

# draft-thomassen-generalised-dns-notify-01 (or -02...)

## Recap and Update Since IETF116

Peter Thomassen    Johan Stenstam

July 23, 2023

## Problem Statement #1: Scanning Issues

The (relatively new) RR types CDS and CSYNC rely on “someone” (typically the parent registry or the registrar) periodically **scanning** some subset of the child zones in search of indications that the child wants to update information in the parent zone.

- Scanning is resource consuming and costly.
- Scanning is inefficient (casting millions of nets return a very small number of fish).
- Slow scanning delays convergence.

Not every parent in the Universe is a TLD.

- A related issue is therefore how to best solve the problem of the parent detecting and acting on publication of CDS and/or CSYNC records in a child **without having to run a separate scanner**.

## Problem Statement #2: Multi-signer Issues

There is also a related problem in the context of so-called “multi-signer” setups.

- I.e. when there are multiple “signers” for the same zone, each with its own set of DNSSEC keys.
- Each signer must sign a complete DNSKEY RRset, containing the DNSKEY RRs from all signers.
- Whenever a signer does a key rollover it adds (and removes) DNSKEY RRs and if not caught by the other signers results in a bad state where the signature chain will be broken along certain paths.

Therefore frequent “scanning” of each signer’s DNSKEY RRset is needed to immediately catch changes to the DNSKEY RRsets, to minimize the risk of broken signature chains.

# The Proposal

- By generalising RFC 1996 NOTIFY(SOA) with NOTIFY(CDS) and NOTIFY(CSYNC) we believe that efficiency of CDS and CSYNC scanning may be improved significantly.
  - ▶ Both in the resource requirements, but primarily via faster convergence.
- By the further generalisation of NOTIFY(DNSKEY) a significant (almost show-stopper) issue in the multi-signer effort gets a clean solution and path forward.
- Generalising RFC1996 (NOTIFY) **does not constitute a protocol change**. This is already allowed (and works fine).

# What About The Security Model?

The security model is suggested to be the same as for RFC 1996 NOTIFY.

- I.e. a generalised NOTIFY does not change the verification logic in the recipient (like a scanner), it is only a hint that this particular child zone likely has recently published a CDS (or CSYNC).
- The point of the hint is to speed up convergence and thereby provide a better service to the child zone, not to change the verification logic.

# The Questions

In discussions on the mailinglist and also in other forums one question in particular comes up:

- **Is this needed?** Some TLDs see no problem with scanning, other TLDs have no intention of scanning. Some registrars would like to manage scanning themselves, etc.

Two other questions are:

- **Where to send** these new NOTIFY(RRtype) messages?
- If these NOTIFY(RRtype) are to be sent to a **service** rather than to a **nameserver**, then why keep the DNS packet format and transport?

Let's talk about each of these questions.

# Do We Need NOTIFY(CDS/CSYNC)? Isn't Scanning Enough?

It seems rather clear that this is a case of “one size doesn't fit all” .

- The requirements (and also abilities) vary for different parent zones. Some will never do scanning. Some already rely heavily on scanning.
- Non-TLD parent zones in particular likely do not want to maintain a separate scanning service for reasons of complexity. But acting on receipt of a NOTIFY(CDS) would be much easier.

From our point-of-view this is good and completely in line with how DNS has always worked. Each zone owner decides

- whether or not to sign the zone at all
- whether or not to publish a CDS record to simplify key rollovers
- whether or not to listen to NOTIFY(CDS) messages from child zones
- etc, etc.

## Where should these NOTIFY(RRtype) Be Sent?

The NOTIFY(CDS) and NOTIFY(CSYNC) are “vertical” (from the child to the parent). So a logical choice is to look for locator information in the parent zone.

After a lot of pondering we believe that the best alternative is to specify a new RR type (with the proposed mnemonic “NOTIFY”):

```
parent.example.  IN SOA  ...
...
parent.example.  IN NOTIFY CDS  1  5301  notifications.parent.
parent.example.  IN NOTIFY CSYNC 1  5302  notifications.parent.
```

The diagram shows three examples of NOTIFY records. The first is a standard SOA record. The second is a NOTIFY CDS record with a scheme of 1, port 5301, and destination notifications.parent. The third is a NOTIFY CSYNC record with a scheme of 1, port 5302, and destination notifications.parent. Red boxes highlight the scheme values (1), port values (5301, 5302), and the destination (notifications.parent.). Red arrows point from labels "Scheme", "Destination", and "Port" to these values.

- The “scheme” is a number indicating potential methods of locating where to send a NOTIFY. Only the value “scheme=1” is defined in the draft, with the method “send the NOTIFY to the specified port at the specified destination”.
- Other schemes may be defined in the future.

## Where should NOTIFY(DNSKEY) Be Sent?

In this case it isn't to the parent ("vertically"), it is more "sideways", eg. to a multi-signer controller or perhaps even directly to each of the other signers.

Therefore the location information can be located in the child zone, instead of in the parent zone:

```
child.parent.example.  IN SOA ...  
...  
child.parent.example.  IN NOTIFY DNSKEY 1 5303 music.service.
```

The diagram shows a DNS record for NOTIFY DNSKEY. The record is: `child.parent.example. IN NOTIFY DNSKEY 1 5303 music.service.` A red box labeled "Scheme" points to the value "1". A red box labeled "Destination" points to the value "music.service.".

- Only the value “scheme=1” is defined in the draft, with the method “send the NOTIFY(DNSKEY) to the specified port at the specified destination”.
- Other schemes may be defined in the future.

## How Would This Work in an RRR Model?

Under an RRR model it may not be possible for the registry to make updates to the registrant data. Sometimes updates may have to be done by the registrar, typically via EPP.

- Note that this is about the viability of the parent scanning, it has nothing to do with NOTIFY(CDS) and NOTIFY(CSYNC) which are only hints to make scanning more effective.

If the NOTIFY(CDS) / NOTIFY(CSYNC) shouldn't be sent to the parent, then it is not obvious where to send them.

- There may be lots of registrars and no single place to look (like `parent.example. IN NOTIFY` in this proposal).
- In the end, if the registrar wants to receive the notification, it is up to the registrar and registry to find the best mechanism, whether that is an EPP Message, NOTIFY Forwarding or something else.

# Why Keep DNS Message Format and Transport?

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

- However, for a smaller parent 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 likely be an authoritative nameserver for the parent zone.
- In controller-less multi-signer setups where there is no central controller. Instead the multi-signer algorithms are implemented inside or near each participating signer. It seems likely that the recipient in such cases sometimee will be an authoritative nameserver.

Therefore it seems reasonable to stay with 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.

# Summary

There are two different things being proposed:

- First: Generalise RFC 1996 NOTIFY(SOA) with NOTIFY(CDS), NOTIFY(CSYNC) and NOTIFY(DNSKEY)
  - ▶ This will facilitate significant improvements to the efficiency of CDS and CSYNC scanning.
  - ▶ Both in the resource requirements, but primarily via faster convergence.
  - ▶ For multi-signer setups a significant (show-stopper) issue gets a clean solution and path forward.
- Second: Define a new DNS RR type, tentatively called NOTIFY, that may be used to publish willingness to receive the new types of NOTIFY messages and also where they should be sent.

The proposed location mechanism (eg. defining a new NOTIFY record type used to locate the recipient of a generalised NOTIFY) would be a protocol change.

We would like this work to be adopted by the working group.

# Contact Information

**Peter Thomassen**

`peter@desec.io`

**Johan Stenstam**

`johan.stenstam@internetstiftelsen.se`

**Working code**

`https://github.com/johanix/gen-notify-test`