Workgroup: DNSOP Working Group
Internet-Draft:
draft-thomassen-dnsop-generalized-dns-notify-00
Published: 28 November 2022
Intended Status: Standards Track
Expires: 1 June 2023
Authors: J. Stenstam
The Swedish Internet Foundation
P. Thomassen
deSEC, Secure Systems Engineering

Generalized DNS Notifications

Abstract

Changes in CDS/CDNSKEY, CSYNC, and other records related to delegation maintenance are usually detected through scheduled scans run by the consuming party (e.g. top-level domain registry), incurring an uncomfortable trade-off between scanning cost and update latency.

A similar problem exists when scheduling zone transfers, and has been solved using the well-known DNS NOTIFY mechanism ([RFC1996]). This mechanism enables a primary nameserver to proactively inform secondaries about zone changes, allowing the secondary to initiate an ad-hoc transfer independently of when the next SOA check would be due.

This document extends the use of DNS NOTIFY 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 via NOTIFY(CDS) and NOTIFY(DNSKEY), and quicker changes to a delegation's NS record set via NOTIFY(CSYNC).

TO BE REMOVED: This document is being collaborated on in Github at: https://github.com/peterthomassen/draft-thomassen-dnsop-generalized-dns-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

Copyright Notice

Table of Contents

1. Introduction
   1.1. Requirements Notation
2. Costs and Dangers of Slow Convergence
3. Efficiency Issues with DNS Scanning vs. Notification
   3.1. CDS and CDNSKEY Scanners
   3.2. CSYNC Scanners
   3.3. Multi-Signer Setups
4. CDS/CDNSKEY and CSYNC Notifications
   4.1. Where to send CDS and CSYNC Notifications
   4.2. How to Interpret CDS and CSYNC Notifications
5. DNSKEY Notifications
   5.1. Where to send DNSKEY Notifications
   5.2. How to Interpret DNSKEY Notifications
6. Who Should Send the Notifications?
7. Timing Considerations
8. Open Questions
   8.1. SRV vs new record type
   8.2. Open Question For DNSKEY Notifications
9. Out of Scope
10. Security Considerations
11. IANA Considerations
12. Acknowledgements
13. Normative References
Appendix A. Change History (to be removed before publication)
Authors' Addresses

1. Introduction

This document introduces a generalization of the DNS NOTIFY mechanism described in [RFC1996].

Traditional DNS notifications, 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 there are several new cases where similar inefficiencies cause significant problems. However, 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.

There are no DNS protocol changes introduced by this document.

There are, however, proposals for how to interpret a wider range of DNS messages that are already allowed (but not used) by the protocol.

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**", "**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.

2. Costs and Dangers of Slow Convergence

[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).

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 the scalability issues of modern CDS and CSYNC scanners that are beginning to be 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).

Another use case is for so-called "multi-signer" setups where there are multiple, semi-independent, "signers" that each sign the same zone, but with individual sets of keys. One requirement for multi-signer setups to work is the ability to insert additional DNSKEY records in each signer's version of the zone. This is not the only multi-signer requirement, but it is the one that matters here. The problem here is that modern signers will commonly roll DNSSEC Zone Signing Keys automatically and without informing anyone, assuming a single-signer setup where ZSK rollovers are a local matter (as opposed to KSK rollovers that involve the parent).

However, when the ZSKs of one signer are rolled the other signers will be unaware of this event. Having the individual signers sign DNSKEY RRsets where the ZSKs are not the same create a possibly broken signature chain and is therefore a dangerous situation. Eventually the signers will catch up, as multi-signer operators import each other's DNSKEY public key(s) to be able to compute the combined RRsets that should be published, but the period between the ZSK rollover in one signer and the other signers becoming aware constitutes a period of potential brokenness.

3. Efficiency Issues with DNS Scanning vs. Notification

The original problem that [RFC1996] addressed was the problem of optimization between frequent checking for new versions of a zone by a secondary and infrequent checking. 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. NOTIFY replaces scheduled scanning with a more efficient mechanism.

In modern DNS infrastructure there are several new scanning based inefficiencies that did not exist at the time of the writing of [RFC1996].

3.1. CDS and CDNSKEY Scanners

An increasing number of TLD registries are implementing periodic scanning for CDS and CDNSKEY records in child zones. Because of the large number of delegations this is rather costly, especially as it is only a very small number of the signed delegations that will have updated the CDS or CDNSKEY record between two scanning runs.

Frequent scanning is costly. Infrequent scanning causes slower convergence (i.e. delay until the DS in the parent is updated).

Not every zone runs a CDS or CSYNC scanner (today). However, for those that do, the problem is actually worse than in the traditional primary to secondary case. The reason is that in the original case the primary is able to indicate how frequently changes are to be expected (via the SOA Refresh parameter). No equivalent mechanism exist for the new scanning services.

3.2. CSYNC Scanners

This is basically the same problem as for the CDS / CDNSKEY scanners: large number of delegations vs infrequent updates to the CSYNC record in the child zones.

Frequent scanning is costly. Infrequent scanning causes slower convergence (i.e. updated delegation information in the parent).

3.3. Multi-Signer Setups

[I-D.wisser-dnssec-automation] describes processes for managing signed zones using multiple semi-independent "signers" (i.e. services that take an unsigned zone, sign it using unique DNSKEYs and publish the zone on the public Internet). The setup most commonly discussed for multi-signer uses a "multi-signer controller", which is a separate service responsible for keeping the signing and delegation information in sync across multiple signers.

To keep information in sync, the signers need to be scanned for current information about the DNSKEY RRset in each signer. The problem is that modern "signers" will typically do DNSKEY rollovers (especially ZSK rollovers) automatically without informing anyone else, because a single-signer setup is often assume (in which case the ZSK rollover is a matter completely internal to the signer).

In the multi-signer case, this is not a correct assumption. It is therefore necessary to run frequent polls frequently to minimize the time window between one signer changing its version of the DNSKEY RRset and the controller noticing and re-synchronizing all signers.

Frequent scanning: costly. Infrequent scanning: slower convergence (i.e. longer potential inconsistency across signers).

4. CDS/CDNSKEY and CSYNC Notifications

The [RFC1996] NOTIFY message sends a SOA record in the Query Section. We refer to this as a NOTIFY(SOA). By generalizing the concept of DNS NOTIFY it is possible to address not only the original inefficiency (primary name server to secondary nameserver convergence) but also the problems with CDS/CDNSKEY, CSYNC and Multi-Signer scanning for zone updates.

The CDS/CDNSKEY inefficiency may be addressed by the child sending a NOTIFY(CDS) to an address where the parent listens for such notifications. To address the CDS/CDNSKEY dichotomy, NOTIFY(CDS) is defined to indicate any child-side changes pertaining to a upcoming update of DS records. Upon receipt of NOTIFY(CDS), the parent SHOULD initiate the same scan that would other be triggered based on a timer.

The CSYNC inefficiency may similarly be addressed by the child sinding a NOTIFY(CSYNC) to an address where the parent is listening to CSYNC notifications.

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

The DNS protocol already allows these new notification types, so no protocol change is required, the only thing needed is specification of where to send the new types of Notifies and how to interpret them in the receiving end.

4.1. Where to send CDS and CSYNC Notifications

In the case of NOTIFY(CDS) and NOTIFY(CSYNC) the ultimate recipient of the notification is the parent, to improve the speed and efficiency of the parent's CDS/CSYNC scanning. Because the name of the parent zone is known, and because the parent obviously controls the contents of its own zone the simplest solution is for the parent to publish the address where it prefers to have notifications sent. To be slightly more general than just using an address record we propose to use SRV records, as they are well suited to this.

In the zone parent.:

```
_cds-notifications.parent.   IN SRV n m  53 scanner.parent.
_csync-notifications.parent. IN SRV n m  53 scanner.parent.
```

4.2. How to Interpret CDS and CSYNC Notifications

Upon receipt of a NOTIFY(CDS) for a particular child zone at the published address for CDS notifications, the parent has two options:

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

   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.

If the parent implements the first option, the convergence time (time between publication of a new CDS 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 parent has exactly the same options to choose among as for the NOTIFY(CDS).

5. DNSKEY Notifications

In the multi-signer case the problem is slightly different, because it is not the parent that should be notified, but rather a third party.

5.1. Where to send DNSKEY Notifications

The question is how the multi-signer controller should inform the signer about where to send the notification. In the NOTIFY(CDS) and NOTIFY(CSYNC) cases the child (the service that signs the child zone) knows the name of the parent zone and can look up the notification address there. In the NOTIFY(DNSKEY) case this is not possible. However, what is possible is to look up the notification address in the child zone itself. This works out as a requirement for multi-signer setups to work is that at least one external party (the multi-signer controller) has the ability to insert or modify RRsets in the child zone. Therefore the controller should be able to insert a record that documents the wanted address for DNSKEY notifications.

In analogy with the other cases, but here in the zone child.parent.:

```
_dnskey-notification.child.parent. IN SRV n m 53 scanner.signerA.
_dnskey-notification.child.parent. IN SRV n m 53 scanner.signerB.
```

should act as a trigger for the signer that whenever there are changes to the DNSKEY RRset it should send a corresponding NOTIFY(DNSKEY) to all signers that have signaled interest via insertion of SRV records in the zone.

5.2. How to Interpret DNSKEY Notifications

The receipt of a NOTIFY(DNSKEY) is a hint that the DNSKEY RRset at the sending signer has been updated. If the child zone is not part of a multi-signer setup this should be a no-op that does not cause any action. In a multi-signer setup, though, this hint provides the recipient of the notification with the same three options as in the NOTIFY(CDS) and NOTIFY(CSYNC) cases: schedule an immediate check, delay regular checks and ignore.

6. Who Should Send the Notifications?

Because of the security model where a notification by itself never causes a change (it can only speed up the time until the next scheduled check for the same thing) who the sender is is not crucial.

This opens up the possibility of having the controller in a multi-signer setup send the CDS and CSYNC notifications to the parent, thereby enabling this new functionality even before the emergence of support for generalized DNS notifications in the name servers used for signing.

However, as multi-signer is strongly dependent on DNSKEY notifications (which, in case of an automatic ZSK rollover, can only be generated by the signing software), this does not alleviate the need for modifications also to signing software to support NOTIFY(DNSKEY).

7. Timing Considerations

When a primary name server publishes a new CDS RRset there will be a time delay until all copies of the zone world-wide will have been updated. Usually this delay is short (on the order of seconds), but it is larger than zero. If the primary sends a NOTIFY(CDS) at the exact time of publication of the new zone there is a potential for the parent to schedule the CDS check that the notification triggers faster than the zone propagates to all secondaries. 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, any such intentional delay should be implemented in the sending end, i.e. at the server generating the NOTIFY(CDS), not in the receiving end.

NOTIFY(CSYNC) has the same timing consideration as NOTIFY(CDS) while NOTIFY(DNSKEY) does not.

8. Open Questions

8.1. SRV vs new record type

In the case of the "vertical" Notify(CDS) and NOTIFY(CSYNC) nothing needs to be done by the nameservers serving the parent zone. So therefore the SRV records appear to be a near perfect match. In the case of the more "horizontal" NOTIFY(DNSKEY), however, the nameserver will have to act on the insertion of a NOTIFY(DNSKEY) instruction.

It may therefore be preferable to use a new record type, to make it possible to more easily associate the special processing needed with the new type rather than the standard SRV RRtype that occurs in completely other scenarios than are described here. If that is the case it could be that a new record type would provide a cleaner solution to all the new types of notification signaling. Eg.:

```
parent.         IN NOTIFY  CDS     53   scanner.parent.
parent.         IN NOTIFY  CSYNC   53   scanner.parent.
child.parent.   IN NOTIFY  DNSKEY  5300 music.service.provider.
```

8.2. Open Question For DNSKEY Notifications

In a multi-signer setup there are multiple signers. How will the multi-signer controller know which signer sent the notification? As the number of signers for a particular zone is low (typically 2 or possibly 3), there is no major problem caused by not knowing which signer sent the notification and instead always check all the signers for updates to the DNSKEY RRset.

9. Out of Scope

In the multi-signer case, where the same zone is signed by multiple semi-independent "signers" it is obviously necessary to devise means of keeping the non-DNSSEC contents of the zone in sync. That is out of scope of the multi-signer work and also out of scope of this document.

A corner case of this scoping regards making changes to the NS RRset. Individual signers (as in DNS service providers) may want to be able to make such changes independently of the other contents of the zone. See previous section.

10. Security Considerations

The original NOTIFY specification ([RFC1996]) sidesteps most security issues by not relying on the information in the NOTIFY message in any way but instead only use it to set the timer until the next scheduled check (for the SOA record) to zero. Therefore it is impossible to affect the behaviour of the recipient of the NOTIFY other than by changing the timing for when different checks are initiated.

This mechanism and security model is reused for all the generalized NOTIFY messages.

Another consideration is whether generalized DNS Notifications can be used as an amplification attack. The answer seems to be "NO", because the size of the generalized NOTIFY messages is mostly equal to the size of the response and it is also mostly equal to the size of the resulting outbound query (for the child zone's CDS, CSYNC or DNSKEY RRset).

Hence the amplification attack potential of generalized Notifications is the same as for the original NOTIFY(SOA), which has never been found to be useful for amplification attacks.

In any case, NOTIFY consumers MAY configure rate limits to address concerns about the impact of unsolicited NOTIFY messages.

11. IANA Considerations

Per [RFC8552], IANA is requested to create the "Generalized DNS Notifications" registry with the following columns and initial entries:

NOTIFY type	Location	relative to	Reference
CDS	_cds-notifications	parent	[this document]
CSYNC	_csync-notifications	parent	[this document]
DNSKEY	TODO	TODO	[this document]

Table 1

[TODO: (1) Detail out use cases? (2) Add entries via RFC or web form?]

12. Acknowledgements

Christian Elmerot

13. Normative References

[I-D.wisser-dnssec-automation] Wisser, U. and S. Huque, "DNSSEC automation", Work in Progress, Internet-Draft, draft-wisser-dnssec-automation-03, 6 March 2022, <https://datatracker.ietf.org/doc/html/draft-wisser-dnssec-automation-03>.

[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/rfc/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/rfc/rfc2119>.

[RFC4033] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "DNS Security Introduction and Requirements", RFC 4033, DOI 10.17487/RFC4033, March 2005, <https://www.rfc-editor.org/rfc/rfc4033>.

[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/rfc/rfc4034>.

[RFC4035] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "Protocol Modifications for the DNS Security Extensions", RFC 4035, DOI 10.17487/RFC4035, March 2005, <https://www.rfc-editor.org/rfc/rfc4035>.

[RFC6781] Kolkman, O., Mekking, W., and R. Gieben, "DNSSEC Operational Practices, Version 2", RFC 6781, DOI 10.17487/RFC6781, December 2012, <https://www.rfc-editor.org/rfc/rfc6781>.

[RFC7344] Kumari, W., Gudmundsson, O., and G. Barwood, "Automating DNSSEC Delegation Trust Maintenance", RFC 7344, DOI 10.17487/RFC7344, September 2014, <https://www.rfc-editor.org/rfc/rfc7344>.

[RFC7477] Hardaker, W., "Child-to-Parent Synchronization in DNS", RFC 7477, DOI 10.17487/RFC7477, March 2015, <https://www.rfc-editor.org/rfc/rfc7477>.

[RFC7583] Morris, S., Ihren, J., Dickinson, J., and W. Mekking, "DNSSEC Key Rollover Timing Considerations", RFC 7583, DOI 10.17487/RFC7583, October 2015, <https://www.rfc-editor.org/rfc/rfc7583>.

[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>.

[RFC8552]
Crocker, D., "Scoped Interpretation of DNS Resource Records through "Underscored" Naming of Attribute Leaves", BCP 222, RFC 8552, DOI 10.17487/RFC8552, March 2019, <https://www.rfc-editor.org/rfc/rfc8552>.

[RFC8901] Huque, S., Aras, P., Dickinson, J., Vcelak, J., and D. Blacka, "Multi-Signer DNSSEC Models", RFC 8901, DOI 10.17487/RFC8901, September 2020, <https://www.rfc-editor.org/rfc/rfc8901>.

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

*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