Generalized DNS Notifications

Abstract

This document extends the use of DNS NOTIFY ([RFC1996] beyond conventional zone transfer hints, bringing the benefits of ad-hoc notifications to DNS delegation maintenance in general. Use cases include DNSSEC key rollovers hints, and quicker changes to a delegation's NS record set.

To enable this functionality, a method for discovering the receiver endpoint for such notification message is introduced, via the new NOTIFY record type.

TO BE REMOVED: This document is being collaborated on in Github at: https://github.com/peterthomassen/draft-ietf-dnsop-generalized-notify. The most recent working version of the document, open issues, etc. should all be available there. The authors (gratefully) accept pull requests.

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 5 September 2024.
1. Introduction

Traditional DNS notifications [RFC1996], which are here referred to as "NOTIFY(SOA)", are sent from a primary server to a secondary server to minimize the latter's convergence time to a new version of
the zone. This mechanism successfully addresses a significant inefficiency in the original protocol.

Today similar inefficiencies occur in new use cases, in particular delegation maintenance (DS and NS record updates). Just as in the NOTIFY(SOA) case, a new set of notification types will have a major positive benefit by allowing the DNS infrastructure to completely sidestep these inefficiencies. For additional context, see Appendix A.

No DNS protocol changes are introduced by this document. The mechanism instead makes use of a wider range of DNS messages allowed by the protocol. Future extension for further use cases (such as multi-signer key exchange) is possible.

Readers are expected to be familiar with DNSSEC, including [RFC4033], [RFC4034], [RFC4035], [RFC6781], [RFC7344], [RFC7477], [RFC7583], and [RFC8901].

1.1. Requirements Notation

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

1.2. Terminology

In the text below there are two different uses of the term "NOTIFY". One refers to the NOTIFY message, sent from a DNSSEC signer or name server to a notification target (for subsequent processing). We refer to this message as NOTIFY(RRtype) where the RRtype indicates the type of NOTIFY message (CDS or CSYNC).

The second is a proposed new DNS record type, with the suggested mnemonic "NOTIFY". This record is used to publish the location of the notification target. We refer to this as the "NOTIFY record".

2. Publication of Notification Targets

To use generalized notifications, it is necessary for the sender to know where to direct each NOTIFY message. This section describes the procedure for discovering that notification target.

Note that generalized NOTIFY messages are but one mechanism for improving the efficiency of automated delegation maintenance. Other alternatives such as contacting the parent via an API or DNS Update ([RFC2136]), may (or may not) be more suitable in individual cases. Like generalized notifications, they similarly
require a means for discovering where to send the API or DNS Update requests.

The scope for the publication mechanism is therefore wider than only to support generalized notifications, and a unified approach that works independently of the notification method is specified in this section.

2.1. Design Requirements

When the parent is interested in notifications for delegation maintenance (such as for DS or NS updates), a service will need to be made available for accepting these notifications. Depending on the context, this service may be run by the parent zone operator themselves, or by a designated entity who is in charge of handling the domain's delegation data (such as a domain registrar).

The simplest solution enabling straightforward discovery is for the parent to publish the address where it prefers to have notifications sent. Potential notification senders, knowing the name of the parent zone, can then simply look up that information.

It is strongly desirable that the notification sender is able to figure out where to send the NOTIFY via a single lookup, even when ignorant of the details of the parent-side business relationships (e.g., whether there is a registrar or not). The mechanism should thus enable the parent to (optionally) announce the notification endpoint in a delegation-specific way. (If the delegation is several labels deep, an extra query may be needed for identifying the parent.)

These requirements suggest making the endpoint discoverable at a child-specific name. The record there is expected to live at a wildcard name, unless the parent intends to publish a child-specific endpoint.

2.2. Signaling Method

Parents participating in the discovery scheme for the purpose of delegation maintenance notifications MUST publish endpoint information using the record type defined in Section 3, as described in this section.

The suggested mnemonic for the new record type is "DSYNC" and it is further described below.

If the parent itself performs CDS/CDNSKEY and CSYNC processing, or if the parent forwards the notifications internally to the designated party (such as as registrar), the following scheme is used:
It is also possible to publish child-specific records, where the wildcard label is replaced by the child's FQDN with the parent zone's labels stripped.

As an example, consider a registrar offering domains like example.se, delegated from se zone. If the registrar provides the notification endpoint, the parent may publish this information using the following scheme:

example._signal.se. IN DSYNC CDS scheme port endpoint.registrar.com

(Note that this is a generic method, allowing the parent to securely publish other sorts of information about a child that currently is not easily represented in DNS, such as the registrar's identity.)

The parent MAY synthesize records under the _signal domain. The _signal domain may be delegated to another nameserver dedicated for this purpose.

To accommodate indirect delegation management models (such as ICANN's RRR model), the parent's designated notification target may relay notifications to the registrar, e.g. via an EPP call, or by forwarding the NOTIFY(CDS) message directly. The same is true also for NOTIFY(CSYNC).

3. The DSYNC Record

3.1. Wire Format

The DSYNC RDATA wire format is encoded as follows:

```
 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 3 3 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| RRtype | Scheme | Port
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Target ... /
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

RRtype  The type of generalized NOTIFY that this DSYNC RR defines the desired target address for. For now, only CDS and CSYNC are supported values.

Scheme  The scheme for locating the desired notification address. The range is 0-255. This is an 8 bit unsigned integer. The value 0 is an error, and values 128-255 are reserved for private use. The
value 1 is described in this document, and all other values are currently unspecified.

**Port**  The port on the target host of the notification service. The range is 0-65535. This is a 16 bit unsigned integer in network byte order.

**Target**  The domain name of the target host providing the service of listening for generalized notifications of the specified type. This name MUST resolve to one or more address records.

### 3.2. Semantics

For now, the only scheme defined is scheme=1 with the interpretation that when a new CDS (or CDNSKEY or CSYNC) is published, a NOTIFY(CDS) or NOTIFY(CSYNC) should be sent to the address and port listed in the corresponding NOTIFY RRset.

Other schemes are possible, but are out of scope for this document.

Example:

```
parent. IN DSYNC CDS 1 5359 cds-scanner.parent.
parent. IN DSYNC CSYNC 1 5360 csync-scanner.parent.
```

From the perspective of this protocol, the NOTIFY(CDS) packet is simply sent to the parent's published notification address. However, should this turn out not to be sufficient, it is possible to define a new "scheme" that specifies alternative logic for dealing with such requirements. Description of internal processing in the recipient end or for locating the recipient are out of scope of this document.

### 3.3. Rationale

(RFC Editor: This subsection is to be removed before publication)

It may look like it's possible to store the same information in an SRV record. However, this would require indicating the RRtype via a label in the owner name, leading to name space pollution. It would also require changing the semantics of one of the integer fields of the SRV record.

Such overloading has not been a good idea in the past. Furthermore, as the generalized notifications are a new proposal with no prior deployments, there is an opportunity to avoid repeating mistakes.

The DSYNC record type also provides a cleaner solution for bundling all the new types of notification signaling in an RRset, like:
For DSYNC records indicating CDS/CDNSKEY/CSYNC notification targets, no special processing needs to be applied by the authoritative nameserver upon insertion of a DSYNC record. The nameserver can thus be "unaware".

Future use cases (such as for multi-signer key exchange) may require the nameserver to trigger special operations, for example when a DSYNC record is inserted during onboarding of a new signer. It seems cleaner and easier that such processing be associated with the insertion of a record of a new type, not an existing type like SRV.

4. Delegation Maintenance: CDS/CDNSKEY and CSYNC Notifications

Delegation maintenance notifications address the inefficiencies related to scanning child zones for CDS/CDNSKEY records [RFC7344]. (For an overview of the issues, see Appendix A.)

Delegation maintenance NOTIFY messages MUST be formatted as described in [RFC1996], with the qtype field replaced as appropriate.

To address the CDS/CDNSKEY dichotomy, the NOTIFY(CDS) message (with qtype=CDS) is defined to indicate any child-side changes pertaining to an upcoming update of DS records. Upon receipt of NOTIFY(CDS), the recipient (the parent registry or a registrar) SHOULD initiate the same DNS lookups and verifications that would otherwise be triggered based on a timer.

The CSYNC [RFC7477] inefficiency may be similarly treated, with the child sending a NOTIFY(CSYNC) message (with qtype=CSYNC) to an address where the parent (or a registrar) is listening to CSYNC notifications.

In both cases the notification will speed up processing times by providing the recipient with a hint that a particular child zone has published new CDS, CDNSKEY and/or CSYNC records.

4.1. Endpoint Discovery

To locate the target for outgoing delegation maintenance notifications, the notification sender MUST perform the following procedure:

1. Construct the lookup name, by injecting the _signal label after the first label of the delegation owner name.
2. Perform a lookup of type DSYNC for the lookup name, and validate the response if DNSSEC is enabled. If a DSYNC RRset results, return it.

3. When the query resulted in a negative response:

   *If the negative response indicates that the parent is more than one label away from the _signal label, construct a new lookup name by inserting the _signal label into the delegation owner name just before the parent zone labels inferred from the negative response, and go to step 2.

   For example, city.ise.mie.jp is delegated from jp (and not from ise.mie.jp or mie.jp!). The initial DSYNC query relating to it is thus directed at city._signal.ise.mie.jp. This is expected to result in a negative response from jp, and another query for city.ise.mie._signal.jp is then required;

   *Otherwise, if the lookup name has any labels in front of the _signal label, remove them to construct a new lookup name (such as _signal.jp), and go to step 2. (This is to enable zone structures without wildcards.)

   *Otherwise, return null (no notification target available).

4.2. Sending Notifications

When changing a CDS/CDNSKEY/CSYNC RRset in the child zone, the DNS operator SHOULD send a suitable NOTIFY message to the endpoint located as described in the previous section.

A NOTIFY message can only carry information about changes concerning one child zone. When there are changes to several child zones, the sender MUST send a separate notification for each one.

Because of the security model where a notification by itself never causes a change (it can only speed up the time until the next check for the same thing), the sender's identity is not crucial. This opens up the possibility of having an arbitrary party (e.g., a side-car service) send the notifications to the parent, thereby enabling this functionality even before the emergence of native support in nameserver software.

While the receiving side will often be a scanning service provided by the registry itself, it is expected that in the ICANN RRR model, some registries will prefer registrars to conduct CDS/CDNSKEY processing. In such cases, the registrar notification endpoint should be published in the parent zone, enabling the child to direct their notifications to the appropriate target. From the perspective of the child, it is inconsequential who's in charge of processing the
4.2.1. Timing

When a primary name server publishes a new RRset in the child, there will be a time delay until all publicly visible copies of the zone will have been updated. If the primary sends a NOTIFY at the exact time of publication of the new zone, there is a potential for the parent to attempt CDS/CDNSKEY/CSYNC processing before the updated zone is visible. In this case the parent may draw the wrong conclusion ("the CDS RRset has not been updated").

Having a delay between the publication of the new data and the check for the new data would alleviate this issue. However, as the parent has no way of knowing how quickly the child zone propagates, the appropriate amount of delay is uncertain.

It is therefore RECOMMENDED that the child delays sending NOTIFY messages to the recipient until a consistent public view of the pertinent records is ensured.

4.2.2. Rationale for Using the DNS Message Format

(RFC Editor: This subsection is to be removed before publication)

In the most common cases of using generalized notifications the recipient is expected to not be a nameserver, but rather some other type of service, like a CDS/CSYNC scanner.

However, this will likely not always be true. In particular it seems likely that in cases where the parent is not a large delegation-centric zone like a TLD, but rather a smaller zone with a small number of delegations there will not be separate services for everything and the recipient of the NOTIFY(CDS) or NOTIFY(CSYNC) will be an authoritative nameserver for the parent zone.

For this reason it seems most reasonable to stay within the the well documented and already supported message format specified in RFC 1996 and delivered over normal DNS transport, although not necessarily to port 53.

4.3. Processing of NOTIFY Messages

NOTIFY(CDS) messages carrying notification payloads (records) for several child zones MUST be discarded, as sending them is an error.

Upon receipt of a (potentially forwarded) valid NOTIFY(CDS) message for a particular child zone at the published address for CDS
notifications, the receiving side (parent registry or registrar) has two options:

1. Schedule an immediate check of the CDS and CDNSKEY RRsets as published by that particular child zone.

   If the check finds that the CDS/CDNSKEY RRset has indeed changed, the parent MAY reset the scanning timer for children for which NOTIFY(CDS) is received, or reduce the periodic scanning frequency accordingly (e.g. to every two weeks). This will decrease the scanning effort for the parent. If a CDS/CDNSKEY change is then detected (without having received a notification), the parent SHOULD clear that state and revert to the default scanning schedule.

   Parents introducing CDS/CDNSKEY scanning support at the same time as NOTIFY(CDS) support are not in danger of breaking children's scanning assumption, and MAY therefore use a low-frequency scanning schedule in default mode.

2. Ignore the notification, in which case the system works exactly as before. (One reason to do this may be a rate limit, see Section 5.)

   If the parent implements the first option, the convergence time (time between publication of a new CDS/CDNSKEY record in the child and propagation of the resulting DS) will decrease significantly, thereby providing improved service to the child zone.

   If the parent, in addition to scheduling an immediate check for the child zone of the notification, also choses to modify the scanning schedule (to be less frequent), the cost of providing the scanning service will be reduced.

   Upon receipt of a NOTIFY(CSYNC) to the published address for CSYNC notifications, the same options and considerations apply as for the NOTIFY(CDS).

5. Security Considerations

   The original NOTIFY specification sidesteps most security issues by not relying on the information in the NOTIFY message in any way, and instead only using it to "enter the state it would if the zone's refresh timer had expired" (Section 4.7 of [RFC1996]).

   This security model is reused for generalized NOTIFY messages. It therefore seems impossible to affect the behaviour of the recipient of the NOTIFY other than by hastening the timing for when different checks are initiated.
The receipt of a notification message will, in general, cause the
receiving party to perform one or more outbound queries for the
records of interest (for example, NOTIFY(CDS) will cause CDS/CDNSKEY
queries). When done via port 53, the size of these queries is
comparable to that of the NOTIFY messages themselves, rendering any
amplification attempts futile. The number of queries triggered per
notification is also limited by the requirement that a NOTIFY message
can refer to one child only.

However, when the outgoing query occurs via encrypted transport, some
amplification is possible, both with respect to bandwidth and
computational burden. In this case, the usual principle of bounding
the work, even under unreasonable events, applies.

Receivers therefore MUST implement rate limiting for notification
processing. It is RECOMMENDED to configure rate limiting
independently for both the notification's source IP address and the
name of the zone that is conveyed in the NOTIFY message. Rate
limiting also mitigates processing load from garbage notifications.

Alternative solutions (such as signing notifications and validating
their signatures) appear significantly more expensive without
tangible benefit.

6. IANA Considerations

Per [RFC8552], IANA is requested to create a new registry on the
"Domain Name System (DNS) Parameters" IANA web page as follows:

Name: DSYNC: Location of Synchronization Endpoints

Assignment Policy: Expert Review

Reference: (this document)

<table>
<thead>
<tr>
<th>DSYNC type</th>
<th>Scheme</th>
<th>Purpose</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>CDS</td>
<td>1</td>
<td>Delegation management</td>
<td>(this document)</td>
</tr>
<tr>
<td>CSYNC</td>
<td>1</td>
<td>Delegation management</td>
<td>(this document)</td>
</tr>
<tr>
<td></td>
<td>128-255</td>
<td>Reserved (private use)</td>
<td>(this document)</td>
</tr>
</tbody>
</table>

Table 1

7. Acknowledgements

Joe Abley, Mark Andrews, Christian Elmerot, Ólafur Guðmundsson, Paul
Wouters, Brian Dickson

8. Normative References

automation", Work in Progress, Internet-Draft, draft-


Appendix A. Efficiency and Convergence Issues in DNS Scanning

A.1. Original NOTIFY for Zone Transfer Nudging

[RFC1996] introduced the concept of a DNS Notify message which was used to improve the convergence time for secondary servers when a DNS zone had been updated in the primary. The basic idea was to augment the traditional "pull" mechanism (a periodic SOA query) with a "push" mechanism (a Notify) for a common case that was otherwise very inefficient (due to either slow convergence or wasteful overly frequent scanning of the primary for changes).

While it is possible to indicate how frequently checks should occur (via the SOA Refresh parameter), these checks did not allow catching zone changes that fall between checkpoints. [RFC1996] addressed the optimization of the time-and-cost trade-off between a secondary checking frequently for new versions of a zone, and infrequent checking, by replacing scheduled scanning with the more efficient NOTIFY mechanism.

A.2. Similar Issues for DS Maintenance and Beyond

Today, we have similar issues with slow updates of DNS data in spite of the data having been published. The two most obvious cases are CDS and CSYNC scanners deployed in a growing number of TLD registries. Because of the large number of child delegations, scanning for CDS and CSYNC records is rather slow (as in infrequent).

It is only a very small number of the delegations that will have updated CDS or CDNSKEY record in between two scanning runs. However, frequent scanning for CDS and CDNSKEY records is costly, and infrequent scanning causes slower convergence (i.e., delay until the DS RRset is updated).
Unlike in the original case, where the primary is able to suggest the scanning interval via the SOA Refresh parameter, an equivalent mechanism does not exist for DS-related scanning.

All of this above also applies to parents that offer automated NS and glue record maintenance via CSYNC scanning [RFC7477]. Again, given that CSYNC records change only rarely, frequent scanning of a large number of delegations seems disproportionately costly, while infrequent scanning causes slower convergence (delay until the delegation is updated).

While use of the NOTIFY mechanism for coordinating the key exchange in multi-signer setups [I-D.wisser-dnssec-automation] is conceivable, the detailed specification is left for future work.
Appendix B. Change History (to be removed before publication)

*draft-ietf-dnsop-generalized-notify-01

Reserve scheme values 128-255

Rename NOTIFY rrtype to DSYNC (to distinguish from NOTIFY message)

Describe endpoint discovery

Discussion on garbage notifications

More discussion on amplification risks

Clean-up, editorial changes

*draft-ietf-dnsop-generalized-notify-00

Revision after adoption.

*draft-thomassen-dnsop-generalized-dns-notify-02

Add rationale for staying in band

Add John as an author

*draft-thomassen-dnsop-generalized-dns-notify-01

Mention Ry-to-Rr forwarding to accommodate RRR model

Add port number flexiblity

Add scheme parameter

Drop SRV-based alternative in favour of new NOTIFY RR

Editorial improvements

*draft-thomassen-dnsop-generalized-dns-notify-00

Initial public draft.

Authors' Addresses

Johan Stenstam
The Swedish Internet Foundation

Email: johan.stenstam@internetstiftelsen.se

Peter Thomassen
deSEC, Secure Systems Engineering
Email: peter@desec.io

John Levine
Standcore LLC

Email: standards@standcore.com