Workgroup: DNSOP Working Group
Internet-Draft:
draft-ietf-dnsop-dns-catalog-zones-03
Published: 25 August 2021
Intended Status: Standards Track
Expires: 26 February 2022
Authors: P. van Dijk L. Peltan
PowerDNS CZ.NIC
O. Sury W. Toorop
Internet Systems Consortium NLnet Labs
L. Vandewoestijne

DNS Catalog Zones

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

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 https://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 26 February 2022.

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
2. Terminology
3. Description
4. Catalog Zone Structure
   4.1. SOA and NS Records
   4.2. Catalog Zone Schema Version
   4.3. List of Member Zones
5. Properties
   5.1. The Change of ownership (Coo) Property
   5.2. The Group Property
      5.2.1. Group Property Example
   5.3. The Epoch Property
      5.3.1. The TIMESTAMP Resource Record
   5.4. The Serial Property
      5.4.1. The SERIAL Resource Record
      5.4.2. SERIAL RDATA Wire Format
      5.4.3. SERIAL Presentation Format
      5.4.4. SERIAL RR Usage
   5.5. Custom properties
6. Nameserver Behavior
   6.1. General Requirements
   6.2. Member zone removal
   6.3. Member zone name clash
   6.4. Migrating member zones between catalogs
   6.5. Zone associated state reset
7. Implementation Notes
8. Implementation Status
9. Security Considerations
10. IANA Considerations
   10.1. TIMESTAMP RR type
   10.2. SERIAL RR type
11. Acknowledgements
12. Normative References
13. Informative References

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

Authors' Addresses

1. Introduction

The content of 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 is also 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, these changes are distributed 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 catalog 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 properties.

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

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

Catalog producer An entity that generates and is responsible for the contents of the catalog zone.

Catalog consumer An entity that extracts information from the catalog zone (such as a DNS server that configures itself according to the catalog zone's contents).

Member node The DNS name of the DNS subtree representing a given member zone (two levels below $CATZ).

3. Description

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

* A list of DNS zones (called "member zones"), plus properties associated with those zones.

Implementations of catalog zones SHOULD ignore any RR in the catalog zone which is meaningless or useless to the implementation.

Authoritative servers may be preconfigured with multiple catalog zones, each associated with a different set of configurations. A member zone can as such be reconfigured with a different set of preconfigured settings by removing it as a member of one catalog zone and making it a member of another.

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 at least one NS record 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.

There is no requirement to be able to query the catalog zone via recursive nameservers. Implementations of catalog zones MUST ignore and MUST NOT assume or require NS records at the apex. However, at least one is still required so that catalog zones are syntactically correct DNS zones. A single NS RR with a NSDNAME field containing the absolute name "invalid." is RECOMMENDED [RFC2606].

4.2. Catalog Zone Schema Version

The catalog zone schema version is specified by an integer value embedded in a TXT RR named version.$CATZ. All catalog zones MUST have a TXT RRset named version.$CATZ with at least one RR. Primary and secondary nameservers MUST NOT apply catalog zone processing to zones without the expected value in one of the RRs in the version. $CATZ TXT RRset, but they may be transferred as ordinary zones. For this memo, the value of one of the RRs in the version.CATZ TXT RRset 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. List of Member Zones

The list of member zones is specified as a collection of member nodes, represented by 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) of a PTR record, so that all valid domain names may be represented regardless of their length [RFC1035]. This PTR record MUST be the only record in the PTR RRset with the same name.

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

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

where <unique-N> is a label that tags each record in the collection. <unique-N> has an unique value in the collection.

Member node labels carry no informational meaning beyond labeling member zones. A changed label may indicate that the state for a zone needs to be reset (see Section 6.5).

Having the zones uniquely tagged with the <unique-N> label ensures that additional RRs can be added below the member node (see Section 5). Further, if member zones do not share a PTR RRset, the list of member zones can be split over multiple DNS messages in a zone transfer.

A catalog zone consumer MUST ignore PTR RRsets with more than a single record.

The CLASS field of every RR in a catalog zone MUST be IN (1).

The TTL field's value is not defined by this memo. Catalog zones are for authoritative nameserver management only and are not intended for general querying via recursive resolvers.

5. Properties

Each member zone MAY have one or more additional properties, described in this chapter. These properties are completely optional and the catalog zone consumer SHOULD ignore those it does not understand. Properties are represented by RRsets below the corresponding member node.

5.1. The Change of ownership (Coo) Property

The 'coo' property facilitates controlled migration of a member zone from one catalog to another.

A Change Of Ownership is signaled by the 'coo' property in the catalog zone currently `owning'' the zone. The name of the new catalog is in the value of a PTR record in the old catalog. For example if member "example.com." will migrate from catalog zoneOLDCATZ` to catalog zone `NEWCATZ, this appears in the$OLDCATZ` catalog zone as follows:

```
<unique-N>.zones.$OLDCATZ 0 IN PTR example.com.
coo.<unique-N>.zones.$OLDCATZ 0 IN PTR zones.$NEWCATZ
```

The PTR RRset MUST consist of a single PTR record. A catalog zone consumer MUST ignore PTR RRsets with more than a single record.

When a catalog zone consumer of catalog zone $OLDCATZ receives an update which adds or changes a coo property for a member zone in $OLDCATZ signalling a new owner $NEWCATZ, it does *not* migrate the member zone immediately.

This is because the catalog zone consumer may not have the <unique-N> identifier associated with the member zone in $NEWCATZ and because name servers do not index Resource Records by RDATA, it may not know wether or not the member zone is configured in $NEWCATZ at all. It may have to wait for an update of $NEWCATZ adding or changing that member zone.

When a catalog zone consumer of catalog zone $NEWCATZ receives an update of $NEWCATZ which adds or changes a member zone, *and* that consumer had the member zone associated with $OLDCATZ, *and* there is a coo property of the member zone in $OLDCATZ pointing to $NEWCATS, *only then* it will reconfigure the member zone with the for $NEWCATZ preconfigured settings.

All associated state for the zone (such as the zone data, or DNSSEC keys) is in such case reset, unless the epoch property (see Section 5.3) is supported by the catalog zone consumer and the member zone in both $OLDCATZ and $NEWCATZ have an epoch property with the same value.

The new owner is advised to increase the serial of the member zone after the ownership change, so that the old owner can detect that the transition is done. The old owner then removes the member zone from old.catalog.

5.2. The Group Property

With a group property, consumer(s) can be signalled to treat some member zones within the catalog zone differently.

The consumer MAY apply different configuration options when processing member zones, based on the value of the group property. The exact handling of configuration referred to by the group property value is left to the consumer's implementation and configuration. The property is defined by a TXT record in the sub-node labelled group.

The producer MAY assign a group property to all, some, or none of the member zones within a catalog zone. The producer MUST NOT assign more than one group property to one member zone.

The consumer MUST ignore either all or none of the group properties in a catalog zone.

The value of the TXT record MUST be at most 255 octets long and MUST NOT contain whitespace characters. The consumer MUST interpret the value case-sensitively.

5.2.1. Group Property Example

```
<unique-1>.zones.$CATZ        0 IN PTR    example.com.
group.<unique-1>.zones.$CATZ  0 IN TXT    sign-with-nsec3
<unique-2>.zones.$CATZ        0 IN PTR    example.net.
group.<unique-2>.zones.$CATZ  0 IN TXT    nodnssec
```

In this case, the consumer might be implemented and configured in the way that the member zones with "nodnssec" group assigned will not be signed with DNSSEC, and the zones with "sign-with-nsec3" group assigned will be signed with DNSSEC with NSEC3 chain.

By generating the catalog zone (snippet) above, the producer signals how the consumer shall treat DNSSEC for the zones example.net. and example.com., respectively.

5.3. The Epoch Property

A epoch property allows a producer to trigger, on the consumer, a reset of all state associated with a zone.

The epoch property is represented by a the TIMESTAMP Resource Record (see Section 5.3.1).

*epoch.<unique-N>.zones.$CATZ 0 IN TIMESTAMP ...

When a member zone's epoch changes, the secondary server resets the member zone's state. The secondary can detect a member zone epoch change as follows:

*When the epoch changes, the primary will set the TIMESTAMP RR of the member zone's epoch property to the current time.

*When the secondary processes a member node with an epoch property that is larger than the point in time when the member zone itself was last retrieved, then a new epoch has begun.

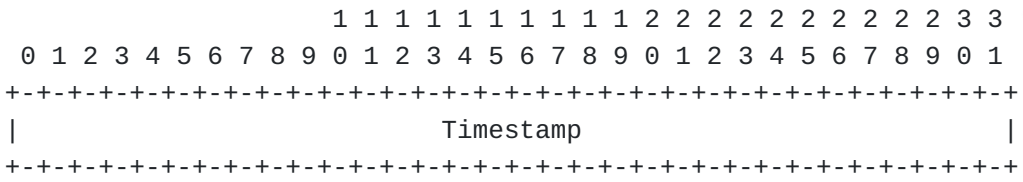
The steps entailed in the process of resetting the zone state depend on the operational context of the secondary (e.g. regenerate DNSSEC keys).

5.3.1. The TIMESTAMP Resource Record

Epoch values (both for the catalog zone and for member zones) are provided with a TIMESTAMP Resource Record. The Type value for the TIMESTAMP RR is TBD. The TIMESTAMP RR is class independent. The RDATA of the resource record consists of a single field: Timestamp.

5.3.1.1. TIMESTAMP RDATA Wire Format

The TIMESTAMP RDATA wire format is encoded as follows:



The wire format is identical to the wire format of the Signature Expiration and Inception Fields of the RRSIG RR ([RFC4034] section 3.1.5) and follows the same rules with respect to wrapping.

5.3.1.2. TIMESTAMP RDATA Presentation Format

The presentation format is identical to that of the Signature Expiration and Inception Fields of the RRSIG RR ([RFC4034] section 3.2). Example:

```
epoch.$CATZ                    0 IN TIMESTAMP    20210304124652
epoch.<unique-1>.zones.$CATZ   0 IN TIMESTAMP    20201231235959
```

5.4. The Serial Property

The serial property helps in increasing reliability of zone update signaling and may help in reducing NOTIFY and SOA query traffic.

The current default mechanism for prompting notifications of zone changes from a primary nameserver to the secondaries via DNS NOTIFY [RFC1996], can be unreliable due to packet loss, or secondary nameservers temporarily not being reachable. In such cases the secondary might pick up the change only after the refresh timer runs out, which might take long time and be out of the control of the primary nameserver operator. Low refresh values in the zones being served can alleviate update delays, but burden both the primary and secondary nameservers with more refresh queries, especially with larger numbers of secondary nameservers serving large numbers of zones. To mitigate this, updates of zones MAY be signalled via catalog zones with the help of a serial property.

The serial number in the SOA record of the most recent version of a member zone MAY be provided by a serial property. When a serial property is present for a member zone, implementations of catalog zones MAY assume this number to be the current serial number in the SOA record of the most recent version of the member zone.

Nameservers that are secondary for that member zone, MAY compare the serial property with the SOA serial since the last time the zone was fetched. When the serial property is larger, the secondary MAY initiate a zone transfer immediately without doing a SOA query first. The SOA query may be omitted, because the SOA serial has been obtained reliably via the catalog zone already.

When a serial property is present for a member zone and it matches the SOA serial of that member zone, implementations of catalog zones which are secondary for that member zone MAY ignore the refresh time in the SOA record of the member zone and rely on updates via the serial property of the member zone. A refresh timer of a catalog zone MUST not be ignored.

Primary nameservers MAY be configured to omit sending DNS NOTIFY messages to secondary nameservers which are known to process the serial property of the member zones in that catalog. However they MAY also combine signalling of zone changes with the serial property of a member zone, as well as sending DNS NOTIFY messages, to anticipate slow updates of the catalog zone (due to packet loss or other reasons) and to cater for secondaries that do not process the serial property.

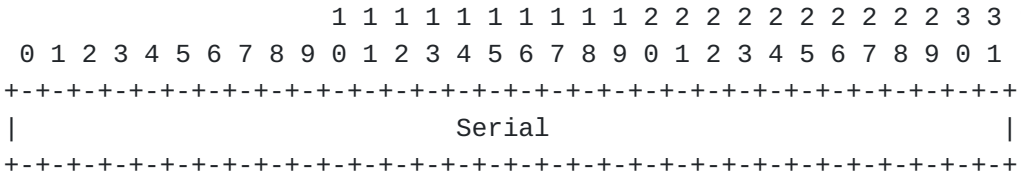
All comparisons of serial numbers MUST use "Serial number arithmetic", as defined in [RFC1982]

5.4.1. The SERIAL Resource Record

The serial property value is provided with a SERIAL Resource Record. The Type value for the SERIAL RR is TBD. The SERIAL RR is class independent. The RDATA of the resource record consist of a single field: Serial.

5.4.2. SERIAL RDATA Wire Format

The SERIAL RDATA wire format is encoded as follows:



5.4.2.1. The Serial Field

The Serial field is a 32-bit unsigned integer in network byte order. It is the serial number of the member zone's SOA record ([RFC1035] section 3.3.13).

5.4.3. SERIAL Presentation Format

The presentation format of the RDATA portion is as follows:

The Serial fields is represented as an unsigned decimal integer.

5.4.4. SERIAL RR Usage

The serial property of a member zone is provided by a SERIAL RRset with a single SERIAL RR named serial.<unique-N>.zones.$CATZ.

For example, if a catalog zone lists three zones "example.com.", "example.net." and "example.org.", and a serial property is provided for each of them, the RRs would appear as follows:

```
<unique-1>.zones.$CATZ        0 IN PTR    example.com.
serial.<unique-1>.zones.$CATZ 0 IN SERIAL 2020111712
<unique-2>.zones.$CATZ        0 IN PTR    example.net.
serial.<unique-2>.zones.$CATZ 0 IN SERIAL 2020111709
<unique-3>.zones.$CATZ        0 IN PTR    example.org.
serial.<unique-3>.zones.$CATZ 0 IN SERIAL 2020112405
```

5.5. Custom properties

Implementations and operators of catalog zones may choose to provide their own properties below the label private-extension.<unique-N>.zones.$CATZ. private-extension is not a placeholder, so a custom

property would have the domain name <your-label>.private-extension.<unique-N>.zones.$CATZ

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).

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.

6.2. Member zone removal

When a member zone is removed from a specific catalog zone, an authoritative server MUST NOT remove the zone and associated state data if the zone was not configured from that specific catalog zone. Only when the zone was configured from a specific catalog zone, and the zone is removed as a member from that specific catalog zone, the zone and associated state (such as zone data and DNSSEC keys) MAY be removed.

6.3. Member zone name clash

If there is a clash between an existing zone's name (either from an existing member zone or otherwise configured zone) 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.

A clash between an existing member zone's name and an incoming member zone's name (via transfer or update), may be an attempt to migrate a zone to a different catalog.

6.4. Migrating member zones between catalogs

If all consumers of the catalog zones involved support the coo property, it is RECOMMENDED to perform migration of a member zone by following the procedure described in Section 5.1. Otherwise a migration of member zone from a catalog zone $OLDCATZ to a catalog zone $NEWCATZ has to be done by: first removing the member zone from $OLDCATZ; second adding the member zone to $NEWCATZ.

If in the process of a migration some consumers of the involved catalog zones did not catch the removal of the member zone from $OLDCATZ yet (because of a lost packet or down time or otherwise), but did already see the update of $NEWCATZ, they may consider the update adding the member zone in $NEWCATZ to be a name clash (see #nameclash) and as a consequence the member is not migrated to $NEWCATZ. This possibility needs to be anticipated with a member zone migration. Recovery from such a situation is out of the scope of this document. It may for example entail a manually forced retransfer of $NEWCATZ to consumers after they have been detected to have received and processed the removal of the member zone from $OLDCATZ.

6.5. Zone associated state reset

It may be desirable to reset state (such as zone data and DNSSEC keys) associated with a member zone. If all consumers of the catalog zone support the epoch property, it is RECOMMENDED to perform a zone state reset following the procedure described in Section 5.3. Otherwise a zone state reset has to be done by: first removing the member zone from the catalog; add the member zone to the catalog again after having made sure all catalog zone consumers did process the member zone removal.

If in the process of a zone state reset some consumers of the involved catalog zone did not catch the removal (because of a lost packet or down time or otherwise) they will not have the zone associated state reset. This possibility needs to be anticipated. Recovery from it is out of the scope of this document. It may for example entail manual removal of the zone associated state from the

catalog zone consumers that did not catch the removal and re-addition of the member.

7. 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. Implementation Status

Note to the RFC Editor: please remove this entire section before publication.

In the following implementation status descriptions, "DNS Catalog Zones" refers to DNS Catalog Zones as described in this document.

- Knot DNS has processing of DNS Catalog Zones since Knot DNS Version 3.0.0, which was released on September 9, 2020.

- Knot DNS has generation of DNS Catalog Zones on a development branch.

- PowerDNS has a proof of concept external program called PowerCATZ, that can process DNS Catalog Zones.

- Proof of concept python scripts that can be used for both generating and consuming DNS Catalog Zones with NSD have been developed during the hackathon at the IETF-109.

Interoperability between the above implementations has been tested during the hackathon at the IETF-109.

9. Security Considerations

As catalog zones are transmitted using DNS zone transfers, it is key for these transfers to be protected from unexpected modifications on the route. So, catalog zone transfers SHOULD be authenticated using TSIG [RFC8945]. 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 [RFC8945]. 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 reveal the zones served by the consumers of the catalog zone. It is RECOMMENDED to limit the systems able to query these zones. It is RECOMMENDED to transfer catalog zones confidentially [RFC9103].

Administrative control over what zones are served from the configured name servers shifts completely from the server operator (consumer) to the "owner" (producer) of the catalog zone content.

10. IANA Considerations

10.1. TIMESTAMP RR type

This document defines a new DNS RR type, TIMESTAMP, in the "Resource Record (RR) TYPEs" subregistry of the "Domain Name System (DNS) Parameters" registry:

TYPE	Value	Meaning	Reference
TIMESTAMP	TBD	Timestamp	[this document]

Table 1

10.2. SERIAL RR type

This document defines a new DNS RR type, SERIAL, in the "Resource Record (RR) TYPEs" subregistry of the "Domain Name System (DNS) Parameters" registry:

TYPE	Value	Meaning	Reference
SERIAL	TBD	Version number of the original copy of the zone	[this document]

Table 2

11. Acknowledgements

Our deepest thanks and appreciation go to 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, Carsten Strotmann, Peter Thomassen and Kees Monshouwer for reviewing draft proposals and offering comments and suggestions.

Thanks to Klaus Darilion who came up with the idea for the serial property during the hackathon at the IETF-109. Thanks also to Shane Kerr, Petr Spacek, Brian Dickson for further brainstorming and discussing the serial property and how it would work best with catalog zones.

12. Normative References

[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>.

[RFC1996] Vixie, P., "A Mechanism for Prompt Notification of Zone Changes (DNS NOTIFY)", RFC 1996, DOI 10.17487/RFC1996, August 1996, <https://www.rfc-editor.org/info/rfc1996>.

[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>.

[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>.

[RFC4034] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "Resource Records for the DNS Security Extensions", RFC 4034, DOI 10.17487/RFC4034, March 2005, <https://www.rfc-editor.org/info/rfc4034>.

[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>.

[RFC8945] Dupont, F., Morris, S., Vixie, P., Eastlake 3rd, D., Gudmundsson, O., and B. Wellington, "Secret Key Transaction Authentication for DNS (TSIG)", STD 93, RFC 8945, DOI 10.17487/RFC8945, November 2020, <https://www.rfc-editor.org/info/rfc8945>.

[RFC9103] Toorop, W., Dickinson, S., Sahib, S., Aras, P., and A. Mankin, "DNS Zone Transfer over TLS", RFC 9103, DOI 10.17487/RFC9103, August 2021, <https://www.rfc-editor.org/info/rfc9103>.

13. Informative References

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

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

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

> Initial public draft.

*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.

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

Addressed some review comments by Patrik Lundin.

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

Revision bump.

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

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

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

New authors to pickup the editor pen on this draft

Remove data type definitions for zone properties Removing configuration of member zones through zone properties altogether

Remove Open issues and discussion Appendix, which was about zone options (including primary/secondary relationships) only.

*draft-toorop-dnsop-dns-catalog-zones-01

Added a new section "The Serial Property", introducing a new mechanism which can help with disseminating zones from the primary to the secondary nameservers in a timely fashion more reliably.

Three different ways to provide a "serial" property with a member zone are offered to or the workgroup for discussion.

Added a new section "Implementation Status", listing production ready, upcoming and Proof of Concept implementations, and reporting on interoperability of the different implementations.

*draft-toorop-dnsop-dns-catalog-zones-02

Adding the coo property for zone migration in a controlled fashion

Adding the group property for reconfigure settings of member zones in an atomic update

Adding the epoch property to reset zone associated state in a controlled fashion

*draft-toorop-dnsop-dns-catalog-zones-03

Big cleanup!

Introducing the terms catalog zone consumer and catalog zone producer

> Reorganized topics to create a more coherent whole
>
> Properties all have consistent format now
>
> Try to assume the least possible from implementations w.r.t.:
>
> 1) Predictability of the <unique-N> IDs of member zones
>
> 2) Whether or not fallback catalog zones can be found for a member
>
> 3) Whether or not a catalog zone consumer can maintain state

Authors' Addresses

Peter van Dijk
PowerDNS
Den Haag
Netherlands

Email: peter.van.dijk@powerdns.com

Libor Peltan
CZ.NIC
Czechia

Email: libor.peltan@nic.cz

Ondrej Sury
Internet Systems Consortium
Czechia

Email: ondrej@isc.org

Willem Toorop
NLnet Labs
Science Park 400
1098 XH Amsterdam
Netherlands

Email: willem@nlnetlabs.nl

Leo Vandewoestijne
Netherlands

Email: leo@dns.company