DNSOP Working Group
Internet-Draft
Intended status: Informational
Expires: 8 February 2026

S. Sheng

P. Thomassen
deSEC

7 August 2025

Operational Recommendations for DS Automation
draft-shetho-dnsop-ds-automation-02

Abstract

Enabling support for automatic acceptance of DS parameters from the Child DNS operator (via RFCs 7344, 8078, 9615) requires the parent operator, often a registry or registrar, to make a number of technical decisions. This document describes recommendations for new deployments of such DS automation.

Discussion Venues

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

Discussion of this document takes place on the Domain Name System Operations Working Group mailing list (dnsop@ietf.org), which is archived at https://mailarchive.ietf.org/arch/browse/dnsop/.

Source for this draft and an issue tracker can be found at https://github.com/peterthomassen/draft-shetho-dnsop-ds-automation.

Status of This Memo

Table of Contents

1. Introduction

[RFC7344], [RFC8078], [RFC9615] automate DNSSEC delegation trust maintenance by having the child publish CDS and/or CDNSKEY records which indicate the delegation's desired DNSSEC parameters ("DS automation").

Parental Agents using these protocols have to make a number of technical decisions relating to issues of validity checks, timing, error reporting, locks, etc. Additionally, when using the RRR model (as is common amongst top-level domains), both the registrar and the registry can effect parent-side changes to the delare reviewers not egation. In such a situation, additional questions arise.

Not all existing DS automation deployments have made the same choices with respect to these questions, leading to somewhat inconsistent behavior. From the perspective of a domain holder with domain names under various TLDs, this may be unexpected and confusing.

New deployments of DS automation therefore SHOULD follow the recommendations set out in this document, to achieve a more uniform treatment across suffixes and to minimize user surprise. The recommendations are intended to provide baseline safety and uniformity of behavior across parents. Registries with additional requirements on DS update checks MAY implement any additional checks in line with local policy.

In the following sections, operational questions are first raised and answered with the corresponding recommendations. Each section is concluded with an analysis of its recommendations, and related considerations.

Readers are expected to be familiar with DNSSEC [RFC9364][RFC9615][I-D.draft-ietf-dnsop-generalized-notify-09]. For terminology, see Appendix A.

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. Validity Checks and Safety Measures

This section provides recommendations to address the following questions:

* What kind of validity checks should be performed on DS parameters?
* Should these checks be performed upon acceptance, or also continually when in place?
* How do TTLs and caching impact DS provisioning? How important is timing in a child key change?

2.1. Recommendations

1. Entities performing automated DS maintenance SHOULD verify

    a. the consistency of DS update requests across all authoritative nameservers in the delegation [I-D.ietf-dnsop-cds-consistency], and

    b. that the resulting DS record set would not break DNSSEC validation if deployed,

    and cancel the update if the verifications do not succeed.

2. Parent operators (such as registries) SHOULD reduce a DS record set’s TTL to a value between 515 minutes when the set of records is changed, and restore the normal TTL value at a later occasion (but not before the previous DS RRset’s TTL has expired).

2.2. Analysis

2.2.1. Continuity of Resolution

To maintain the basic resolution function, it is important to avoid the deployment of flawed DS record sets in the parent zone. It is therefore desirable for the Parent to verify that the DS record set resulting from an automated (or even manual) update does not break DNSSEC validation if deployed, and otherwise cancel the update.

This is best done by

1. verifying that consistent CDS/CDNSKEY responses are served by all of the delegation's nameservers [I-D.ietf-dnsop-cds-consistency];

2. verifying that the resulting DS RRset does not break the delegation if applied ([RFC7344], Section 4.1), i.e., that it provides at least one valid path for validators to use ([RFC6840], Section 5.11). This is the case if there is at least one DS record referencing a key that actually signs the child's DNSKEY RRset, where the digest type and signing algorithm are listed as mandatory ("MUST") in the "Implement for DNSSEC Validation" columns of the relevant IANA registries [DS-IANA] and [DNSKEY-IANA].

TODO Should checks be done continually? (Why is that the parent's task?) Or on demand, e.g., on a no-op NOTIFY? Even when no update was requested, it may be worthwhile to occasionally check whether the current DS contents would be accepted today (see Section 2), and communicate any failures without changing the published DS record set.

TODO Maybe RECOMMEND periodical rechecks and allow requesting recheck in case of operational difficulties (large parent). Allow the parent to communicate interval? See draft-berra-dnsop-announce-scanner.

2.2.2. TTLs and Caching

To further reduce the impact of any misconfigured DS record set — be it from automated or from manual provisioning — the option to quickly roll back the delegation's DNSSEC parameters is of great importance. This is achieved by setting a comparatively low TTL on the DS record set in the parent domain, at the cost of reduced resiliency against nameserver unreachability due to the earlier expiration of cached records. The availability risk can be mitigated by limiting such TTLs to a brief time period after a change to the DS configuration, during which rollbacks are most likely to occur.

Registries therefore should significantly lower the DS RRset's TTL for some time following an update. Pragmatic values for the reduced TTL value range between 515 minutes. Such low TTLs might be expected to cause increased load on the corresponding authoritative nameservers; however, recent research has demonstrated them to have negligible impact on the overall load of a registry's authoritative nameserver infrastructure [LowTTL].

The reduction should be in effect at least until the previous DS record set has expired from caches, that is, the period during which the low-TTL is applied should exceed the normal TTL value. The routine re-signing of the DS RRset (usually after a few days) provides a convenient opportunity for resetting the TTL. When using EPP, the server MAY advertise its TTL policy via the domain <info> command described in [RFC9803], Section 2.1.1.2.

While this approach enables quick rollbacks, timing of the desired DS update process itself is largely governed by the previous DS RRset's TTL, and therefore does not generally benefit from an overall speed-up. Note also that nothing is gained from first lowering the TTL of the old DS RRset: such an additional step would, in fact, require another wait period while resolver caches adjust. For the sake of completeless, there likewise is no point to increasing any DS TTL values beyond their normal value.

3. Reporting and Transparency

This section provides recommendations to address the following question:

* Should a failed (or even successful) DS update trigger a notification to anyone?

3.1. Recommendations

TODO consider practicality of email notifications, or what else to do, see https://mailarchive.ietf.org/arch/msg/dnsop/aXGm1FuEPF5TV1PsVD2zK2fMBvY/

TODO "in accordance with the communication preferences established by the child zone operator"? Should there be an ability for the zone operator to establish their communication (who and how) preferences? How would that be signaled?

1. For certain DS updates (see analysis (Section 3.2)) and for DS deactivation, relevant points of contact known to the zone operator SHOULD be notified.

2. For error conditions, the domain's technical contact and the DNS operator serving the affected Child zone SHOULD be first notified. The registrant SHOULD NOT be notified unless the problem persists for a prolonged amount of time (e.g., three days).

3. Notifications to humans SHOULD be done via email. Child DNS operators SHOULD be notified using a report query [RFC9567] to the agent domain as described in ([I-D.draft-ietf-dnsop-generalized-notify-09], Section 4). The same condition SHOULD NOT be reported unnecessarily frequently to the same recipient.

4. In the RRR model, registries performing DS automation SHOULD inform the registrar of any DS record changes via the EPP Change Poll Extension [RFC8590] or a similar channel.

5. The currently active DS configuration as well as the history of DS updates SHOULD be made accessible to the registrant (or their designated party) through the customer portal available for domain management.

3.2. Analysis

When accepting or rejecting a DS update, it cannot be assumed that relevant parties are aware of what's happening. For example, a registrar may not know when an automatic DS update is performed by the registry. Similarly, a Child DNS operator may not be aware when their CDS/CDNSKEY RRsets are out of sync across nameservers, thus being ignored. Early reporting of such conditions helps involved parties to act appropriately and in a timely manner.

A delegation can break even without an update request to the DS record set. This may occur during key rollovers ([RFC6781], Section 4.1) when the Child DNS operator proceeds to the next step early, without verifying that the delegation's DS RRset is in the expected state. For example, when an algorithm rollover is performed and the old signing algorithm is removed from the Child zone before the new DS record is added, validation errors may result. TODO Reduce fearmongering: find numbers, better example, or loosen up wording.

Entities performing automated DS maintenance should report on conditions they encounter. The following success situations may be of particular interest:

1. A DS RRset has been provisioned

    a. manually;

    b. due to commencing DS automation (either via DNSSEC bootstrapping, or for the first time after a manual change; see Section 5);

   c. automatically, as an update to an existing DS RRset that had itself been automatically provisioned.

2. The DS RRset has been removed

   a. manually;

   b. automatically, using a delete signal ([RFC8078], Section 4).

In addition, there are error conditions worthy of being reported:

3. A pending DS update cannot be applied due to an error condition. There are various scenarios where an automated DS update might have been requested, but can't be fulfilled. These include:

   a. The new DS record set would break validation/resolution or is not acceptable to the Parent for some other reason (see Section 2).

   b. A lock prevents DS automation (see Section 4).

4. No DS update is due, but it was determined that the Child zone is no longer compatible with the existing DS record set (e.g., DS RRset only references non-existing keys).

For these reportworthy cases, the entity performing DS automation would be justified to attempt communicating the situation. Potential recipients are:

* Child DNS operator, preferably by making a report query [RFC9567] to the agent domain listed in the EDNS0 Report-Channel option of the DS update notification that triggered the DS update ([I-D.draft-ietf-dnsop-generalized-notify-09], Section 4), or alternatively via email to the address contained in the child zone's SOA RNAME field (see Sections 3.3.13 and 8 of [RFC1035]);

* Registrar (if DS automation is performed by the registry);

* Registrant (domain holder; in non-technical language, such as "DNSSEC security for your domain has been enabled and will be maintained automatically") or technical contact, via email.

For manual updates (case 1a), commencing DS automation (case 1b), and deactivating DNSSEC (case 2), it seems worthwhile to notify both the domain's technical contact and the registrant. This will typically lead to one notification during normal operation of a domain. (Case 1c, the regular operation of automation, is not an interesting condition to report to a human.)

For error conditions (cases 3 and 4), the registrant need not always be involved. It seems advisable to first notify the domain's technical contact and the DNS operator serving the affected Child zone, and only if the problem persists for a prolonged amount of time (e.g., three days), notify the registrant.

When the RRR model is used and the registry performs DS automation, the registrar should always stay informed of any DS record changes, e.g., via the EPP Change Poll Extension [RFC8590].

The same condition SHOULD NOT be reported unnecessarily frequently to the same recipient (e.g., no more than twice in a row). For example, when CDS and CDNSKEY records are inconsistent and prevent DS initialization, the registrant may be notified twice. Additional notifications may be sent with some back-off mechanism (in increasing intervals).

The history of DS updates SHOULD be kept and, together with the currently active configuration, be made accessible to the registrant (or their designated party) through the customer portal available for domain management.

4. Registration Locks

This section provides recommendations to address the following question:

* How does DS automation interact with other registration state parameters, such as registration locks?

4.1. Recommendations

1. To secure ongoing operations, automated DS maintenance SHOULD NOT be suspended based on a registrar update lock alone (such as EPP status clientUpdateProhibited).

2. When performed by the registry, automated DS maintenance SHOULD NOT be suspended based on a registry update lock alone (such as EPP status serverUpdateProhibited).

4.2. Analysis

Registries and registrars can set various types of locks for domain registrations, usually upon the registrant's request. An overview of standardized locks using EPP, for example, is given in Section 2.3 of [RFC5731]. Some registries may offer additional (or other) types of locks whose meaning and set/unset mechanisms are defined according to a proprietary policy.

While some locks clearly should have no impact on DS automation (such as transfer or deletion locks), other types of locks, in particular "update locks", deserve a closer analysis.

4.2.1. Registrar vs. Registry Lock

A registrar-side update lock (such as clientUpdateProhibited in EPP) protects against various types of accidental or malicious change (like unintended changes through the registrar's customer portal). Its security model does not prevent the registrar's (nor the registry's) actions. This is because a registrar-side lock can be removed by the registrar without an out-of-band interaction.

Under such a security model, no tangible security benefit is gained by preventing automated DS maintenance based on a registrar lock alone, while preventing it would make maintenance needlessly difficult. It therefore seems reasonable not to suspend automation when such a lock is present.

When a registry-side update lock is in place, the registrar cannot apply any changes (for security or delinquency or other reasons). However, it does not protect against changes made by the registry itself. This is exemplified by the serverUpdateProhibited EPP status, which demands only that the registrar's "[r]equests to update the object [...] MUST be rejected" ([RFC5731], Section 2.3). This type of lock therefore precludes DS automation by the registrar, while registry-side automation may continue.

4.2.2. Detailed Rationale

Pre-DNSSEC, it was possible for a registration to be set up once, then locked and left alone (no maintenance required). With DNSSEC comes a change to this operational model: the configuration may have to be maintained in order to remain secure and operational. For example, the Child DNS operator may switch to another signing algorithm if the previous one is no longer deemed appropriate, or roll its SEP key for other reasons. Such changes entail updating the delegation's DS records.

If authenticated, these operations do not qualify as accidental or malicious change, but as legitimate and normal activity for securing ongoing operation. The CDS/CDNSKEY method provides an automatic, authenticated means to convey DS update requests. The resulting DS update is subject to the parent's acceptance checks; in particular, it is not applied when it would break the delegation (see Section 2).

Given that registrar locks protect against unintended changes (such as through the customer portal) while not preventing actions done by the registrar (or the registry) themself, such a lock is not suitable for defending against actions performed illegitimately by the registrar or registry (e.g., due to compromise). Any attack on the registration data that is feasible in the presence of a registrar lock is also feasible regardless of whether DS maintenance is done automatically; in other words, DS automation is orthogonal to the attack vector that a registrar lock protects against.

Considering that automated DS updates are required to be authenticated and validated for correctness, it thus appears that honoring such requests, while in the registrant’s interest, comes with no additional associated risk. Suspending automated DS maintenance therefore does not seem justified.

Following this line of thought, some registries (e.g., .ch/.cz/.li) today perform automated DS maintenance even when an "update lock" is in place. Registries offering proprietary locks should carefully consider for each lock whether its scope warrants suspension.

In case of a domain not yet secured with DNSSEC, automatic DS initialization is not required to maintain ongoing operation; however, authenticated DNSSEC bootstrapping [RFC9615] might be requested. Besides being in the interest of security, the fact that a Child is requesting DS initialization through an authenticated method expresses the registrant’s intent to have the delegation secured.

Further, some domains are equipped with an update lock by default. Not honoring DNSSEC bootstrapping requests then imposes an additional burden on the registrant, who has to unlock and relock the domain in order to facilitate DS provisioning after registration. This is a needless cost especially for large domain portfolios. It is also unexpected, as the registrant already has arranged for the necessary CDS/CDNSKEY records to be published. It therefore appears that DS initialization and rollovers should be treated the same way with respect to locks.

5. Multiple Submitting Parties

This section provides recommendations to address the following questions:

* How are conflicts resolved when DS parameters are accepted through multiple channels (e.g. via a conventional channel and via automation)?

* In case both the registry and the registrar are automating DS updates, how to resolve potential collisions?

5.1. Recommendations

1. Registries and registrars SHOULD provide a another (e.g., manual) channel for DS maintenance in order to enable recovery when the Child has lost access to its signing key(s). This out-of-band channel is also needed when a DNS operator does not support DS automation or refuses to cooperate.

2. DS update requests SHOULD be executed immediately after verification of their authenticity, regardless of whether they are received in-band or via an out-of-band channel.

3. Only when the entire DS record set has been removed, SHOULD DS automation be suspended, in order to prevent accidental re-initialization of the DS record set when the registrant intended to disable DNSSEC.

4. In all other cases where a non-empty DS record set is provisioned through whichever channel, DS automation SHOULD NOT (or no longer) be suspended (including after an earlier removal).

5. In the RRR model, if the registry performs DS automation, the registry SHOULD notify the registrar of all DS updates (see also Recommendation 4 under Section 3).

6. In the RRR model, registries SHOULD NOT perform automated DS maintenance if it is known that the registrar does not support DNSSEC.

5.2. Analysis

In the RRR model, there are multiple channels through which DS parameters can be accepted:

* The registry can retrieve information about an intended DS update automatically from the Child DNS Operator and apply the update directly;

* The registrar can retrieve the same and relay it to the registry;

* Registrars or (less commonly) registries can obtain the information from the registrant through another channel (such as a non-automated "manual update" via webform submission), and relay it to the registry.

There are several considerations in this context, as follows.

5.2.1. Necessity of Non-automatic Updates

Under special circumstances, it may be necessary to perform a non-automatic DS update. One important example is when the key used by for authentication of DS updates is destroyed: in this case, an automatic key rollover is impossible as the Child DNS operator can no longer authenticate the associated information. Another example is when several providers are involved, but one no longer cooperates (e.g., when removing a provider from a multi-provider setup). Disabling manual DS management interfaces is therefore strongly discouraged.

Similarly, when the registrar is known to not support DNSSEC (or to lack a DS provisioning interface), it seems adequate for registries to not perform automated DS maintenance, in order to prevent situations in which a misconfigured delegation cannot be repaired by the registrant.

5.2.2. Impact of Non-automatic Updates: When to Suspend Automation

When an out-of-band (e.g., manual) DS update is performed while CDS/CDNSKEY records referencing the previous DS RRset’s keys are present, the delegation’s DS records may be reset to their previous state at the next run of the automation process. This section discusses in which situations it is appropriate to suspend DS automation after such a non-automatic update.

One option is to suspend DS automation after a manual DS update, but only until a resumption signal is observed. In the past, it was proposed that seeing an updated SOA serial in the child zone may serve as a resumption signal. However, as any arbitrary modification of zone contents — including the regular updating of DNSSEC signature validity timestamps — typically causes a change in SOA serial, resumption of DS automation after a serial change comes with a high risk of surprise. Additional issues arise if nameservers have different serial offsets (e.g., in a multi-provider setup). It is therefore advised to not follow this practice.

Note also that "automatic rollback" due to old CDS/CDNSKEY RRsets can only occur if they are signed with a key authorized by one of new DS records. Validity checks described in Section 2 further ensure that updates do not break validation.

All in all:

* It appears advisable to generally not suspend in-band DS automation when an out-of-band DS update has occurred.

* An exception from this rule is when the entire DS record set was removed, in which case the registrant likely wants to disable DNSSEC for the domain. DS automation should then be suspended so that automatic re-initialization (bootstrapping) does not occur.

* In all other cases, any properly authenticated DS updates received, including through an automated method, are to be considered as the current intent of the domain holder.

5.2.3. Concurrent Automatic Updates

When the RRR model is used, there is a potential for collision if both the registry and the registrar are automating DS provisioning by scanning the child for CDS/CDNSKEY records. No disruptive consequences are expected if both parties perform DS automation. An exception is when during a key rollover, registry and registrar see different versions of the Child's DS update requests, such as when CDS/CDNSKEY records are retrieved from different vantage points. Although unlikely due to Recommendation 1a of Section 2, this may lead to flapping of DS updates; however, it is not expected to be harmful as either DS RRset will allow for the validation function to continue to work, as ensured by Recommendation 1b of Section 2. The effect subsides as the Child's state eventually becomes consistent (roughly, within the child's replication delay); any flapping until then will be a minor nuisance only.

The issue disappears entirely when scanning is replaced by notifications that trigger DS maintenance through one party's designated endpoint [I-D.draft-ietf-dnsop-generalized-notify-09], and can otherwise be mitigated if the registry and registrar agree that only one of them will perform scanning.

As a standard aspect of key rollovers (RFC 6781), the Child DNS operator is expected to monitor propagation of Child zone updates to all authoritative nameserver instances, and only proceed to the next step once replication has succeeded everywhere and the DS record set was subsequently updated (and in no case before the DS RRset's TTL has passed). Any breakage resulting from improper timing on the Child side is outside of the Parent's sphere of influence, and thus out of scope of DS automation considerations.

6. CDS vs. CDNSKEY

This section provides recommendations to address the following question:

* Are parameters for DS automation best conveyed as CDNSKEY or CDS records, or both?

6.1. Recommendations

1. DNS operators SHOULD publish both CDNSKEY records as well as CDS records, and follow best practice for the choice of hash digest type [DS-IANA].

2. Parents, independently of their preference for CDS or CDNSKEY, SHOULD require publication of both RRsets, and SHOULD NOT proceed with updating the DS RRset if one is found missing or inconsistent with the other.

3. Registries (or registrars) scanning for CDS/CDNSKEY records SHOULD verify that any published CDS and CDNSKEY records are consistent with each other, and otherwise cancel the update [I-D.ietf-dnsop-cds-consistency].

TODO clarify that this does not prevent parent from chosing a digest type that’s not in CDS (separate recommendation?)

6.2. Analysis

DS records can be generated from information provided either in DS format (CDS) or in DNSKEY format (CDNSKEY). While the format of CDS records is identical to that of DS records (so the record data be taken verbatim), generation of a DS record from CDNSKEY information involves computing a hash.

Whether a Parent processes CDS or CDNSKEY records depends on their preference:

* Conveying (and storing) CDNSKEY information allows the Parent to control the choice of hash algorithms. The Parent may then unilaterally regenerate DS records with a different choice of hash algorithm(s) whenever deemed appropriate.

* Conveying CDS information allows the Child DNS operator to control the hash digest type used in DS records, enabling the Child DNS operator to deploy (for example) experimental hash digests and removing the need for registry-side changes when new digest types become available.

The need to make a choice in the face of this dichotomy is not specific to DS automation: even when DNSSEC parameters are relayed to the Parent through conventional channels, the Parent has to make some choice about which format(s) to accept.

Some registries have chosen to prefer DNSKEY-style input which seemingly comes with greater influence on the delegation's security properties (in particular, the DS hash digest type). It is noted that regardless of the choice of input format, the Parent cannot prevent the Child from following insecure cryptographic practices (such as insecure key storage, or using a key with insufficient entropy). Besides, as the DS format contains a field indicating the hash digest type, objectionable ones (such as those outlawed by [DS-IANA]) can still be rejected even when ingesting CDS records, by inspecting that field.

The fact that more than one input type needs to be considered burdens both Child DNS operators and Parents with the need to consider how to handle this dichotomy. Until this is addressed in an industry-wide manner and one of these mechanisms is deprecated in favor of the other, both Child DNS operators and Parents implementing automated DS maintenance should act as to maximize interoperability:

* As there exists no protocol for Child DNS Operators to discover a Parent's input format preference, it seems best to publish both CDNSKEY as well as CDS records, in line with [RFC7344], Section 5. The choice of hash digest type should follow current best practice [DS-IANA].

* Parents, independently of their input format preference, are advised to require publication of both CDS and CDNSKEY records, and to enforce consistency between them, as determined by matching CDS and CDNSKEY records using hash digest algorithms whose support is mandatory [DS-IANA]. (Consistency of CDS records with optional or unsupported hash digest types is not required.)

Publishing the same information in two different formats is not ideal. Still, it is much less complex and costly than burdening the Child DNS operator with discovering each Parent's policy; also, it is very easily automated. Operators should ensure that published RRsets are consistent with each other.

By rejecting the DS update if either RRset is found missing or inconsistent with the other, Child DNS operators are held responsible when publishing contradictory information. At the same time, Parents can retain whatever benefit their policy choice carries for them, while facilitating a later revision of that choice. This approach also simplifies possible future deprecation of one of the two formats, as no coordination or implementation changes would be needed on the child side.

7. IANA Considerations

This document has no IANA actions.

8. Security Considerations

This document considers security aspects throughout, and has not separate considerations.

9. Acknowledgments

The authors would like to thank the SSAC members who wrote the [SAC126] report on which this document is based.

In order of first contribution or review: Barbara Jantzen, Matt Pounsett, Matthijs Mekking, Ondej Caletka, Oli Schacher, Kim Davies

10. References

10.1. Normative References

[DNSKEY-IANA]
IANA, "DNS Security Algorithm Numbers", n.d., <https://www.iana.org/assignments/dns-sec-alg-numbers/dns-sec-alg-numbers.xml#dns-sec-alg-numbers-1>.

[DS-IANA] IANA, "Delegation Signer (DS) Resource Record (RR) Type Digest Algorithms", n.d., <http://www.iana.org/assignments/ds-rr-types>.

[I-D.draft-ietf-dnsop-generalized-notify-09]
Stenstam, J., Thomassen, P., and J. R. Levine, "Generalized DNS Notifications", Work in Progress, Internet-Draft, draft-ietf-dnsop-generalized-notify-09, 19 March 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-dnsop-generalized-notify-09>.

[I-D.ietf-dnsop-cds-consistency]
Thomassen, P., "Clarifications on CDS/CDNSKEY and CSYNC Consistency", Work in Progress, Internet-Draft, draft-ietf-dnsop-cds-consistency-08, 1 August 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-dnsop-cds-consistency-08>.

[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>.

[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>.

[RFC8078] Gudmundsson, O. and P. Wouters, "Managing DS Records from the Parent via CDS/CDNSKEY", RFC 8078, DOI 10.17487/RFC8078, March 2017, <https://www.rfc-editor.org/rfc/rfc8078>.

[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>.

[RFC8590] Gould, J. and K. Feher, "Change Poll Extension for the Extensible Provisioning Protocol (EPP)", RFC 8590, DOI 10.17487/RFC8590, May 2019, <https://www.rfc-editor.org/rfc/rfc8590>.

[RFC9364] Hoffman, P., "DNS Security Extensions (DNSSEC)", BCP 237, RFC 9364, DOI 10.17487/RFC9364, February 2023, <https://www.rfc-editor.org/rfc/rfc9364>.

[RFC9567] Arends, R. and M. Larson, "DNS Error Reporting", RFC 9567, DOI 10.17487/RFC9567, April 2024, <https://www.rfc-editor.org/rfc/rfc9567>.

[RFC9615] Thomassen, P. and N. Wisiol, "Automatic DNSSEC Bootstrapping Using Authenticated Signals from the Zone’s Operator", RFC 9615, DOI 10.17487/RFC9615, July 2024, <https://www.rfc-editor.org/rfc/rfc9615>.

10.2. Informative References

[LowTTL] paek, P., "DS and DNSKEY low TTL experiments", at DNS OARC 41, 6 September 2023, <https://indico.dns-oarc.net/event/47/contributions/1010/attachments/958/1811/DS%20and%20DNSKEY%20TTL%20experiment.pdf>.

[RFC5730] Hollenbeck, S., "Extensible Provisioning Protocol (EPP)", STD 69, RFC 5730, DOI 10.17487/RFC5730, August 2009, <https://www.rfc-editor.org/rfc/rfc5730>.

[RFC5731] Hollenbeck, S., "Extensible Provisioning Protocol (EPP) Domain Name Mapping", STD 69, RFC 5731, DOI 10.17487/RFC5731, August 2009, <https://www.rfc-editor.org/rfc/rfc5731>.

[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>.

[RFC6840] Weiler, S., Ed. and D. Blacka, Ed., "Clarifications and Implementation Notes for DNS Security (DNSSEC)", RFC 6840, DOI 10.17487/RFC6840, February 2013, <https://www.rfc-editor.org/rfc/rfc6840>.

[RFC9803] Brown, G., "Extensible Provisioning Protocol (EPP) Mapping for DNS Time-to-Live (TTL) Values", RFC 9803, DOI 10.17487/RFC9803, June 2025, <https://www.rfc-editor.org/rfc/rfc9803>.

[SAC126] (SSAC), I. S. and S. A. C., "SAC126: DNSSEC Delegation Signer (DS) Record Automation", 12 August 2024, <https://itp.cdn.icann.org/en/files/security-and-stability-advisory-committee-ssac-reports/sac-126-16-08-2024-en.pdf>.

Appendix A. Terminology

Unless defined in this section, the terminology in this document is as defined in [RFC7344].

Child zone: DNS zone whose delegation is in the Parent zone.

Child (DNS operator): DNS operator responsible for a Child zone.

DNS operator: The entity holding the zone’s primary copy before it is signed. Typically a DNS hosting provider in the domain holder’s name, it controls the authoritative contents and delegations in the zone, and is thus operationally responsible for maintaining the "purposeful" records in the zone file (such as IP address, MX, or CDS/CDNSKEY records). The parties involved in other functions for the zone, like signing and serving, are not relevant for this definition.

Parent zone: DNS zone that holds a delegation for a Child zone.

Parent (DNS operator): The DNS operator responsible for a Parent

zone, and thus involved with the maintenance of the delegation's DNSSEC parameters (in particular, the acceptance of these parameters and the publication of corresponding DS records).

Registrant: The entity responsible for records associated with a particular domain name in a domain name registry (typically under a TLD such as .com or an SLD such as co.uk). When the RRR model is used, the registrant maintains these records through a registrar who acts as an intermediary for both the registry and the registrant.

Registrar: An entity through which registrants register domain names; the registrar performs this service by interacting directly with the registry in charge of the domain's suffix. A registrar may also provide other services such as DNS service or web hosting for the registrant. In some cases, the registry directly offers registration services to the public, that is, the registry may also perform the registrar function.

Registry: The entity that controls the registry database and authoritative DNS service of domain names registered under a particular suffix, for example a top-level domain (TLD). A registry receives requests through an interface (usually EPP [RFC5730]) from registrars to read, add, delete, or modify domain name registrations, and then makes the requested changes in the registry database and associated DNS zone.

RRR Model: The registrant-registrar-registry interaction framework, where registrants interact with a registrar to register and manage domain names, and registrars interact with the domain's registry for the provision and management of domain names on the registrant's behalf. This model is common amongst top-level domains.

Appendix B. Recommendations Overview

TODO Paste all recommendations here

Appendix C. Approaches not pursued

C.1. Validity Checks and Safety Measures

C.1.1. TTLs and Caching

It is not necessary to equally reduce the old DS RRset's TTL before applying a change. If this were done, the rollover itself would have to be delayed without any apparent benefit. With the goal of enabling timely withdrawal of a botched DS RRset, it is not equally important for the previous (functional) DS RRset to be abandoned very quickly. In fact, not reducing the old DS TTL has the advantage of providing some resiliency against a botched DS update, as clients would continue to use the previous DS RRset according to its normal TTL, and the broken RRset could be withdrawn without some of them ever seeing it. Wrong DS RRsets will then only gradually impact clients, minimizing impact overall.

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

* draft-shetho-dnsop-ds-automation-02

  Allow DS automation during registry update lock

  Editorial changes

* draft-shetho-dnsop-ds-automation-01

  Incorporated various review feedback (editorial + adding TODOs)

* draft-shetho-dnsop-ds-automation-00

  Initial public draft.

Authors' Addresses

Steve Sheng
Email: steve.sheng@gmail.com

Peter Thomassen
deSEC
Email: peter@desec.io