Network Working Group
Internet-Draft
Updates: 9432 (if approved)
Intended status: Standards Track
Expires: 26 September 2025

K. Dyson
Nominet UK

25 March 2025

Initialisation of Zone Files on the DNS Primary Server
draft-dyson-primary-zonefile-initialisation-01

Abstract

This document describes an update to "DNS Catalog Zones" ([RFC9432]) that facilitates a method for the primary server of a DNS zone to create the underlying master file for member zone(s) using information contained within a catalog zone.

Discussion Venues

This note is to be removed before publishing as an RFC.

Source for this draft and an issue tracker can be found at https://github.com/karldyson/draft-dyson-primary-zonefile-initialisation.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

Once a DNS zone's master file exists on the primary server, there is a standard way to automate the distribution of that zone to secondary servers defined in "DNS Catalog Zones" ([RFC9432]). Further, there is a standard way to dynamically alter the contents of a zone defined in "Dynamic Updates in the Domain Name System (DNS UPDATE)" ([RFC2136]).

However there is no standards-defined method of initialising a new master file for the zone, ready for such operations.

Various DNS software products have proprietary mechanisms for achieving this, some requiring that the zone master file is somehow pre-populated on the primary servers' filesystem.

Operators of large scale DNS systems may want to be able to signal the creation of a new file for a new zone without wanting to be tied to a particular vendor's proprietary software. Further, they may want to avoid the need or overhead of engineering a bespoke solution with the ongoing need to support and maintain it.

Having dynamically provisioned a new zone on the primary server, the operator may then manage resource records in the zone via "DNS Dynamic Updates" ([RFC2136]). In this scenario, they may also want to distribute the zones to secondary servers via "DNS Catalog Zones" ([RFC9432]).

This document defines a vendor-independent mechanism of signalling to the primary server that a new file is to be created for the new zone, populated with basic minimal initial zone data, and then loaded into the server to be authoritatively served.

The scope of this document is confined to the initial provisioning and loading of the zone on the primary server, including the creation of it's initial zone file, configuration and state.

Broader provisioning of the base nameserver configuration is beyond the scope of this document.

2. Conventions and Definitions

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.

The following is in addition to the conventions and definitions as defined in "DNS Catalog Zones" ([RFC9432]).

The use of parenthesis in the examples is as described in "Domain Names - Implementation and Specification" ([RFC1035]) Section 5.1.

2.1. Primary Server

Within DNS servers, specifically when transferring zones to other servers, there is the concept of a primary server and a secondary server in each transfer relationship.

Each secondary server will be transferring the zone from a configured upstream primary server, which may, itself, be a secondary server to a further upstream primary server, and so on.

However, within this document, the term "primary server" is used specifically to mean the primary server at the root of the AXFR/IXFR dependency graph. This server is where the resource records that form the zone’s content are maintained in the master file. Thus, that server does not transfer the zone from another server.

2.2. Master File

The term "master file" is as per the description in "Domain Names - Implementation and Specification" ([RFC1035]) Section 5, noting that some software products offer data stores for the master file that are not an actual file on a filesystem, such as a database.

3. Catalog Zone Properties

This section specifies new Catalog Zone level properties, additional to those defined in "DNS Catalog Zones" ([RFC9432]).

If initialisation of the underlying master file for the member zone is not required or is disabled in an implementation’s configuration, then the various initialisation properties defined in this document MAY be absent, and in that context, their absence DOES NOT constitute a broken catalog zone.

However, if the initialisation of the underlying master file for the member zone is enabled, and the properties and parameters defined below constitute a broken configuration as defined in this document, then the catalog is broken, and MUST NOT be processed ("DNS Catalog Zones" ([RFC9432]) Section 5.1).

3.1. Schema Version (version property)

For this memo, the value of the version resource record is unchanged.

"DNS Catalog Zones" ([RFC9432]) Section 3 is clear that "Catalog consumers MUST ignore any RRs in the catalog zone for which no processing is specified or which are otherwise not supported by the implementation." and as such the addition of the records outlined in this document will be ignored by implementations that do not recognise them.

3.2. Zone File Initialisation (init property)

When suitable configuration is activated in the implementation, and a new member zone entry is added to the catalog, the primary server MUST create the underlying master file for the zone using the values of the properties and parameters outlined in the init property. The implementation MUST also create the relevant dynamic zone configuration and state, load the zone, and serve it authoritatively.

It is not necessary for the catalog zone’s primary server to be the member zone’s primary server, however, the same server MUST be the primary server for all member zones within a given catalog zone.

The implementation may permit the following on a global, or per catalog basis, by way of suitable configuration parameters:

* The master file is ONLY created for the zone if the master file does not already exist

* The master file is NEVER created (effectively, the initialisation capability is disabled for this catalog or primary server, and the master file would be expected to exist as is the case before this document)

* The master file is ALWAYS created when a new member zone is added to the catalog zone, overwriting any existing master file for the zone

If a server is consuming a catalog zone and is configured to be primary server for the member zones therein, it MUST perform the actions as defined within this document. All other servers MUST ignore the additional records defined herein, as per "DNS Catalog Zones" ([RFC9432]) Section 3.

A number of sub-properties, expressed as labels within the bailiwick of the "init" label, define the initialisation parameters.

3.3. Start Of Authority (soa property)

The soa property is used to specify the SOA that will be applied to the created zone file for the member zone.

There MUST be one and ONLY one soa property record defined in a given scope.

Multiple soa property records within a given scope constitutes a broken catalog zone.

See the Section 4 for clarity on scope inheritance.

Absence of a soa property similarly constitutes a broken catalog zone.

3.3.1. Parameters

With the exception of the serial number, the SOA record parameters are supplied as three character-string values in the RDATA of a TXT resource record.

The first being the MNAME value, the second being the RNAME value, and the third containing the numeric timer values in decimal in the same order as expected in an SOA resource record, as defined in "Domain Names - Implementation and Specification" ([RFC1035]) Section 3.3.13.

All three MUST be present.

The MNAME and RNAME MUST be fully qualified, however a terminal @ label can be supplied to indicate the member zone’s name. In this case, the @ label MUST be substituted with the member zone’s name at zone file creation. See also Section 5.1.

3.3.2. Examples

```
soa.init.$CATZ 0 TXT ( "ns1.example.com"
      "hostmaster.example.com"
      "14400 900 2419200 3600" )

soa.init.$CATZ 0 TXT ( "ns1.@" "hostmaster.@"
      "14400 900 2419200 3600" )
```

3.4. Nameservers (ns property)

Actual NS records cannot be used, as we do not want to actually delegate outside of this catalog zone.

The nameserver parameters are supplied as key=value pairs in the RDATA of a TXT resource record, with the pairs separated by whitespace.

If the nameservers are in-bailiwick and address records are therefore required, suitable address records MUST be created in the member zone’s master file from the parameters specified.

If the nameservers are in-bailiwick of a zone in the catalog, and an address is not specified, this would result in a zone that will not load - this denotes a broken catalog zone.

There MUST be at least one ns property record.

Therefore, a catalog zone that contains no nameserver entries applicable to a given member zone constitutes a broken catalog zone.

The ns property can be specified multiple times, with one nameserver specified per entry.

3.4.1. name Parameter

The "name" parameter MUST be present, and contains the hostname of the nameserver as it is intended to appear in the corresponding NS record’s RDATA in the zone’s master file. See also Section 5.1.

The value of the "name" parameter MUST be compliant with "Domain Names - Implementation and Specification" ([RFC1035]) Section 3.3.11.

An ns property record that does not contain a "name" parameter consistutes a broken catalog zone.

3.4.2. ipv4 and ipv6 Parameters

The "ipv4" and "ipv6" parameters are OPTIONAL. They contain the IP address(es) of the hostname specified in the "name" parameter.

If the value in the "name" parameter is in-bailwick, and hence requires that the relevant address enties are also created in the zone, at least one of either the "ipv4" or "ipv6" parameters MUST be specified.

The value of the "ipv4" parameter, if present, MUST be a valid IPv4 address, compliant with "Domain Names - Implementation and Specification" ([RFC1035]) Section 3.4.1.

The value of the "ipv6" parameter, if present, MUST be a valid IPv6 address, compliant with "DNS Extensions to Support IP Version 6" ([RFC3596]) Section 2.1 and SHOULD use the representation defined in "A Recommendation for IPv6 Address Text Representation ([RFC5952]) Section 4.

An ns property record that contains an in-bailiwick name, but does not contain at least one address parameter constitutes a broken catalog zone.

Only records within the member zone are within the scope of this document; if the primary server is also coincidentally the primary server for a member zone’s parent, regardless of whether the parent zone is also a member zone, it is the responsibility of the parent zone’s administrator to ensure the delegation and any required glue resource records are present in the parent zone.

3.4.3. Examples

```
ns.init.$CATZ 0 TXT ( "name=some.name.server."
      "ipv4=192.0.2.1 ipv6=2001:db8::1" )

ns.init.$CATZ 0 TXT ( "name=another.name.server."
      "ipv4=192.0.2.129 ipv6=2001:db8:44::1" )
```

4. Member Zone Properties

The default properties outlined above can be overridden per member zone. If properties are specified in a more specific scope than the defaults, the most specific scope MUST be used.

A subset MAY be specified in a more specific scope, for example, the SOA could be omitted, and just the NS records specified.

The omitted properties would be inherited from the catalog level values.

```
<unique-N>.zones.$CATZ          0 PTR example.com.
soa.init.<unique-N>.zones.$CATZ 0 TXT ( "<mname>"
      "<rname>" "<refresh> <retry> <expire> <minimum>" )
ns.init.<unique-N>.zones.$CATZ  0 TXT ( "name=some.name.server. "
      "ipv4=192.0.2.1 ipv6=2001:db8::1" )
```

4.1. Change Of Ownership (coo property)

There is no change to the coo property; if the member zone changes ownership to another catalog, fundamentally, the zone's master file already exists.

The scope of this document is solely concerned with the initialisation of a new zone's master file, and so in the case of the zone changing ownership, the initialisation parameters MUST NOT be processed.

Noting that the primary server for a given catalog's member zones may not be the primary server for the catalog zone itself, nor the primary server for another catalog's member zones, operators should consider their implementation's configuration when planning a change of ownership operation.

5. Name Server Behaviour

5.1. General Behaviour

Some of the parameters specified in the initialisation properties contain domain-name values as defined in "Domain Names - Implementation and Specification" ([RFC1035]) Section 3.3, for example in the NS records and in the SOA. These will be used to specify values in the corresponding resource records in the member zone's file. The domain-name values MUST be fully qualified in the parameter specification in the property.

Similar to its use in "Domain Names - Implementation and Specification" ([RFC1035]) Section 5.1, a terminal @ label may be used as a short cut for the member zone's name, and in such cases, the terminal @ label MUST be substituted by the member zone name at the point of zone file creation.

5.2. Member Zone Removal

If the member zone is removed from the catalog zone, then the zone's master file MUST be removed along with related zone configuration and state data.

5.3. Zone-Associated State Reset

In the event of a zone state reset being carried out, the state of the zone's master file SHOULD be reset as if the file was being initialised for the first time per this document.

6. Implementation and Operational Notes

When configuring the use of catalog zones, implementations should give the operator the ability to indicate whether the catalog zone consumer is a primary or secondary for a given catalog's member zones.

Secondary servers are not interested in the properties and parameters defined within this document and MUST ignore them.

A given consumer MAY be primary or secondary, but of course cannot be both. A consumer that has undefined consumer status should default to secondary, which will result in backward compatibility with "DNS Catalog Zones" ([RFC9432]).

It is not mandatory that the primary server for a given catalog zone is also the primary server for the catalog's member zones.

As well as creating the undelying zone file and initial contents, the implementaion MUST also dynamically create and maintain related zone configuration and state. Further, the implementation MUST load and authoritatively serve the zone to clients.

7. Security Considerations

This document does not alter the security considerations outlined in "DNS Catalog Zones" ([RFC9432]).

8. IANA Considerations

IANA is requested to add the following entries to the registry:

Registry Name: DNS Catalog Zones Properties

Reference: this document

Property Prefix	Description	Status	Reference
init	Zone Initialisation Properties	Standards Track	this document
soa.init	Start Of Authority Property	Standards Track	this document
ns.init	Name Server Property	Standards Track	this document

Table 1: DNS Catalog Zones Properies Registry

Field meanings are unchanged from the definitions in "DNS Catalog Zones" ([RFC9432]).

9. 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/rfc/rfc1035>.

[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/rfc/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/rfc/rfc2136>.

[RFC3596] Thomson, S., Huitema, C., Ksinant, V., and M. Souissi, "DNS Extensions to Support IP Version 6", STD 88, RFC 3596, DOI 10.17487/RFC3596, October 2003, <https://www.rfc-editor.org/rfc/rfc3596>.

[RFC5952] Kawamura, S. and M. Kawashima, "A Recommendation for IPv6 Address Text Representation", RFC 5952, DOI 10.17487/RFC5952, August 2010, <https://www.rfc-editor.org/rfc/rfc5952>.

[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/rfc/rfc8174>.

[RFC9432] van Dijk, P., Peltan, L., Sur箪, O., Toorop, W., Monshouwer, C.R., Thomassen, P., and A. Sargsyan, "DNS Catalog Zones", RFC 9432, DOI 10.17487/RFC9432, July 2023, <https://www.rfc-editor.org/rfc/rfc9432>.

Appendix A. Examples

A.1. Catalog Zone Example

The following is an example catalog zone showing the additional properties and parameters as outlined in this document.

There are defaults specified for the SOA and NS records, which would be used by the example.com. zone.

The example.net. zone would utilise the default SOA record, but would utilise the more specific NS records.

The default nameservers are in-bailiwick of example.com, which is in the catalog, and so the address record details are supplied in order to facilitate the addition of the address records.

```
catz.invalid.                   0 SOA invalid. invalid. (
      1 3600 600 2419200 3600 )
catz.invalid.                   0 NS invalid.
soa.init.catz.invalid.          0 TXT ( "ns1.example.com."
      "hostmaster.example.com." "14400 900 2419200 3600" )
ns.init.catz.invalid.           0 TXT ( "name=ns1.example.com. "
      "ipv4=192.0.2.1 ipv6=2001:db8::1" )
ns.init.catz.invalid.           0 TXT ( "name=ns2.example.com. "
      "ipv4=192.0.2.2 ipv6=2001:db8::2" )
kahdkh6f.zones.catz.invalid.    0 PTR example.com.
hajhsjha.zones.catz.invalid.    0 PTR example.net.
ns.hajhsjha.zones.catz.invalid. 0 TXT "name=ns1.example.com"
ns.hajhsjha.zones.catz.invalid. 0 TXT ( "name=ns1.example.net "
      "ipv4=192.0.2.250 ipv6=2001:db8:ff::149" )
```

A.2. example.com Master File Example

This is the resulting zonefile for example.com as initilised from the above catalog zone example.

```
example.com.     3600 SOA ns1.example.com. hostmaster.example.com. (
                          1 14400 900 2419200 3600 )
example.com.     3600 NS   ns1.example.com.
example.com.     3600 NS   ns2.example.com.
ns1.example.com. 3600 A    192.0.2.1
ns1.example.com. 3600 AAAA 2001:db8::1
ns2.example.com. 3600 A    192.0.2.2
ns2.example.com. 3600 AAAA 2001:db8::2
```

A.3. example.net Master File Example

This is the resulting zonefile for example.net as initilised from the above catalog zone example.

```
example.net.     3600 SOA ns1.example.com. hostmaster.example.com. (
                           1 14400 900 2419200 3600 )
example.net.     3600 NS   ns1.example.com.
example.net.     3600 NS   ns1.example.net.
ns1.example.net. 3600 A    192.0.2.250
ns1.example.net. 3600 AAAA 2001:db8:ff::149
```

Appendix B. Author Notes/Thoughts

NB: To be removed by the RFC Editor prior to publication.

The term "Primary Master" ("DNS Dynamic Updates" ([RFC2136]) Section 1 is not applicable as the primary server noted in this document likely is NOT listed in the MNAME or NS.

B.1. Is catalog zones the right place for this?

Much consideration has been given as to whether the primary server should be consuming the/a catalog zone, rather than simply serving it to secondary servers for consumption.

It does feel a little bit like it muddies the waters between zone distribution and zone "provisioning" but:

1. In a catalog zone scenario, the catalog equally feels like the place for zone related parameters

2. It feels less like Dynamic Updates would be the right place for it, for example; Dynamic Updates has specific purpose around updating existing zones, which seems further removed from this functionality.

3. An API for _just_ zone initialisation feels like a big thing that would likely be overkill, and probably not get implemented, and would likely be a part of a wider implementation's general nameserver configuration and operations API, which is waaaaay beyond the scope of this document, and possibly beyond standardisation.

It may be considered that this is "nameserver configuration", however, it has strong parallels in this regard to the "configuration" on secondary servers, including such considerations as to which entities are allowed to notify and/or transfer the zone, as are conveyed to those secondary servers in "DNS Catalog Zones" ([RFC9432]). Indeed, much of the same configuration may be needed by or shared with the primary server for those same zones.

Implementing via an extension of catalog zones feels like it closes the gap in the end-to-end ecosystem whereby catalog zones + dynamic updates gives an end-to-end approach to the creation of a zone, its underlying master file, distribution of that zone to secondary servers, and the ongoing manipulation of records in the zone.

TODO - add more detail explaining the above, reasoning, etc...?

B.2. Properties

B.2.1. General

TODO: Do we need to signal the initial TTL of the records being added (SOA, NS, A, AAAA)... I think so... Could specifiy with an extra key=value pair, or could leave it to pick up from the SOA

Do we even need to supply these properties (soa, ns, etc) ? The reason an operator would be doing this would be because they want to create a zonefile with standard tooling and then immediatly commence making dynamic updates to the zone. An implementation could simply drop in basic SOA and NS with the expectation being that the operator then _replaces_ them.

B.2.1.1. ACLs...?

Do we need to consider a mechanism for providing also-notify, allow-transfer and similar style ACL configuration? It seems limiting to certain use cases, otherwise, to assume that server level configuration of such parameters is enough.

Some implementations facilitate this within the *.ext tree, but this is implementation specific, and unhelpful for operators using a combination of vendor products. Consider, for example, an operator using one vendor's product for unsigned primary, another for signing, and a third for serving the auth zones to the target network(s).

B.2.2. coo Property

Are there any considerations around change of ownership that need mentioning or documenting here...?

14/11 - section added to address this; is there anything else that needs clarifying or covering?

B.2.3. soa Property

Consideration was given as to whether things like SOA parameters should be individual records, but it seemed unnecessary to break them out and create the additional records.

Should we permit the property to be made up of multiple TXT records so long as a given parameter is not repeated?

Should we specify a SOA serial format? or an initial soa serial value...? If not, should we specify in the text, or leave it to the implemention, which may have a default, such as BIND's "serial-update-method"

Given that it is pretty much expected that the operator is going to start making changes to the zone via dynamic updates, it'd be reasonable to expect them to be able to set those parameters. Which does beg the question, do we need to specify soa and nameserver values at all, or just specify that the zone file is or is not to be created, and fill some template default values with the expectation that the operator would immediatly overwrite them with "correct" values...?

B.2.4. ns Property

Is there a circular dependency or race condition issue here...?

Do we need to consider the possibility of multiple IP addresses for a nameserver name? If so, maybe comma separate them like this: ipv4=192.0.2.1,192.0.2.2

Appendix C. Change Log

NB: To be removed by the RFC Editor prior to publication.

C.1. 00 - Initial draft

C.2. 00 - 01

Final feedback incorporated before wider circulation for discussion and feedback

Acknowledgments

The author wishes to thank Ray Bellis and Ruth Trevor-Allen for their reviews, feedback and discussion during the initial drafting of this document.

Author’s Address

Karl Dyson
Nominet UK
Minerva House
Edmund Halley Road
Oxford Science Park
Oxford
OX4 4DQ
United Kingdom
Email: karl.dyson@nominet.uk
URI: https://nominet.uk