalog zones completes a zone transfer for a catalog zone,
it SHOULD apply changes to the catalog within the running nameserver
automatically without any manual intervention.

As with regular zones, primary and secondary nameservers for a
catalog zone may be operated by different administrators. The
secondary nameservers may be configured to synchronize catalog zones

from the primary, but the primary's administrators may not have any
administrative access to the secondaries.

A catalog zone can be updated via DNS UPDATE on a reference primary
nameserver, or via zone transfers. Nameservers MAY allow loading and
transfer of broken zones with incorrect catalog zone syntax (as they
are treated as regular zones), but nameservers MUST NOT process such
broken zones as catalog zones. For the purpose of catalog
processing, the broken catalogs MUST be ignored. If a broken catalog
zone was transferred, the newly transferred catalog zone MUST be
ignored (but the older copy of the catalog zone SHOULD be left
running subject to values in SOA fields).

If there is a clash between an existing member zone's name and an
incoming member zone's name (via transfer or update), the new
instance of the zone MUST be ignored and an error SHOULD be logged.

When zones are introduced into a catalog zone, a primary SHOULD first
make the new zones available for transfers before making the updated
catalog zone available for transfer, or sending NOTIFY for the
catalog zone to secondaries. Note that secondary nameservers may
attempt to transfer the catalog zone upon refresh timeout, so care
must be taken to make the member zones available before any update to
the list of member zones is visible in the catalog zone.

When zones are deleted from a catalog zone, a primary MAY delete the
member zone immediately after notifying secondaries. It is up to the
secondary nameserver to handle this condition correctly.

TBD: Transitive primary-secondary relationships

7. Updating Catalog Zones

TBD: Explain updating catalog zones using DNS UPDATE.

7.1. Implementation Notes

Catalog zones on secondary nameservers would have to be setup
manually, perhaps as static configuration, similar to how ordinary
DNS zones are configured. Members of such catalog zones will be
automatically synchronized by the secondary after the catalog zone is
configured.

An administrator may want to look at data inside a catalog zone.
Typical queries might include dumping the list of member zones,
dumping a member zone's effective configuration, querying a specific
property value of a member zone, etc. Because of the structure of
catalog zones, it may not be possible to perform these queries

intuitively, or in some cases, at all, using DNS QUERY. For example
it is not possible to enumerate the contents of a multi-valued
property (such as the list of member zones) with a single QUERY.
Implementations are therefore advised to provide a tool that uses
either the output of AXFR or an out-of-band method to perform queries
on catalog zones.

8. Security Considerations

As catalog zones are transmitted using DNS zone transfers, it is
absolutely essential for these transfers to be protected from
unexpected modifications on the route. So, catalog zone transfers
SHOULD be authenticated using TSIG [RFC2845]. A primary nameserver
SHOULD NOT serve a catalog zone for transfer without using TSIG and a
secondary nameserver SHOULD abandon an update to a catalog zone that
was received without using TSIG.

Use of DNS UPDATE [RFC2136] to modify the content of catalog zones
SHOULD similarly be authenticated using TSIG.

Zone transfers of member zones SHOULD similarly be authenticated
using TSIG [RFC2845]. The TSIG shared secrets used for member zones
MUST NOT be mentioned anywhere in the catalog zone data. However,
key identifiers may be shared within catalog zones.

Catalog zones do not need to be signed using DNSSEC, their zone
transfers being authenticated by TSIG. Signed zones MUST be handled
normally by nameservers, and their contents MUST NOT be DNSSEC-
validated.

9. IANA Considerations

This document has no IANA actions.

10. Acknowledgements

Our deepest thanks and appreciation go to Mukund Sivaraman, Stephen
Morris, Ray Bellis and Witold Krecicki who initiated this draft and
did the bulk of the work.

Catalog zones originated as the chosen method among various proposals
that were evaluated at ISC for easy zone management. The chosen
method of storing the catalog as a regular DNS zone was proposed by
Stephen Morris.

The initial authors discovered that Paul Vixie's earlier [Metazones]
proposal implemented a similar approach and reviewed it. Catalog

zones borrows some syntax ideas from Metazones, as both share this
scheme of representing the catalog as a regular DNS zone.

Thanks to Brian Conry, Tony Finch, Evan Hunt, Patrik Lundin, Victoria
Risk and Carsten Strettman for reviewing draft proposals and offering
comments and suggestions.

11. References

11.1. Normative References

[FIPS.180-4.2015]
          National Institute of Standards and Technology, "Secure
          Hash Standard", FIPS PUB 180-4, August 2015,
          <http://csrc.nist.gov/publications/fips/fips180-4/fips-
          180-4.pdf>.

[ISO.9899.1990]
          International Organization for Standardization,
          "Programming languages - C", ISO Standard 9899, 1990.

[RFC1035] Mockapetris, P., "Domain names - implementation and
          specification", STD 13, RFC 1035, DOI 10.17487/RFC1035,
          November 1987, <https://www.rfc-editor.org/info/rfc1035>.

[RFC1982] Elz, R. and R. Bush, "Serial Number Arithmetic", RFC 1982,
          DOI 10.17487/RFC1982, August 1996,
          <https://www.rfc-editor.org/info/rfc1982>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate
          Requirement Levels", BCP 14, RFC 2119,
          DOI 10.17487/RFC2119, March 1997,
          <https://www.rfc-editor.org/info/rfc2119>.

[RFC2136] Vixie, P., Ed., Thomson, S., Rekhter, Y., and J. Bound,
          "Dynamic Updates in the Domain Name System (DNS UPDATE)",
          RFC 2136, DOI 10.17487/RFC2136, April 1997,
          <https://www.rfc-editor.org/info/rfc2136>.

[RFC2308] Andrews, M., "Negative Caching of DNS Queries (DNS
          NCACHE)", RFC 2308, DOI 10.17487/RFC2308, March 1998,
          <https://www.rfc-editor.org/info/rfc2308>.

[RFC2606] Eastlake 3rd, D. and A. Panitz, "Reserved Top Level DNS
          Names", BCP 32, RFC 2606, DOI 10.17487/RFC2606, June 1999,
          <https://www.rfc-editor.org/info/rfc2606>.

   [RFC2845] Vixie, P., Gudmundsson, O., Eastlake 3rd, D., and B.
              Wellington, "Secret Key Transaction Authentication for DNS
              (TSIG)", RFC 2845, DOI 10.17487/RFC2845, May 2000,
              <https://www.rfc-editor.org/info/rfc2845>.

   [RFC3123] Koch, P., "A DNS RR Type for Lists of Address Prefixes
              (APL RR)", RFC 3123, DOI 10.17487/RFC3123, June 2001,
              <https://www.rfc-editor.org/info/rfc3123>.

   [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC
              2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174,
              May 2017, <https://www.rfc-editor.org/info/rfc8174>.

11.2. Informative References

   [Metazones]
              Vixie, P., "Federated Domain Name Service Using DNS
              Metazones", 2005, <http://ss.vix.su/~vixie/mz.pdf>.

   [RPZ]   Vixie, P. and V. Schryver, "DNS Response Policy Zones (DNS
              RPZ)", 2010,
              <http://ftp.isc.org/isc/dnsrpz/isc-tn-2010-1.txt>.

Appendix A. Open issues and discussion (to be removed before final publication)

Config options
   We want catalog zones to be adopted by multiple DNS
   implementations. Towards this, we have to generalize zone config
   options and adopt a minimal set that we can expect most
   implementations to support.

Catalog zone and member zones on different primary nameservers
   Will it be possible to setup a catalog zone on one nameserver as
   primary, and allow its member zones to be served by different
   primary nameservers?

Transitive relationships
   For a catalog zone, a secondary nameserver may be a primary
   nameserver to a different set of nameservers in a nameserver farm.
   In these transitive relationships, zone configuration options
   (such as also-notify and allow-transfer) may differ based on the
   location of the primary in the hierarchy. It may not be possible
   to specify this within a catalog zone.

Overriding controls
   A way to override zone config options (as prescribed by the
   catalog zones) on secondary nameservers was requested. As this

would be configured outside catalog zones, it may be better to
leave this to implementations.

Appendix B. Change History (to be removed before final publication)

o draft-muks-dnsop-dns-catalog-zones-00

   Initial public draft.

o draft-muks-dnsop-dns-catalog-zones-01

   Added Witold, Ray as authors. Fixed typos, consistency issues.
   Fixed references. Updated Area. Removed newly introduced custom
   RR TYPEs. Changed schema version to 1. Changed TSIG requirement
   from MUST to SHOULD. Removed restrictive language about use of
   DNS QUERY. When zones are introduced into a catalog zone, a
   primary SHOULD first make the new zones available for transfers
   first (instead of MUST). Updated examples, esp. use IPv6 in
   examples per Fred Baker. Add catalog zone example.

o draft-muks-dnsop-dns-catalog-zones-02

   Addressed some review comments by Patrik Lundin.

o draft-muks-dnsop-dns-catalog-zones-03

   Revision bump.

o draft-muks-dnsop-dns-catalog-zones-04

   Reordering of sections into more logical order. Separation of
   multi-valued properties into their own category.

o draft-toorop-dnsop-dns-catalog-zones-00

   New authors to pickup the editor pen on this draft

Authors' Addresses

   Libor Peltan
   CZ.NIC
   CZ

   Email: libor.peltan@nic.cz

Ondrej Sury
Internet Systems Consortium
CZ

Email: ondrej@isc.org

Willem Toorop
NLnet Labs
Science Park 400
Amsterdam 1098 XH
Netherlands

Email: willem@nlnetlabs.nl

Leo Vandewoestijne
Netherlands

Email: leo@dns.company