SACnnn: Towards an Industry Best Practice for DNSSEC Delegation Signer (DS) Record Automation

A Report from the Security and Stability Advisory Committee

DD Month 2023

Preface

In this report the Security and Stability Advisory Committee (SSAC) considers the issues of automated provisioning of DNSSEC Delegation-Signer (DS) Records.

The SSAC supports the use of DNSSEC and ICANN's call to its full deployment. We observe that there are practical hindrances with the deployment of DNSSEC, and removing these roadblocks would make DNSSEC more accessible. One of those problems is the maintenance of DS records, specifically when the DNS service is not provided by the registrar, but by a third-party service provider.

The SSAC focuses on matters relating to the security and integrity of the Internet's naming and address allocation systems. This includes operational matters (e.g., pertaining to the correct and reliable operation of the root zone publication system), technical administration matters (e.g., pertaining to address allocation and Internet number assignment), and registration matters (e.g., pertaining to registry and registrar services). SSAC engages in ongoing threat assessment and risk analysis of the Internet naming and address allocation services to assess where the principal threats to stability and security lie, and advises the ICANN community accordingly.

The SSAC has no authority to regulate, enforce, or adjudicate. Those functions belong to other parties, and the advice offered here should be evaluated on its merits. SSAC members participate as individuals, not as representatives of their employers or other organizations. SSAC consensus on a document occurs when the listed authors agree on the content and recommendations with no final objections from the remainder of the SSAC, with the exception of any withdrawals included at the end of the document.

Table of Contents

Executive Summary

Although ICANN has been calling for full deployment of domain name system security extensions (DNSSEC),[1] DNSSEC adoption has been hindered by several obstacles. This report addresses one such obstacle: the maintenance of Delegation Signer (DS) records, which connect a domain's DNS data to the chain of trust provided by its parent (e.g., a top-level domain).

Currently, enabling or updating the DNSSEC DS record for a delegated domain involves coordinated actions by the DNS operator, registrant, registrar, and registry. It entails (1) obtaining DNSSEC public key information from the domain's DNS Operator, (2) relaying it to the registry operating the parent zone (possibly via one or more intermediaries), and (3) placing the key information in the parent zone (in the form of DS records).

In the case where the domain's DNS service is operated by the registrar, the process can be simplified as there are fewer actors involved. However, when the domain's DNS service is *not* operated by the registrar, current practice holds the registrant responsible for coordinating the DS provisioning process. The registrant (or someone designated by them) would need to first obtain DNSSEC public key parameters from the DNS Operator. Then the registrant would need to convey these DNSSEC parameters to the registrar (potentially via a Reseller).

The problem with the current approach is that it puts the burden on the registrant to complete the administrative tasks of relaying the DNSSEC parameters. At the same time, the average registrant has neither the proper understanding of DNSSEC in general nor an understanding of their role to convey the DS parameters. Further, handling of DNSKEY and DS records involves technical details that the registrant often does not have knowledge of. Last but not least, the process is idiosyncratic for every combination of DNS Operator and registry/registrar. Correct operation of the process requires a level of engagement and time investment, awareness, and understanding of the process that may not match with what every registrant knows or expects.

This can be alleviated by employing automation for the data exchanges required for DS maintenance so that, when the domain's DNS service is operated by a third party, registries (or registrars) can retrieve all information needed for keeping DS records up to date. By giving registries (or registrars) the ability to automatically retrieve DS records, a higher number of possible workflows can be automated than today, specifically the ones where the DNS operator is not the registrar. DNSSEC deployment at scale can greatly benefit from such automation by removing the need for human initiative and by reducing the opportunity for (human) error. Such technologies have recently been standardized by the IETF, and additional work is underway to advance and mature these technologies.

The SSAC believes that automated DS maintenance should be a required functionality for registries and registrars. Likewise, Child DNS Operators are encouraged to facilitate automatic ingestion of DNSSEC parameters by the parent. To make this a reality, the SSAC makes several recommendations with the goal towards an industry best practice for DNSSEC DS automation.

[1] See "ICANN Calls for Full DNSSEC Deployment, Promotes Community Collaboration to Protect the Internet," Available at: https://www.icann.org/en/announcements/details/icann-calls-for-full-dnssec-deployment-promotes-community-collaboration-to-protect-the-internet-22-2-2019-en

1 Introduction

Although ICANN has been calling for full deployment of domain name system security extensions (DNSSEC),[2] DNSSEC adoption has been hindered by several obstacles. This report addresses one such obstacle: the maintenance of Delegation Signer (DS) records, which connect a domain's DNS data to the chain of trust provided by its parent (e.g., a top-level domain).

This report considers the automation of DS record maintenance for zones which are already signed using DNSSEC and require their DS records to be included (or updated) in the parent's chain of trust.[3] When the domain's DNS service is not provided by the registrar, today's default method of transferring DNSSEC key information to the parent for this purpose involves four steps:

1. the signing party provides the public key information to the registrant,
2. the registrant interacts with the registrar (or a registrar's designee, e.g., a Reseller) and provides the information to the registrar,
3. the registrar validates and forwards the information to the registry,
4. the registry validates and publishes the new DS record set in the TLD zone.

The above description was the industry best practice when DNSSEC was developed, and mechanisms for simplifying or automating these steps were not available. Much has changed since, and automation techniques have been emerging.

This report investigates the options available for performing DS maintenance when the DNS operator is not affiliated with the registry or registrar, and describes ways of automating these updates. In addressing this problem space, the SSAC focuses on domain registrations, such as delegations from a TLD (like .com) or other registry (like .co.uk).

The document is organized as follows: Section 2 provides background information on the role of DS records in the TLD zone; Section 3 analyzes the pros and cons of current and proposed ways to coordinate the transmission of DS record parameters, and discusses ways to automate the DS updates; Section 4 lists operational considerations for relevant parties deploying DS automation technologies with the goal to provide a consistent user experience across TLDs; Section 5 contains findings; Section 6 provides recommendations, which are further divided into high-level (section 6.1) and operational recommendations (section 6.2).

The report contains several appendices. Appendix A defines the terms used in this document that readers may wish to consult for precise definitions. Appendix B elaborates on the operational considerations raised in Section 4, providing background analyses for the corresponding recommendations in Section 6.2. Appendix C illustrates steps a typical registrant has to take to secure DNSSEC delegation, highlighting the need for DS automation. Ultimately the goal of this report is to drive toward a set of industry best practices for DS automation.

[2] See "ICANN Calls for Full DNSSEC Deployment, Promotes Community Collaboration to Protect the Internet," Available at: https://www.icann.org/en/announcements/details/icann-calls-for-full-dnssec-deployment-promotes-community-collaboration-to-protect-the-internet-22-2-2019-en

[3] The report is not concerned with whether, when or how a zone should be signed.

2 Background: The Role of DS Records

To secure a zone with DNSSEC, there are essentially two steps that need to be performed:

1. The child zone needs to be equipped with DNSKEY records that contain the zone's DNSSEC public key(s). The private counterparts of these, not published in the DNS, are used to compute signatures that are published in RRSIG record sets.
2. The parent zone is modified such that it contains an assertion linking at least one of the child zone's DNSSEC public keys to the parent zone. This is achieved by storing DS (delegation signer) records alongside the delegation's NS referral records in the parent. They contain a cryptographic fingerprint (digest) of the child's DNSKEY information and are signed using keys published in the parent zone; the resulting signatures are published as an RRSIG resource record set over the DS resource record set.

It is through the presence of these signatures in the parent zone that the child's DNSSEC public keys are made part of the global DNSSEC chain of trust,[4] enabling DNSSEC-aware resolvers to cryptographically verify the legitimacy of the DNSSEC public keys found in the child zone. Illustrated primers on this topic can be found on the web.[5]

When the child's DNSSEC keys are changed, two steps commonly apply: the child zone needs to be updated with new signatures, and the delegation's public key information needs to be updated by replacing the appropriate DS records in the parent zone.

While Step 1 involves only the child's DNS Operator, Step 2 requires that public key information from the child is somehow conveyed "upwards" for inclusion in the parent zone. This communication process originates at the entity that generates the child zone's DNSSEC keys, and terminates at the parent zone operator, potentially involving one or more intermediaries.

Note that in practice, various levels of sophistication are possible. For example, the entity running the child's nameservers and the entity producing the signatures may not be the same; there may even be several such entities when the registrant has chosen a multi-provider setup for their domain. In order to keep the text readable, this document, including the above description, subsumes these entities under the term "Child DNS Operator" (and similarly, for the parent, under "Parent DNS Operator"). To account for distributed roles, readers may expand the use of these terms as appropriate. Further details on how these and related terms are used in this document are found in Appendix A.

3 Analysis of DS Provisioning Methods

For turning an insecure delegation into a secure one ("DS bootstrapping"), for the management of keys while the delegation is secure (e.g., key rollovers), and for turning a secure delegation

[4] Once a path of trust has been established for one zone, it can extend its trust path to its own children (by including DS records alongside any delegations in the zone). In a typical resolver configuration where a copy of the root DS records is configured as a trust anchor, DNSSEC trust thus unfolds from the top level (root zone) to the lower levels of the DNS hierarchy.

[5] See https://efficientip.com/glossary/what-is-dnssec/ or https://www.cloudflare.com/dns/dnssec/how-dnssec-works/

into an insecure one ("going insecure"), the relevant information needs to be transmitted from the child to the parent.

The communication path used in this process often consists of several parties, with the exact number and their relationships depending on the distribution of technical and business roles.[6] This report considers the case where the various roles are fulfilled by different entities; in particular, the DNS provider is not affiliated with the registry or the registrar.

Conceptually, there are three main ways of coordinating the transmission of DNSSEC parameters from the Child DNS Operator to the parent registry (illustrated in Fig. 1):

1. **Registrant Pull & Push:** The role of the registrant is to coordinate the DS maintenance process. The registrant follows instructions from the domain's DNS Operator to retrieve the necessary technical parameters and forward them in the direction of the Parent DNS Operator (that is, to the registrar or Reseller). For a (typically) non-technical person, this is a non-trivial task.
2. **Parent Pull:** In this model, the parent side (registrar or registry) takes a proactive role and collects the DNSSEC parameters from the Child DNS Operator.
3. **Provider Push:** The DNS Operator pushes DNSSEC parameters upwards. There currently exists no common interface for executing this approach automatically.

These concepts so far have no generally accepted names; the above names are introduced in this report for ease of reference.

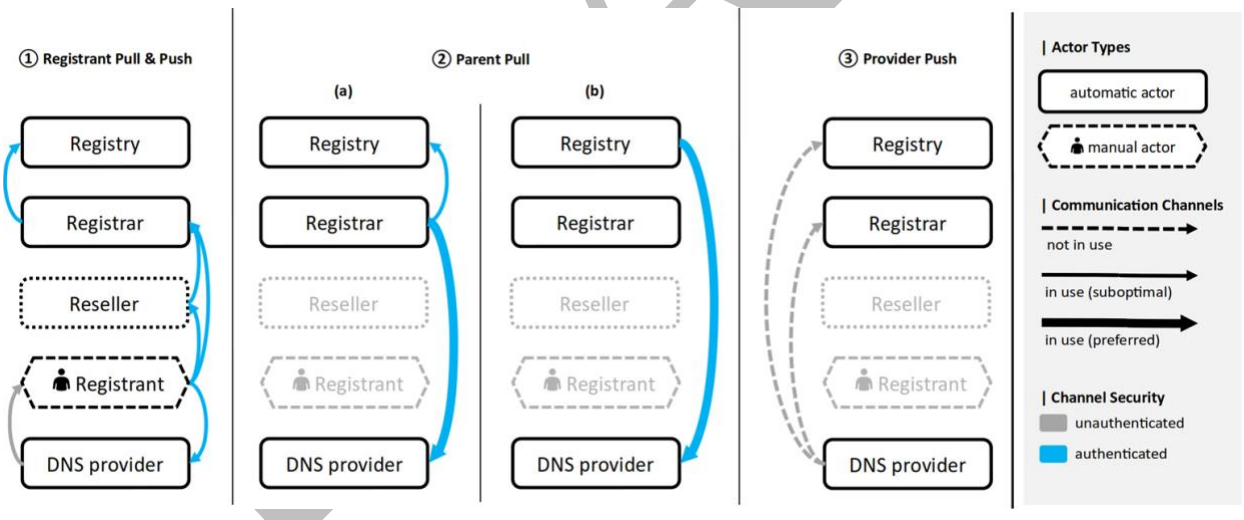

Figure 1. Transmission Channels of various DS Provisioning Methods.

It is worth noting that key changes ("key rollovers") for a given zone are often performed as a regular maintenance task by that zone's DNS Operator. Apart from such routine updates, the DNS Operator may also want to replace the key material due to non-routine circumstances, such as to change the signing algorithm or when key compromise is suspected. In these cases, the

[6] A notable special case is when the signing party and the registrar are closely related. In particular, when they are the same party, the path is "cut short": a signing registrar can convey the DNSSEC parameters directly to the parent. This document, however, considers the situation when this short-cut is not available.

system would ideally allow coordinating the corresponding DS updates in direct interaction with the registrar or registry without human intervention (e.g. of the registrant).

The remainder of this section reviews usability and automation aspects of the three DS provisioning methods. All of these methods assume that the Child zone is already signed upon the registrant's request.

3.1 Registrant Pull & Push

The registrant assumes a coordinating role. Typically, the registrant (or someone designated by them) will (1) retrieve their domain's DNSSEC parameters from the DNS Operator and (2) forward them to the party where the domain was registered (registrar or intermediary Reseller), who then forwards the information on to the registry.

Communication in the first two path segments usually occurs in an authenticated fashion through web forms secured via TLS. Transmission from registrar to registry typically occurs via EPP or a similar protocol, over an authenticated channel (such as TLS). Figure 2 details these steps:

1. DNS Operator equips child zone with DNSSEC;
2. Registrant obtains DNSSEC public key parameters;
3. Registrant conveys parameters directly to the registrar (potentially via a Reseller);
4. Registrar forwards parameters to registry;
5. Registry sets/updates/removes DS records in the parent zone.

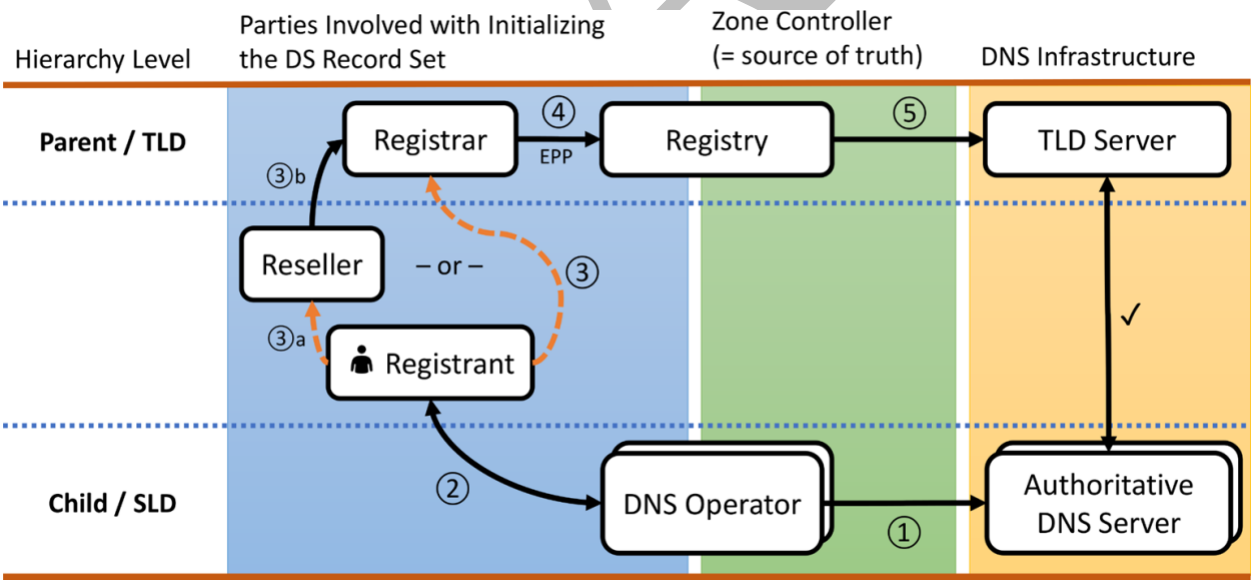

Figure 2. Entities and their relationships during DS provisioning.

While this is the dominant method in use except for registrar or registry provided DNSSEC service, the *Registrant Pull & Push* method has several drawbacks due to the registrant's direct involvement in the operational maintenance of DNSSEC.

First, to perform DS initialization, the registrant would first have to learn about the need to do something, and then perform the actual initialization by pulling and pushing DNSSEC

information from the DNS Operator to the Parent. Put differently: the registrant has to get back into the loop as the clerk in service to the child to do administrative tasks that they can reasonably expect to be driven by their DNS Operator, effectively reversing the relationship.

As a human-driven process, this approach necessarily includes delays and uncertainty. Further, it creates an imposition on registrants' time and attention that not all may be willing to give. The cost of time spent is exacerbated when large domain portfolios need to be maintained, in which case a human-driven approach does not scale well, multiplying the opportunity for error.

Second, once in the loop, registrants often struggle to properly fill the corresponding forms, as several kinds of values have to be entered. User interfaces vary drastically across different DNS Operators and registrars. Some use single-string DS input fields, some use separate fields for DS substructure, and some use drop-down menus for cryptographic algorithms with numerical values or with mnemonics; see Appendix C for examples.

The handling of DNSKEY and DS records involves technical details that the registrant is not expected to have knowledge of. For example, the exact type of information which the registrant needs to retrieve and forward is dependent on the Parent's preferences: Some only accept DNSKEY-style record data (containing the public key) and compute the DS record from it, while others will accept DS-style data directly. As a result, child DNS Operators – unaware of what exactly the Parent expects – often provide both types of data (see the example in Appendix C), leaving the registrant on their own to puzzle out which of these pieces need to go into which form fields, or are at all relevant.

Hence, while the act of copying a string may, in itself, not pose a significant challenge, there is a gap between what the average person would know about DNSSEC and what is necessary to know in order to perform the task. The fact that the process is idiosyncratic for every combination of DNS Operator and registry/registrar requires a level of engagement, awareness, and understanding of the process that may not match with what every registrant knows or expects.

A research paper[7] surveying DNSSEC deployment found that using DNSSEC with third-party DNS operators requires the domain owner to take a number of steps that many domain owners did not successfully complete, leading to a failure rate of 40% of DNSSEC deployment attempts at one DNS operator.

3.2 Parent Pull

The mechanism, in the general case, consists of two components:

1. The DNS Operator makes it known which is the public key information that it wishes to be included in the delegation's DS record set in the parent zone.
2. The registry or registrar fetches this information to update the DS record set accordingly.

[7] See "Understanding the Role of Registrars in DNSSEC Deployment," Available at: https://conferences.sigcomm.org/imc/2017/papers/imc17-final53.pdf

Typically, the first step (child-side publishing) is done using CDS and/or CDNSKEY records,[8] published by the DNS Operator at the apex of the child zone (next to the zone's SOA record).[9] Next, these records need to be fetched by the parent. In the RRR model, this can most naturally be done by the registrar who then forwards the information to the registry, typically via EPP (see Fig. 3). Alternatively, the registry can fetch the information directly from the Child DNS Operator and then update the DS record set; if the domain has a registrar, the registry can inform the registrar about the change (see Figure 4). The task is best performed by only one of these parties, in order to avoid excessive work and potential race conditions.[10]

Retrieval of CDS and CDNSKEY records from the Child DNS Operator is most reliable when authenticated. For DS initialization (first deployment of a DS record set), a specification is under development,[11] while rollovers are authenticated via the child's existing chain of trust.[12]

The benefit of the *Parent Pull* approach is that it avoids the issues of the *Registrant Pull & Push* method by having the registrar or registry proactively pull the required information from the DNS Operator's nameservers. The registrant (as well as any Resellers) are not involved in this picture.

Parent Pull is the only standardized method of DS automation and is currently in use at about 10 ccTLDs and, for reverse DNS zones, at one Regional Internet registry (RIR), RIPE.[13]

The downside of the *Parent Pull* model is that it is not known a priori when CDS/CDNSKEY records will be added, removed, or changed in the child zone. Periodic scanning is used to discover whether a change has occurred.[14] It needs to be done for all domains under management by the registrar or registry: Child zones that already have DS records might have published new records in order to cause DS records to be updated or removed, and newly signed zones without DS records might await DS initialization (bootstrapping).

As noted in section 6.1.1 of RFC 7344, scanning comes with a cost proportional to both its frequency and the number of Child domains to scan. Nevertheless, it is expected to be feasible even for very many delegations when performed daily (on the order of several hundred

[8] CDS records are in the same format as DS records, so can be identically incorporated as such in the parent zone. Alternatively (or additionally), the DNS Operator may publish CDNSKEY records, which are in DNSKEY format and can be used by the Parent DNS Service Operator to compute DS records. The question of whether CDS or CDNSKEY records (or both) should be published is discussed in Appendix B.1.

[9] See RFC 7344 - Automating DNSSEC Delegation Trust Maintenance, available at: https://datatracker.ietf.org/doc/rfc7344/ and RFC 8078 - Managing DS Records from the Parent via CDS/CDNSKEY, available at: https://datatracker.ietf.org/doc/rfc8078/

[10] If the registry and registrar both scan for DS updates, the result will likely be benign since the update will be the same. However, if there is an error or confusing behavior, it may be hard to diagnose and correct the situation if both the registry and the registrar are attempting to update the DS record.

[11] See "IETF Draft - Automatic DNSSEC Bootstrapping using Authenticated Signals from the Zone's Operator," Available at: https://datatracker.ietf.org/doc/draft-ietf-dnsop-dnssec-bootstrapping/

[12] See RFC 7344 - Automating DNSSEC Delegation Trust Maintenance, Available at: https://datatracker.ietf.org/doc/html/rfc7344.

[13] See https://github.com/oskar456/cds-updates

[14] A notable exception is when a registration is new or has its NS record set replaced, in which case the Parent may use the opportunity to look for and process the Child's CDS/CDNSKEY records right away. (This is known as "bootstrapping on inception".)

million)[15]. Still, it is inefficient: Only very few scans are expected to result in an actual DS update, as DNSSEC configurations typically do not change very often.

Further, as the scanning interval is not determined by any standard, it is difficult for DNS operators and registrants to predict after what time changes can be expected to be detected and take effect. Even when the scanning interval is known, the time between publishing and processing a CDS or CDNSKEY record set ranges between "almost immediately" (when the scan happens to occur just after the publication) and the maximum interval between scans (e.g., 1 day).

These downsides can be addressed by having the Child DNS Operator actively inform the parent when DS record maintenance is needed. This approach is described in the next section.

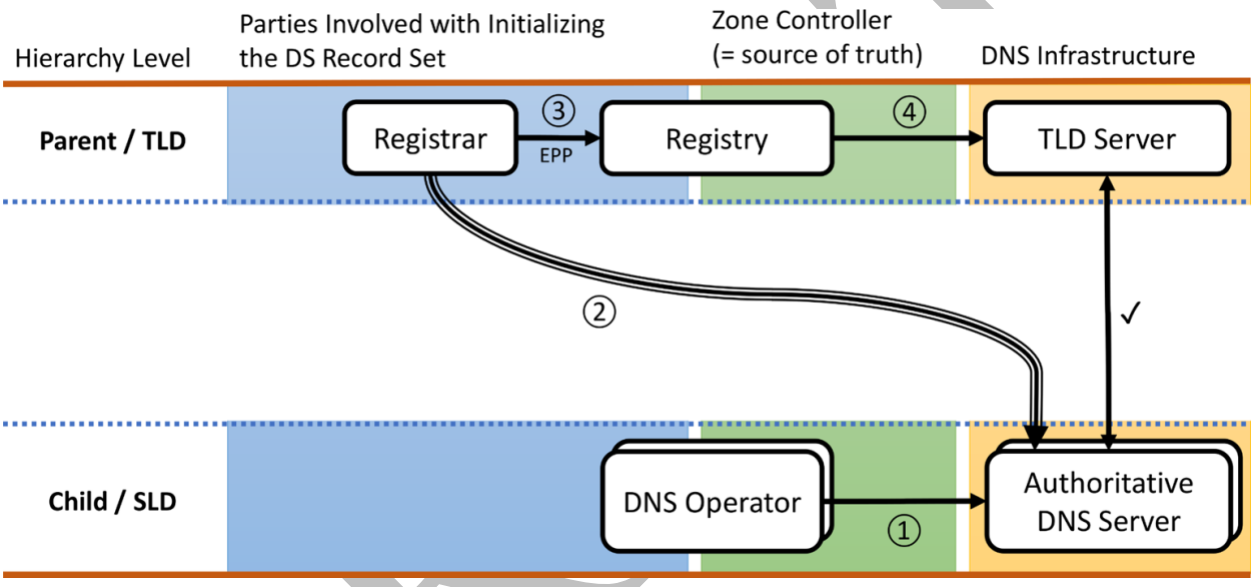

Figure 3. Simplified entity relationships during DS initialization with CDS/CDNSKEY scanning performed by the Registrar. This is the procedure of Recommendation 2a.

[15] An SSAC survey of existing implementations revealed that scans are generally performed daily on a very small number of machines (example: one machine for 20M delegations).

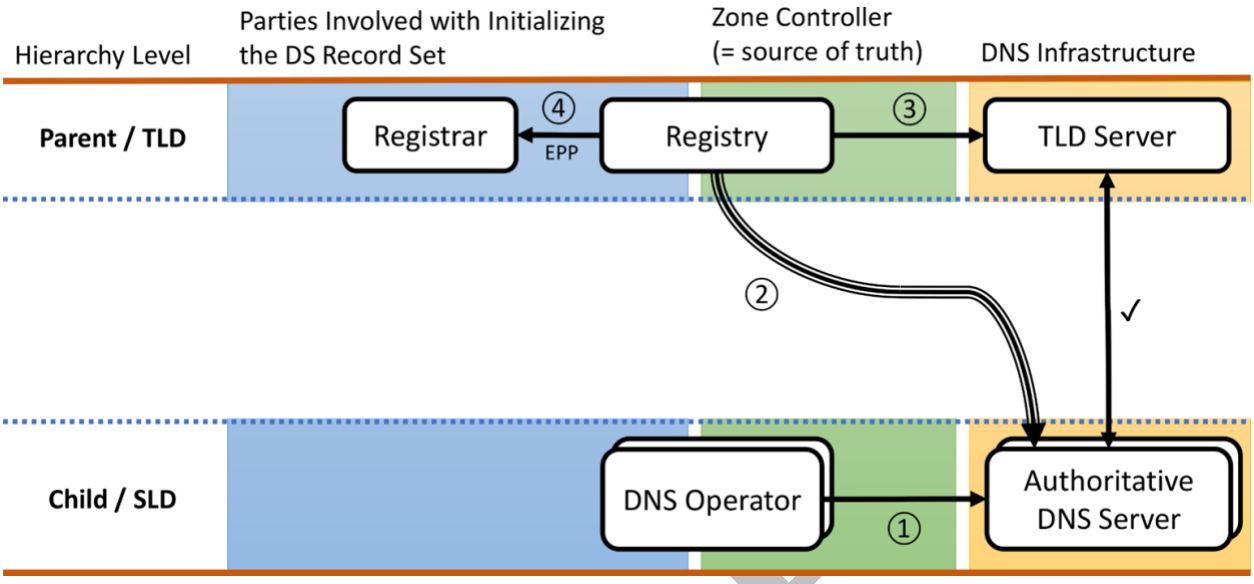

Figure 4. Simplified entity relationships during DS initialization with CDS/CDNSKEY scanning performed by the Registry. This is the procedure of Recommendation 2b.

3.3 Provider Push

In the *Provider Push* method, the DNS Operator pushes DNSSEC parameters, such as DS or DNSKEY records, upwards to the parent. The idea of doing this in an on-demand fashion is conceptually attractive.

The concept comes with certain challenges, such as the treatment of authorization between Child DNS Operators in case the zone is co-hosted by several providers. There currently exists no IETF standards document, nor is there active consideration of how this could be done.[16]

Similar advantages may be achieved by extending the *Parent Pull* method with a notification mechanism that enables the Child DNS Operator to contact the Parent whenever the Child's CDS or CDNSKEY records have changed, so that the Parent can reset the scanning timer for the relevant Child domain. The scan for the domain can then be triggered right away, without waiting for the next regular turn, and DS updates can take effect without unnecessary delay. When implemented universally and reliably, such notifications may further render scheduled scans fully unnecessary, as a scheduled scan could only confirm what has been discovered after a notification.

While there exists no standardized method for such notifications at this time, a proposal based on DNS NOTIFY messages (RFC 1996)[17] has been brought to the IETF DNSOP WG for consideration. Traditionally, NOTIFY messages are used to coordinate replication from primary to secondary authoritative DNS servers. This mechanism is proposed to be generalized to other

[16] In 2015, an effort was undertaken to specify a *Provider Push* REST interface, but the protocol has neither reached final specification nor seen any deployments. https://datatracker.ietf.org/doc/draft-latour-dnsoperator-to-rrr-protocol/

[17] See RFC 1996 - A Mechanism for Prompt Notification of Zone Changes (DNS NOTIFY), available at: https://datatracker.ietf.org/doc/html/rfc1996.

kinds of notifications, in particular for the purpose of DS automation. For details, see Internet-Draft "Generalized DNS Notifications."[18]

While not directly conveying the delegation's DNSSEC parameters, such a notification mechanism would deliver benefits with respect to efficiency and timing that are similar to those in an ideal *Provider Push* setup.

3.4 Discussion

Of the various DS provisioning methods outside of registrar or registry provided DNSSEC service, the *Registrant Pull & Push* method is the most common one. However, it is also least suitable for seamless maintenance of DS records. This is because there is a gap between what the average person would know about DNSSEC and what is necessary to know in order to perform DS maintenance with this method. Furthermore, this human-driven process creates an imposition on registrants' time and attention, and includes delays and uncertainty. As a result of these downsides, about 40% of DNSSEC deployment attempts at one DNS operator surveyed[19] have been found to not be completed.

The *Provider Push* method is envisioned to provide a mechanism for the Child DNS Operator to provision a Child zone's DS records directly with the Parent. However, no IETF standards document for this approach exists at this time.

Instead, registries and registrars who have already implemented DS automation have taken to the *Parent Pull* method, which allows for fully automatic DS provisioning coordination between the Child DNS Operator and a parent-side entity (registrar or registry). While the mechanism is not perfect and suffers from some inefficiency and delays related to the scanning-based approach which it employs, ongoing use of this method has not surfaced any serious operational problems.

The SSAC observes that over the years, DNSSEC operations have shown a general tendency towards automation. For example, key rollovers and regular refreshing of signatures are handled automatically by many nameserver implementations today. The trend towards DS automation is a natural extension of this development. The ultimate goal is to form a best practice that does not involve the need for human intervention. Recently standardized technologies like the *Parent Pull* method support this goal.

When performing DS automation using the *Parent Pull* method, a number of operational aspects needs to be considered. These are elaborated in the next section.

4 Operational Considerations for DS Automation

Automation of DS parameter transmission using standardized protocols seems most practical using the *Parent Pull* approach: If the Child DNS Operator adds CDS or CDNSKEY records to the Child zone, these can be leveraged in this model, resulting in full automation.

[18] See Internet-Draft, Generalized DNS Notifications (draft-thomassen-dnsop-generalized-dns-notify-02), available at: https://datatracker.ietf.org/doc/draft-thomassen-dnsop-generalized-dns-notify/.

[19] See https://conferences.sigcomm.org/imc/2017/papers/imc17-final53.pdf

Enabling support for automatic acceptance of DS parameters directly from the Child DNS Operator requires the registry/registrar to make a number of technical decisions, some of which are listed here.

- Should DS parameters be conveyed via CDS or CDNSKEY records, or both?
- What kind of validity checks should be performed when ingesting DS parameters? Should those checks be performed upon acceptance, or also continually when in place?
- How are conflicts resolved when DS parameters are accepted through multiple channels (e.g. via EPP and via CDS/CDNSKEY)? If both the registry and the registrar are automating DS updates, how to resolve potential collisions?
- What is the relationship with other registration state parameters, such as EPP locks?
- Should a successful or rejected DS update trigger a notification to anyone?
- What is a suitable scanning interval? How can the cost of scanning be reduced?

Not all existing DS automation deployments have made the same choices with respect to these questions, leading to somewhat inconsistent behavior across TLDs.[20] From the perspective of a registrant with domain names under various TLDs, this is unexpected and confusing.

It is thus crucial to address the above operational aspects in a uniform manner across TLDs, so that predictable behavior is obtained. Relevant recommendations are given in Section 6, while a detailed discussion can be found in Appendix B.

5 Findings

Finding 1: While DNS and domain name industry practices regarding signing procedures and other DNSSEC operational tasks have seen significant automation in recent years, progress in the way of automating DS management has been rather limited.

Finding 2: With the addition or removal of DS records being delegation events, registries and registrars obviously play a critical role in DS management. In cases where registries and registrars provide signing services themselves, this role is well established and includes substantial automation of typical scenarios (e.g., key rollovers). It is less well developed when DNS service is provided by a third party, which is not generally supported by parent-side automation.

Finding 3: The domain name industry has evolved to include a new industry player, the Child DNS Operator. This party typically provides both DNS services and DNS security services, but has no dedicated place in the ICANN system to engage in the ICANN policy development or for the registries and registrars to gain an understanding of their concerns.

[20] .ch/.li: https://www.nic.ch/export/shared/.content/files/SWITCH_CDS_Manual_en.pdf;
.cz: https://fred.nic.cz/documentation/html/Concepts/AKM.html;
.fo: https://centralnic.support/hc/en-gb/articles/5957742209309;
.se: https://internetstiftelsen.se/app/uploads/2019/02/se-dnssec-dps-eng.pdf;
RIPE: https://apps.db.ripe.net/docs/Database-Support/Configuring-Reverse-DNS/

Finding 4:

- In the registry-registrar-registrar (RRR) model, when DNS service for the child is provided by the registrar, the key change and subsequent DS update can be administered "internally", such as in direct interaction by the registrar with the registry via EPP.
- Such direct DS updates are also possible outside the RRR model, such as when the registry itself provides DNS service for the child. Examples of this are the suffixes .edu.za or .dedyn.io.
- Outside of these, DS records are typically deployed using the manual deployment method, i.e., *Registrant Pull & Push*. This particularly applies to cases where the RRR model is in use, and DNS service is not provided by the registrar.

Finding 5: The manual method usually involves registrants submitting key information to their registrar, who in turn submits it to the registry. This first part of that process can be onerous and error-prone, and is often perceived as frustrating and difficult.

Finding 6: The need for human intervention around setting up a domain name is not aligned with registrants' expectations, and does not scale sufficiently well when maintaining large domain portfolios.

Finding 7: Non-automatic DS management has emerged as a major obstacle for broad deployment of DNSSEC. For example, studies have shown that 40% of registrants who actively requested zone signing at a large DNS provider did not subsequently configure DS records for their domain. As the number of domains with signing enabled increased by a factor of three during the observation period of the study, the fraction without DS records remained stagnant at about 40%.[21]

Finding 8: The only standardized automated method currently in use for automated DS maintenance is periodic scanning of the child zone for CDS and CDNSKEY records by the parent (RFC 7344, 8078). Over the last years, the method has been gradually deployed at various DNS operators and about ten ccTLD registries, further evolving the DNSSEC ecosystem.

Finding 9: Scanning has two potential defects:

- *Scaling:* Scanning poses load on the system, and both the shorter the frequency of scanning and the greater the total number of delegations[22], the greater the load. However, based on deployment experience, scaling does not appear to be a problem.
- *Delay:* Scanning takes time to detect changes. As the scanning interval is not determined by any standard, it is difficult for DNS operators and registrants to predict after what time changes can be expected to be detected and take effect. This might make scanning a practical method for routine maintenance, but not a good choice for use in emergencies.

[21] See Figure 8 of https://conferences.sigcomm.org/imc/2017/papers/imc17-final53.pdf and related discussion.

[22] Both signed and unsigned delegations need to be scanned, to detect key rollovers and for bootstrapping purposes, respectively.

Finding 10: Sending a notification when the child's key is changed could alleviate both the scaling and delay defects. However, the notification mechanism requires further technical specification as to where to send the notification, and corresponding implementations.

Finding 11: In the RRR model, scans are most naturally done by the registrar. However, deployments exist at ccTLDs and RIRs where the registry takes on that task, and in case both parties attempt DS automation, there is a potential for collision. This collision risk can be mitigated if the registry and registrar agree that only one of them will automate DS updates.

Finding 12: Enabling support for automatic acceptance of DS parameters directly from the Child DNS Operator requires the involved parties to make a number of technical and operational decisions. For example, acceptance criteria have to be considered, both to avoid validation breakage from misconfigured DS records and to resolve conflicts between competing CDS and CDNSKEY submissions. Existing deployments show that these aspects are not show-stoppers; nevertheless, addressing them is crucial for obtaining consistent behavior across TLDs.

6 Recommendations

Automation of DS updates should be a required functionality, especially when the DNS provider is an independent party not affiliated with the registry or registrar. All registries, registrars, and Child DNS Operators should enable DS updates via an interoperable, automated process.

To make this a reality, the SSAC issues both high-level and operational recommendations.

6.1 High-level Recommendations

Recommendation 1: Automated DS maintenance should be facilitated across all domains.

a. ICANN org should create a new program or extend an existing program to advocate for and motivate the deployment of automated DS maintenance as described in this document.

b. ICANN org should strongly encourage registries and registrars to provide support for DNSSEC operations provided by a third party, including automated DS maintenance both for DS initialization (bootstrapping) as well as DS updates and removal, as standardized by the IETF. This should include encouraging appropriate policy development.

c. ICANN org should consider automated DS maintenance to be an enhancement of DNSSEC, and not a new registry service.

d. DNS operators should support DS automation protocols as standardized by the IETF.

e. RIRs are encouraged to provide support for automated DS maintenance of their reverse DNS delegations.

f. Enabling the Child to notify the registry or registrar of a DS maintenance event (e.g. key change) promises benefits with regards to scaling and timing. The community is encouraged to consider the development of a standardized method for this purpose.

Recommendation 2: Automatic DS maintenance (interacting with third-party Child DNS Operators) should be required. This requirement should include involved parties agreeing as to how they will address the operational aspects outlined in section 6.2. The following guidelines appear practical:

a. If the registry wishes to implement automatic DS maintenance, it should be able to do so on the basis of the agreements with the registrars it may have.

b. In the RRR model, if the registrar wishes to do so instead, the registry should defer to the registrar.

Recommendation 3: Each registry's DNSSEC Practice Statement (DPS, RFC 6841) should be updated to include a section on DS Automation, including a clear description of DS input validation. The DPS should also consider the handling of operational edge cases and other aspects listed in the following section.

6.2 Operational Recommendations

Below are a set of operational recommendations for registries, registrars, DNS operators when they implement automation of DS updates. These recommendations are not directed at ICANN, and do not need to be tracked as part of the ICANN advice process.

6.2.1 CDS vs CDNSKEY

For the discussion, see Appendix B.1. The SSAC recommends:

a. Registries should indicate in their DNSSEC Practice Statement (DPS) whether CDS-style or CDNSKEY-style input is accepted.

b. DNS Operators should publish both CDNSKEY records as well as CDS records (as also recommended in Section 5 of RFC 7344,[23] and follow current best practice for the choice of hash digest type (at the time of this writing, RFC 8624[24]).

c. Parents, independently of their choice for CDS or CDNSKEY, are advised to require publication of both kinds of records, and not proceed with updating the DS record set if one is found missing or inconsistent with the other.

6.2.2 Validity Checks and Safety Measures

For the discussion, see Appendix B.2. The SSAC recommends:

a. Entities scanning for CDS/CDNSKEY records should verify that CDS and CDNSKEY responses are consistent across all authoritative nameservers in the delegation, and otherwise cancel the update.

[23] See RFC 7344 - Automating DNSSEC Delegation Trust Maintenance, Available at: https://datatracker.ietf.org/doc/html/rfc7344

[24] See RFC 8624 - Algorithm Implementation Requirements and Usage Guidance for DNSSEC, Available at: https://datatracker.ietf.org/doc/html/rfc8624

b. Entities scanning for CDS/CDNSKEY records should verify that any published CDS and CDNSKEY records are consistent with each other, and otherwise cancel the update.

c. Entities scanning for CDS/CDNSKEY records should verify that the resulting DS record set would not break DNSSEC validation if deployed, and otherwise cancel the update.

d. Registries should reduce a DS record set's TTL to a value between 5–15 minutes when it is updated, and restore the normal TTL value at the occasion of routinely re-signing the record set.

e. When accepting or rejecting a DS update, the Parent should notify relevant parties (such as the Child DNS Operator or registrant) using the methods described in Appendix B.5.

6.2.3 Consistency Issues between Submitting Parties

For the discussion, see Appendix B.3. The SSAC recommends:

a. Registrars and (outside the RRR model) registries should provide a channel for manual DS maintenance in order to enable recovery when the Child has lost access to its signing key(s). It is also needed when a DNS Operator does not support DS automation or refuses to cooperate.

b. When DS updates are received through a manual or EPP interface, they should be executed immediately.

c. Only when the entire DS record set has been removed through a manual or EPP submission should DS automation be suspended in order to prevent accidental re-initialization of the DS record set when the registrant intended to disable DNSSEC.

d. In all other cases where a non-empty DS record set is provisioned manually or via EPP (including after an earlier removal), DS automation should not (or no longer) be suspended. Any CDS/CDNSKEY records present in the Child zone and properly signed should in this case be considered as the current intent of the domain owner.

e. In the RRR model, if the registry performs DS automation, the registry should notify the registrar of all DS updates (see also Appendix B.5).

6.2.4 Registration Locks

For the discussion, see Appendix B.4. The SSAC recommends:

a. Automated DS maintenance should be suspended when a registry lock is set (in particular, EPP lock *serverUpdateProhibited*).

b. To secure ongoing operations, automated DS maintenance should *not* be suspended based on a registrar lock alone (in particular, EPP lock *clientUpdateProhibited*).

c. The wider community of registrars, registries, and standardization bodies like the IETF should consider specifying more clearly under which locks automated DS maintenance is permissible, including by potentially defining a new "maintenance lock" (such as

serverMaintenanceProhibited) whose presence would disable automated DS maintenance.

6.2.5 Reporting

For the discussion, see Appendix B.5. The SSAC recommends:

a. For certain DS updates (see discussion in Appendix B.5) and for DS deactivation, both the domain's technical contact and the registrant should be notified.

b. For error conditions, the domain's technical contact and the DNS operator serving the affected Child zone should be first notified, and only if the problem persists for a prolonged amount of time (e.g., three days), should the registrant be notified.

c. These notifications should be done via email. The same condition should not be reported unnecessarily frequently to the same recipient.

d. In the RRR model, if the registry performs DS automation, the registry should inform the registrar of any DS changes via EPP (RFC 8590) or a similar channel.

e. The history of DS updates should be made accessible to the registrant (or their designated party) through the customer portal available for domain management.

7 Future Work

In this section, we list some potential work for SSAC, as well as the DNS technical and policy community to consider.

- **Interaction of DS maintenance and registration locks.** As explained in Appendix B.4, the impact of registration locks on DS maintenance requires non-trivial analysis. The SSAC's current recommendation is derived from an approach that gives preference to security expectations over operational convenience, yielding a rather practical result (allowing automated DS maintenance during a registrar lock, but not during a registry lock). The community may (or may not) consider a more complete specification of the behavior to be helpful, including a potentially more explicit means for maintenance lock configuration. It seems worthwhile to observe related experiences as deployment of DS automation progresses, and turn them into a guidance document if that seems appropriate at the time.

- **Notifications from the Child DNS Operator to the parent.** With regards to the timing and scaling implications of parent-side scanning, a study or survey could be conducted to determine the degree to which predictable scanning timing is needed, or to quantify the extent to which scanning at larger numbers of delegations or (potentially) at shorter scanning intervals would lead to practical problems. The community might then consider specifying a mechanism that allows the Child DNS Operator to notify the parent in order to trigger a CDS/CDNSKEY scan, in order to address these issues.

- **Authenticated DNSSEC bootstrapping.** Automatic initialization of DS records is supported by several TLDs, using the unauthenticated "Accept after Delay" approach of RFC 8078 (Section 3.3). A better approach would be to use an authenticated mechanism for this purpose, such as the one specified in draft-ietf-dnsop-dnssec-bootstrapping. This protocol already has seen some implementations (such as by .ch/.li on the parent side and by Cloudflare on the child side). The Internet-Draft has been adopted by the IETF DNSOP Working Group, where final steps are needed in order to publish it as an RFC.

- **DNSSEC multi-signer setups.** When several DNS providers sign and serve the same zone (for example for extra redundancy without a single point of failure), it is necessary for the involved providers to coordinate the DNSSEC configuration so that resolution and validation work reliably (RFC 8901). In particular, all involved providers must serve DNSKEY record sets that include all public keys that may be necessary to validate a response from any one of the involved providers. Similarly, each provider's DNSKEY record set must be signed with a key that is represented in the domain's DS record set. Additional consideration needs to be given to the choice of signing algorithms in such multi-signer setups in order to comply with applicable specifications. While this subject area is beginning to be understood, more work is needed for the development of both automated protocols and practical guidance.

- **Glitch-free DNS service during DNS Operator or registrar transfer.** When transferring a signed domain from one DNS provider to another (or from one registrar to another with the DNS service provided by the registrar), a temporary multi-signer setup (see above) is needed in order to ensure glitch-free operation throughout the transition. While a very small number of DNS providers supports such transitions via manual configuration, the required protocols would ideally be supported by all DNS providers and registrars that offer signed DNS service. Once the topic of multi-signer configurations has been addressed (see above), such automated glitch-free provider transfers can be made possible.

- **Building innovative applications on top of the DNSSEC global trust anchor.** In addition to its original purpose of preventing cache poisoning and other forms of DNS attack, DNSSEC was also envisioned as a general purpose authentication platform: it is believed that major benefits can be unlocked from DNSSEC by building applications on top of the global trust anchor it provides. In particular, solutions that provide trust in digital identities of IoT devices or facilitate novel ways for user authentication are conceivable. In the light of this potential, it seems worthwhile to juxtapose these progressive technologies with the costs and incentives that are incurred when deploying DNSSEC, and assess the value DNSSEC provides to operators and users of the Internet.

8 Acknowledgments, Statements of Interest, and Withdrawals

In the interest of transparency, these sections provide the reader with information about aspects of the SSAC process.

The Acknowledgments section lists the SSAC members, outside experts, and ICANN staff who co-authored or contributed directly to this particular document (Contributors) or who provided reviews (Reviewers). The Statements of Interest section points to the biographies of all SSAC members and invited guests, which disclose any interests that might represent a conflict—real, apparent, or potential—with a member's or invited guest's participation in the preparation of this Report. The Withdrawals section identifies individuals who have recused themselves from the discussion of the topic with which this Report is concerned.

Except for members listed in the Withdrawals section, this document has the consensus approval of all of the members of SSAC.

8.1 Acknowledgments

The committee wishes to thank the following SSAC members and invited guests for their time, contributions, and review in producing this report.

SSAC Members

Invited Guests

ICANN Staff
Andrew McConachie
Danielle Rutherford
Kathy Schnitt
Steve Sheng (editor)

8.2 Disclosures of Interest

SSAC member biographical information and Disclosures of Interest at the time of publication are available at:
https://www.icann.org/resources/pages/ssac-biographies-2022-05-02-en

8.3 Withdrawals

Appendix A: Terms Used in the Document

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

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

DS (Delegation Signer) record: A DNS record located at a delegation in the Parent zone and containing the cryptographic fingerprint (hash digest, algorithm) of a DNSSEC public key (DNSKEY) whose private counterpart the Child uses (or intends to use) to sign its DNSKEY record set. The DS record set thus provides validation entry points to the Child zone and is signed by the Parent, thereby connecting the Child's signing key(s) to the DNSSEC chain of trust that the Parent zone already has. There may be zero (DNSSEC off), one, or more DS records for any given delegation.

DNSKEY record: A DNS record containing a public DNSSEC validation key (matching a private DNSSEC signing key at the DNS Operator). The key pair for a given DNSKEY record may be (operationally) constrained to sign/validate the zone's DNSKEY record set only ("key-signing key", KSK), thus authorizing additional keys to sign the rest of the zone's content ("zone-signing keys", ZSK).

DNS (Zone) Operator: The entity controlling the authoritative contents of (and delegations in) the zone file for a given domain name, and thus responsible for maintaining the "purposeful" records in the zone file (such as IP address, MX, or CDS/CDNSKEY records).
The DNSSEC signing function, during whose execution additional records get added (such as RRSIG or NSEC(3) records), may be fulfilled by the same entity, or by one or more signing providers directly or indirectly appointed by the registrant. Zone contents are then made available for query by transferring them to the domain's authoritative nameservers, which similarly may be operated by one or more entities. For the purpose of this definition, the details of how this is arranged are not relevant, as it is the content controlled by the DNS Operator that eventually is served when queries are answered for the given zone.[25]

As a DNS query yields the response specified by the DNS Operator, we thus say that the query is answered by the DNS Operator. Other terms such as "DNS hosting provider", "DNS provider", "DNS service provider" are also often used to describe this concept.

DNSSEC Signer: see DNS Operator.

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

[25] In a DNSSEC multi-signer setup with multiple signing entities, the zone file copies distributed to the various authoritative nameservers may contain varying sets of signatures (RFC 8901: Multi-Signer DNSSEC Models). In all cases, however, there is an entity that, in the name of the registrant, holds the final authority over the zone contents which are subject to the various signing procedures. It is this entity that, by this definition, is the DNS Operator.

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 a SLD such as co.uk). The registrant maintains the records they are responsible for in the registry through the use of a registrar; the registrar 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) 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. In some cases, the registry directly offers registration services to the public, that is, the registry may also perform the registrar function.

Reseller: An entity through which registrants register domain names if they don't interact with the registrar directly. The Reseller is part of a registrar's retail channel and relays the registration request to the registrar. A Reseller may also provide other services such as DNS service or web hosting for the registrant.

RRR Model: Refers to the registrant-registrar-registry interaction framework used for generic top level domains (gTLDs) as well as some country-code top-level domains (ccTLDs). In this model, registrants interact with a registrar to register and manage domain names. Registrars interact with the domain's registry for the provision and management of domain names on the registrant's behalf.

Appendix B: Operational Considerations of DS Automation

This Appendix discusses in detail the operational considerations raised in Section 4. While some of them also arise during conventional DS provisioning, deployment of DS automation across TLDs increases the need for best-practice recommendations towards interoperable and consistent solutions, so that predictable behavior is achieved (from a registrant's perspective).

The recommendations themselves are given in Section 6.2, for which this Appendix provides the background discussion. Each set of Operational Recommendations from Section 6.2 has a corresponding section here that provides the necessary analysis and rationale. For easy reference, each set of recommendations is also copied below.

For some aspects, the SSAC recommends a specific way forward. For other aspects on the boundary to protocol development, the SSAC encourages the community to develop a consensus position.

B.1 CDS versus CDNSKEY

Should DS parameters be conveyed via CDS or CDNSKEY records, or both? If both, how are conflicts resolved?

Analysis:

DS records can be generated from either CDS records or from CDNSKEY records. The former are in a format identical to that of DS records (so their content can be taken verbatim), while the latter are in the same format as DNSKEY records (and generation of a DS record involves computing a hash and other information based on the record content).

Whether CDS or CDNSKEY is ingested by the Parent depends on the Parent's preference:

- Conveying (and storing) parameters in DNSKEY format (such as via CDNSKEY records) allows the Parent to exert control over 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 parameters in DS format (such as via CDS records) allows the Child DNS Operator to control the hash digest type used in computing 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.

Note that the need to make a choice in the face of this dichotomy is not particular 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.

In the context of DS automation, both the Child DNS Operator and the Parent should act as to maximize interoperability. This means that:

- Registries should indicate in their DNSSEC Practice Statement (DPS) whether CDS-style or CDNSKEY-style input is accepted.

- Child DNS Operators are encouraged to publish both CDNSKEY records as well as CDS records, so that they can be ignorant about which format the Parent prefers. The choice of hash digest type should follow current best practice, currently RFC 8624.[26]

  The SSAC would like to emphasize that while publishing the same information in two different formats is not ideal, it seems to be the simpler choice (as opposed to requiring the DNS Operator to discover which Parent prefers which format). In order to avoid operational issues, DNS Operators should take great care to make sure that published records are consistent with each other, for example by programmatically updating them at once.

- Parents, independent of their input format preference, are advised to require publication of both CDS and CDNSKEY records, and to enforce consistency between them. By rejecting the DS update if either type is found missing or inconsistent with the other, Child DNS Operators are held responsible for publishing contradicting information. Registries can retain whatever benefit their choice carries for them, while at the same time facilitating the possibility to later revise their choice without breakage.

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). The SSAC notes 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 that lacks sufficient entropy). Besides, blatantly insecure hash digest types (based on RFC 8624) can still be rejected even when parameters are accepted as DS-style input. (See Appendix B.2.)

The fact that more than one input type is currently specified burdens both the Child DNS Operators and Parents with the need to consider how to handle this dichotomy. While the SSAC points out that this state of things is suboptimal, other community venues such as the IETF are better suited to address the dichotomy and evaluate the possibility of deprecating one of these mechanisms, with Parents transitioning to accepting input of the other type exclusively.

Recommendations:
Based on the above analysis, the SSAC recommends:

a. Registries should indicate in their DNSSEC Practice Statement (DPS) whether CDS-style or CDNSKEY-style input is accepted.

b. DNS Operators should publish both CDNSKEY records as well as CDS records, as also recommended in Section 5 of RFC 7344,[27] and follow best practice for the choice of hash digest type, currently published in RFC 8624.[28]

[26] See RFC 8624 - Algorithm Implementation Requirements and Usage Guidance for DNSSEC, Available at: https://datatracker.ietf.org/doc/html/rfc8624
[27] See RFC 7344 - Automating DNSSEC Delegation Trust Maintenance, Available at: https://datatracker.ietf.org/doc/html/rfc7344
[28] See RFC 8624 - Algorithm Implementation Requirements and Usage Guidance for DNSSEC, Available at: https://datatracker.ietf.org/doc/html/rfc8624

c. Parents, independently of their choice for CDS or CDNSKEY, are advised to require publication of both kinds of records, and not proceed with updating the DS record set if one is found missing or inconsistent with the other.

B.2 Validity Checks and Safety Measures

What kind of validity checks should be performed when ingesting DS parameters? Should those checks be performed upon acceptance, or also continually when in place?

Analysis:

The CDS/CDNSKEY specification (RFC 7344) requires the Parent to verify that the resulting DS record set would not break DNSSEC validation if deployed, and otherwise cancel the update. This is a reasonable strategy to avoid DNSSEC validation breakage; bad DS record sets should not be deployed in the parent zone.

In order to reduce the risk of wrongful DS deployment such as from nameserver hijacking or negligent multi-signer operation, it is important to ensure that CDS and CDNSKEY responses are consistent across nameservers.[29] To that end, Parents should attempt collecting them from all authoritative nameservers listed in the delegation. When a key is referenced in a CDS or CDNSKEY record set returned by one nameserver, but is missing from at least one other nameserver's answer, the Child's intent is unclear, and DS provisioning should not proceed.

Parents are further advised to require publication of both CDS and CDNSKEY records, and not proceed with updating a DS record set if one is found missing or inconsistent with the other. For further CDS vs. CDNSKEY considerations, see Appendix B.1.

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 can be 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. To mitigate this availability risk from permanently lowered TTLs, it may be prudent to limit very short TTLs to the time period shortly after a change to the DS configuration, during which rollbacks are most likely to occur.

One might expect low TTLs to cause increased load on the corresponding authoritative nameservers. However, recent research performed with 5-minute DS TTLs at ISC and a real-world deployment under .goog with (permanent) DS TTLs of 15 minutes have shown[30] such TTLs to have negligible impact on the overall load of a registry's authoritative nameserver infrastructure.

The SSAC thus recommends that registries *reduce* the DS record set's TTL to a low value *when it was updated*, and restore the normal TTL value after a certain time has passed. Pragmatic values for the reduced TTL value range between 5–15 minutes. The TTL reduction should be in

[29] A document specifying this type of consistency check is currently under consideration by the IETF DNSOP Working Group.

[30] Petr Špaček (ISC), Viktor Dukhovni (Google): "March towards shorter DNSSEC outages", DNS OARC 41: https://indico.dns-oarc.net/event/47/contributions/1010/attachments/958/1811/DS%20and%20DNSKEY%20TTL%20experiment.pdf

effect at least until the previous DS record set has expired from caches, that is, the duration of the low-TTL period should exceed the normal TTL value. The routine re-signing of the DS record set (usually after a few days) provides a convenient opportunity for resetting the TTL.

It might look reasonable to also reduce the DS TTL "symmetrically on the other side", that is, well in advance of an upcoming change. However, the important aspect here is to enable timely withdrawal of a botched DS RRset; it is not equally important for a new DS RRset to take effect very quickly. Further, reducing the TTL ahead of time would require the Parent to anticipate when a DS change would occur, but the Parent has no a-priori knowledge of that. The TTL reduction could thus only occur after detecting the CDS/CDNSKEY, but doing so at this point in time would require delaying the DS update by another full TTL (so that the long-TTL DS RRset has expired from all resolver caches). Introducing this additional delay counteracts early use of the new DS RRset and thus does not seem justified. On the other hand, not reducing the DS TTL ahead of time has the advantage of providing some resiliency against a botched DS update, as clients using the previous DS RRset (cached with the normal TTL) would not see the updated DS RRset, and it could be withdrawn without them ever seeing it. Wrong DS RRsets will then only gradually impact clients, minimizing impact overall.

Finally, when accepting or rejecting a DS update, it may be helpful to notify relevant parties (such as the Child DNS Operator or registrant). This reporting mechanism may also be used for notifications in case a problem is detected with an operational DS record set, such as due to an incompatible change in the Child zone. For details, see Appendix B.5.

Recommendations:
Based on the above analysis, the SSAC recommends:

a. Entities scanning for CDS/CDNSKEY records should verify that CDS and CDNSKEY responses are consistent across all authoritative nameservers in the delegation, and otherwise cancel the update.

b. Entities scanning for CDS/CDNSKEY records should verify that any published CDS and CDNSKEY records are consistent with each other, and otherwise cancel the update.

c. Entities scanning for CDS/CDNSKEY records should verify that the resulting DS record set would not break DNSSEC validation if deployed, and otherwise cancel the update.

d. Registries should reduce a DS record set's TTL to a value between 5–15 minutes when it is updated, and restore the normal TTL value at the occasion of routinely re-signing the record set.

e. When accepting or rejecting a DS update, the Parent should notify relevant parties (such as the Child DNS Operator or registrant) using the methods described in Appendix B.5.

B.3 Consistency Issues Between Submitting Parties

How are conflicts resolved when DS parameters are accepted through multiple channels (e.g. via EPP and via CDS/CDNSKEY)? If both the registry and the registrar are automating DS updates, how to resolve potential collisions?

Analysis:

There are multiple channels through which DS parameters can be accepted:

- The registry can scan for CDS/CDNSKEY records and update the DS records;
- The registrar can do the scan and relay the information to the registry;
- Registrars can obtain the information from the registrant via webform submission or other means and relay it to the registry.

The SSAC would like to note several considerations in this context:

- When a manual DS update is performed without adjusting the existing CDS/CDNSKEY records accordingly, the delegation's DS records may be reset to their previous state at the next run of the DS automation process. (Note that due to the validity checks described in Appendix B.2, this can only occur when the existing CDS/CDNSKEY records are compatible with the new DS records, in which case no harm is done.) To address this race condition, some experts have proposed to suspend CDS/CDNSKEY processing once a manual DS update has occurred, until after the Child's SOA serial is found to be updated.

  While appealing at first glance, automatic resumption of DS automation at SOA update time has a high risk of "timing surprises" – namely, when the SOA serial changes for some reason unrelated to the CDS/CDNSKEY records. Any arbitrary modification to the zone content will suffice to trigger DS automation resumption, including the very common updating of DNSSEC signature validity timestamps (often a weekly routine). Furthermore, authoritative nameservers may have different serial offsets (e.g. in multi-provider setups). It is therefore advised to not follow this practice.

  It is further observed that when a zone is equipped with new keys and signatures, followed by a manual deployment of corresponding DS records, the existing CDS/CDNSKEY records – if left in place without modification – will not pass the Parent's acceptance checks as they don't match the new key material. Keeping DS automation active will thus not break the delegation, but can instead help correct the faulty CDS/CDNSKEY configuration via the error reporting mechanism described in Appendix B.5. Once the misconfiguration is fixed, DS automation will return to regular operation without further intervention.

  Automated DS updates should thus generally not be suspended when a manual 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 on the delegation.[31] DS automation should thus be suspended so that automatic re-initialization (bootstrapping) does not occur. In all other cases, any CDS/CDNSKEY records present in the Child zone and properly signed should be considered as the current intent of the domain owner. (The presence of CDS/CDNSKEY records indicates that DS automation is desired.)

- Under special circumstances, it may be necessary to perform a manual DS update. One important example is when the Child zone's private key is destroyed, in which case a CDS/CDNSKEY-based key rollover is impossible because the Child DNS Operator can no longer sign any record sets. Another example is when several providers are involved, but one no longer cooperates (e.g., when removing a provider from a multi-provider

[31] A cleaner way to disable DNSSEC is to configure special CDS/CDNSKEY as described in RFC 8078 Section 4.

setup). Disabling manual DS management interfaces is therefore strongly discouraged.

- When the RRR model is used, there is a potential for collision if both the registry and the registrar are automating DS updates. This collision risk can be mitigated if the registry and registrar agree that only one of them will automate DS updates.

  Note that no adverse 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 zone file from different vantage points, with different CDS/CDNSKEY record contents. This may lead to flapping of DS updates, which is expected to subside as replication eventually becomes consistent. Parents can reduce this effect by checking responses from authoritative nameservers for consistency (see Appendix B.2).

  Further, as a standard aspect of key rollovers (RFC 6781[32]), 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. 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.

Recommendations:
Based on the above analysis, the SSAC recommends:

a. Registrars and (outside the RRR model) registries should provide a channel for manual DS maintenance in order to enable recovery when the Child has lost access to its signing key(s). It is also needed when a DNS Operator does not support DS automation or refuses to cooperate.

b. When DS updates are received through a manual or EPP interface, they should be executed immediately.

c. Only when the entire DS record set has been removed through a manual or EPP submission should DS automation be suspended, in order to prevent accidental re-initialization of the DS record set when the registrant intended to disable DNSSEC.

d. In all other cases where a non-empty DS record set is provisioned manually or via EPP (including after an earlier removal), DS automation should not (or no longer) be suspended. Any CDS/CDNSKEY records present in the Child zone and properly signed should in this case be considered as the current intent of the domain owner.

e. In the RRR model, if the registry performs DS automation, the registry should notify the registrar of all DS updates (see also Appendix B.5).

B.4 Registration Locks

What is the relationship with other registration state parameters, such as EPP locks?

[32] See RFC 6781 - DNSSEC Operational Practices, Version 2, available at: https://www.rfc-editor.org/rfc/rfc6781

Analysis:

Registries and registrars can set various types of locks for domain registrations, usually upon the registrant's request. Some locks clearly should have no impact on DS automation (such as *clientDeleteLock* / *serverDeleteLock*), while for other types of locks, in particular "update locks", the interaction with automated DS maintenance is more interesting.

An overview of the various types of standardized EPP locks and their recommended impact on DS automation is given in Table 1. Some registries may offer additional types of locks whose meaning and set/unset mechanisms are defined according to a proprietary policy.

	Impact on …				
Lock	**Update**	**Delete**	**Transfer**	**Renew**	**Automated DS maintenance**
Registry					
Transfer Lock			prohibited		(allowed)
serverUpdateProhibited	prohibited				**prohibited**
serverDeleteProhibited		prohibited			(allowed)
serverTransferProhibited			prohibited		(allowed)
serverRenewProhibited				prohibited	(allowed)
URS Lock	prohibited	prohibited	prohibited		**prohibited**
Registrar					
clientUpdateProhibited	prohibited				**(allowed)**
clientDeleteProhibited		prohibited			(allowed)
clientTransferProhibited			prohibited		(allowed)
clientRenewProhibited				prohibited	(allowed)

Table 1. SSAC analysis of registry and registrar locks[33] and recommendation on their impact on DS automation. "(allowed)" indicates that this lock in itself should not prevent automated DS maintenance, but more restrictive locks take precedence if present (e.g., serverUpdateProhibited*).*

The only lock types that may be recognized as having an impact on DS automation are the ones prohibiting "Update" operations (see first "Impact" column in the table).

It is the SSAC's position that when a *serverUpdateProhibited* lock ("registry lock") is in place, all updates to the domain's registration data, including any DS updates regardless of how they are submitted, should be disallowed. This is necessary in order to satisfy the expectation that this

[33] For more information on locks, see https://www.icann.org/resources/pages/epp-status-codes-2014-06-16-en.

lock renders all otherwise updateable registration data immutable.[34] (The only exception to this is the registry's out-of-band process for removing this type of lock.)

The situation presents itself differently when a *clientUpdateProhibited* lock ("registrar lock") is in place. While protecting against various types of accidental or malicious change (such as unintended changes through the registrar's customer portal), this lock is much weaker than the registry lock, as its security model does not prevent the registrar's (nor the registry's) actions. This is because the *clientUpdateProhibited* lock can be removed by the registrar without an out-of-band interaction.

Under such a security model, no significant security benefit is gained by preventing automated DS maintenance based on a *clientUpdateProhibited* lock alone, while preventing it would make maintenance needlessly difficult. The SSAC therefore recommends not to suspend automation when such a lock is present. The remainder of this section discusses this in detail.

In a world without 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 DNSSEC 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. Such changes entail updating the delegation's DS records. If authenticated by the Child DNS Operator, these operations do not qualify as accidental or malicious change, but as normal and appropriate activity for securing ongoing operation.

To accommodate key or algorithm rollovers performed by the Child DNS Operator, a means for maintaining DS records is needed. (Making this practical is the subject of this entire report.) It is worth recalling that a DS update request published via CDS/CDNSKEY records on all nameservers constitutes a legitimate request in the name of the registrant, underlined by the fact that it is signed. Further, the resulting DS update is subject to the parent's acceptance checks, and not applied when inconsistent with the DNSSEC keys published in the child zone (see Appendix B.2).

Given that a *clientUpdateProhibited* lock protects against unintended changes (such as through the customer portal) while not preventing actions done by the registrar (or the registry) themself, the lock is not suitable for defending against actions performed illegitimately by the registrar or registry (e.g., due to compromise). In other words, 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; automatic processing of CDS/CDNSKEY records is orthogonal to the attack vector that a registrar lock protects against. Considering that automated DS updates are authenticated and validated for correctness, it thus appears that honoring such requests comes with no additional associated risk, while in the registrant's interest. (Automated DS maintenance may be disabled by requesting a registry lock, if so desired.)

Automated DS maintenance performed by either the registrar or the registry should therefore not be suspended unless the registration data is locked *at the registry level* (e.g. when the registrant has explicitly requested a *serverUpdateProhibited* lock to be set). In particular, a registrar lock alone should not prevent DS maintenance.

[34] See https://www.verisign.com/en_US/channel-resources/domain-registry-products/registry-lock/.

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

The situation might appear less clear for DNSSEC bootstrapping, where automatic DS initialization is not required to maintain ongoing operation. In the absence of a registry lock, however, it is in the interest of security to enable DNSSEC when requested. The fact that a Child zone has been signed and equipped with CDS/CDNSKEY records requesting DS initialization clearly expresses the registrant's intent to have the delegation secured, especially when the request is authenticated as currently under consideration by the IETF[35]. It would hardly be understandable why the registrant would take (or order) these preparatory steps if not for their request to be acted upon.

Further, considering that some domains are put into *clientUpdateProhibited* lock by default, not honoring authenticated DS initialization requests needlessly imposes an additional burden of human intervention for unlocking and relocking the domain in order to facilitate DS provisioning after registration, in spite of the registrant already having expressed their intent of securing their domain with DNSSEC to their Child DNS Operator. The SSAC therefore holds that DS initialization and rollovers should be treated the same way with respect to locks, and only be suspended while in *serverUpdateProhibited* lock status.

An analysis like the above of the security properties of registry and registrar locks with respect to automated DS maintenance is so far not contained in any specification document. The SSAC suggests that this aspect should be specified more clearly, such as by explicitly defining under which locks automated DS maintenance is permissible, or by defining a new "maintenance lock" status whose absence would indicate that automated DS maintenance is permissible.

While this report is concerned with DS management using CDS/CDNSKEY records, the SSAC would like to comment on the related matter of delegation NS record management. NS record updates can be automated using a mechanism similar to DS automation, via so-called CSYNC records (RFC 7477). This technique allows transferring control of a domain's DNS service to another entity by effecting an NS change in the delegation. Its capabilities thus overlap with update as well as transfer actions, both of which can be locked via EPP. As NS automation is out of scope for this report, the SSAC at this time makes no recommendation about CSYNC processing. In particular, the below recommendations should not be construed to endorse automatic processing of CSYNC records in the presence of relevant EPP locks.

Recommendations:
Based on the above analysis, the SSAC recommends:

a. Automated DS maintenance should be suspended when a registry lock is set (in particular, EPP lock *serverUpdateProhibited*).
b. To secure ongoing operations, automated DS maintenance should *not* be suspended based on a registrar lock alone (in particular, EPP lock *clientUpdateProhibited*).
c. The wider community of registrars, registries, and standardization bodies like the IETF should consider specifying more clearly under which locks automated DS maintenance is

[35] See draft-ietf-dnsop-dnssec-bootstrapping.

permissible, including by potentially defining a new "maintenance lock" (such as *serverMaintenanceProhibited*) whose presence would disable automated DS maintenance.

B.5 Reporting

Should a successful or rejected DS update trigger a notification to anyone?

Analysis:

In general, it cannot be assumed that the Child DNS Operator is aware of the error conditions observed by the Parent. For example, they may not know that their CDS/CDNSKEY record contents are not acceptable to the Parent. Early reporting of rejected DS updates may help the Child DNS Operator handle the situation in a timely manner.

Similarly, a delegation can break even when the CDS/CDNSKEY or DS record sets have not changed. Such breakage may occur during key rollovers (RFC 6781) when the Child DNS Operator proceeds to the next step early, without verifying that the delegation's DS record set is in the expected state. For example, when an algorithm rollover is performed and the old signing algorithm is removed from the Child zone while the DS record set still references a key with that algorithm, validation errors will result.

The SSAC therefore suggests that entities scanning for CDS/CDNSKEY records should report on error conditions they encounter. Even when the CDS/CDNSKEY record set has not changed, they should use the occasion and check whether it would be accepted today (see Appendix B.2), and communicate any failures without changing the published DS record set.

The following situations are of particular interest and considered worthy of being reported:

1. The DS record set has been provisioned
    a. manually, or
    b. automatically for initialization (DNSSEC bootstrapping), or
    c. for the first time after a manual change (so DS records are now in sync with CDS/CDNSKEY, see Appendix B.3);

2. The DS record set has been removed
    a. manually, or
    b. automatically via special CDS/CDNSKEY value;

3. A pending DS update cannot be applied due to an error condition. There are various scenarios where authenticated CDS/CDNSKEY records are available, but the associated DS update can't be fulfilled. These include:
    a. Both CDS and CDNSKEY records are available, but contradict each other (see Appendix B.1);
    b. The new DS record set would break validation/resolution or is not acceptable to the Parent for some other reason (see Appendix B.2);
    c. Some kind of lock prevents DS automation (see Appendix B.4);

4. No DS update is due, but in determining this it was found that the Child zone is no longer compatible with the existing DS record set (e.g. lack of DNSKEY algorithm).

For these reportworthy cases, the entity performing DS automation should attempt to communicate the situation. Potential recipients are:

- Registrant (domain holder), via email;
- Domain's technical contact, via email;
- Registrar (if DS automation is performed by the registry), via EPP (or similar channel);
- DNS operators, via email (from SOA contact data).

For DS updates and deactivation (cases 1 and 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.

For error conditions (cases 3 and 4), the registrant need not always be involved. It seems worthwhile 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 changes via EPP (RFC 8590[36]) or a similar channel.

During regular maintenance of a secure delegation, standard updates of DS record sets are not of particular interest. In particular, if the delegation is already secure and a DS update has been applied automatically without any indication of irregularity (authenticated and valid CDS/CDNSKEY records), email notifications need not be triggered.

Further, 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 made accessible to the registrant (or their designated party) through the customer portal available for domain management.

Recommendations:
Based on the above analysis, the SSAC recommends:

a. For certain DS updates (see discussion in Appendix B.5) and for DS deactivation, both the domain's technical contact and the registrant should be notified.

b. For error conditions, the domain's technical contact and the DNS operator serving the affected Child zone should be first notified, and only if the problem persists for a prolonged amount of time (e.g., three days), the registrant should be notified.

c. These notifications should be done via email. The same condition should not be reported unnecessarily frequently to the same recipient.

[36] See RFC 8590 - Change Poll Extension for the Extensible Provisioning Protocol (EPP), Available https://www.rfc-editor.org/rfc/rfc8590.html

d. In the RRR model, if the registry performs DS automation, the registry should inform the registrar of any DS changes via EPP (RFC 8590) or a similar channel.

e. The history of DS updates should be made accessible to the registrant (or their designated party) through the customer portal available for domain management.

Appendix C: Steps Registrants Have to Do to Secure DNSSEC Delegation

In this Appendix, we list the steps a registrant must take to secure a delegation. For illustration purposes, we take it for granted that the registrant's Child DNS Operator already signed the zone, and the registrant's registrar has an interface/ability to accept DNSSEC information.

Step 1: The registrant needs to obtain the DNSSEC DS information from the DNS Operator. In order to do that, the registrant needs to (1) be able to locate the appropriate menu/interface/webpage on the DNS operator's website, and (2) be able to copy these key information down. Figure X below shows how the registrant can retrieve such information from one DNS Operator's interface.

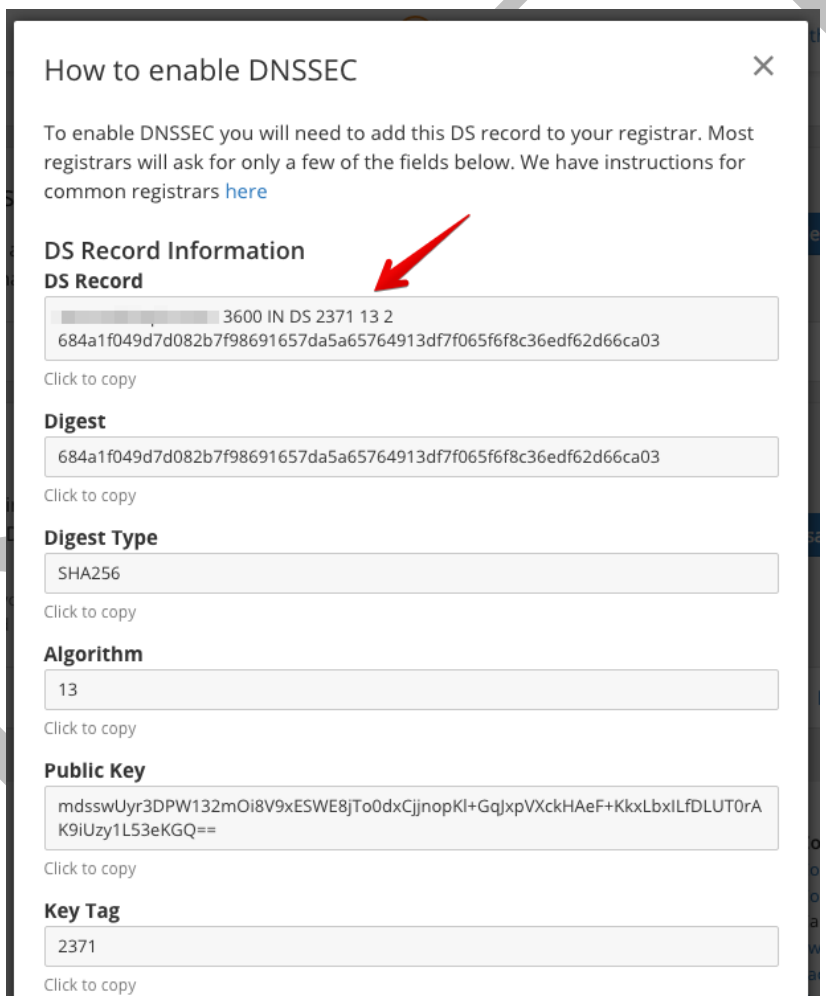

Figure 5. A DNS Operator's Interface for registrant to Retrieve DS Record Information

Challenges to registrants for completing Step 1:

- The registrant first needs to understand that s:he needs to complete these steps, otherwise the domain would not be secured. This may or may not align with the registrant's expectations. Furthermore, the registrant who is not familiar with DNSSEC does not know what a DS record is, and how it differs from other records (such as DNSKEY).
- Even if the registrant knows what the DS record is, the registrant may not be familiar with their DNS Operator's interface to properly locate the DS records.
- The screenshot displays six (6) different values, namely:

    a. *DS Record*
    b. *Digest:* An alpha-numeric (hexadecimal) string representing a hash digest over the DNSKEY record which this DS record pertains to. The length of the string depends on the *Digest Type* chosen, and is 64 characters for *Digest Type* 2 and 96 characters for Digest Type 4.
    c. *Digest Type:* The hash algorithm type used to construct the DS digest field. Values are numeric, but mnemonics are common. For example, this screenshot displays "SHA256" which corresponds to value 2.
    d. *Algorithm:* The cryptographic signing algorithm of the associated DNSKEY. Values are numeric, but mnemonics are common. Although this screenshot has a mnemonic for the *Digest Type*, it uses the value 13 here (the mnemonic would be "ECDSAP256SHA256")[37].
    e. *Public Key:* Public key of the associated DNSKEY record (in base64 encoding).
    f. *Key Tag:* Contains the tag value of the DNSKEY record referenced by this DS record.

    The registrant may assume the DS record is one long string ("presentation format"), which is one valid perspective and given as the first field (a). That the fields (b)–(d) and (f) contain the same information, however, is neither explained nor cannot be inferred by inspection due to the inconsistent use of mnemonics.

    In addition, the second-to-last entry (e) contains the *Public Key* itself, which is not part of the DS record. Instead, it is a component of the DNSKEY record, which is determined by the *Public Key* and *Algorithm* fields (the latter of which is also part of the DS record).

    To summarize, it is rather unclear which of these fields the registrant may or may not need in order to assemble all the necessary information.

Step 2: The registrant needs to put the DNSSEC information obtained from the DNS Operator into the registrant's registrar interface. To do that, the registrant needs to (1) be able to locate the appropriate menu/interface/webpage on the registrar's website, and (2) be able to input the DS parameters in the correct location in the registrar's interface. Figures below show some registrars' interfaces.

[37] See https://www.iana.org/assignments/dns-sec-alg-numbers/dns-sec-alg-numbers.xhtml

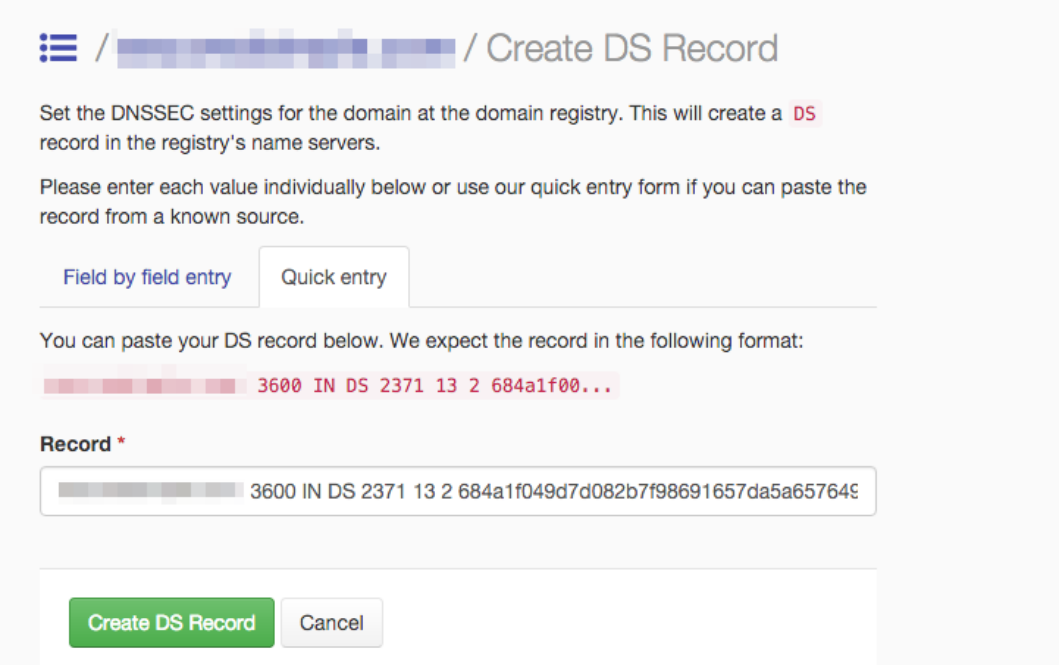

Figure 6. A registrar's interface for registrants to create DS Record. This interface requires direct input of the full DS record in presentation format.

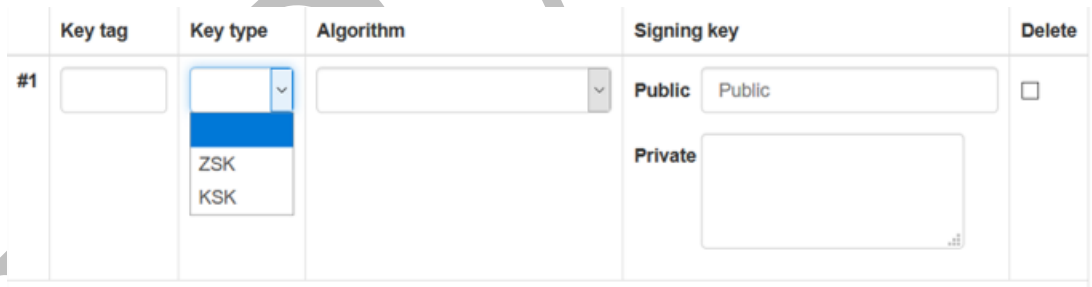

Figure 7. A registrar's interface for registrants to create DS Record. This interface requires entering the various components of the DNSKEY record, namely the Algorithm and the Public Key (labeled as "Signing Key – Public"). The presence of "Key type" is unnecessary (the value always being 257/KSK). The "Key tag" field is misplaced, as it is not part of the DNSKEY record – it belongs to the DS record, other components of which are not displayed (such as Digest Type). Finally, the "Signing Key – Private" is confusing and potentially dangerous, as there is no need to input any private key material for DS provisioning.

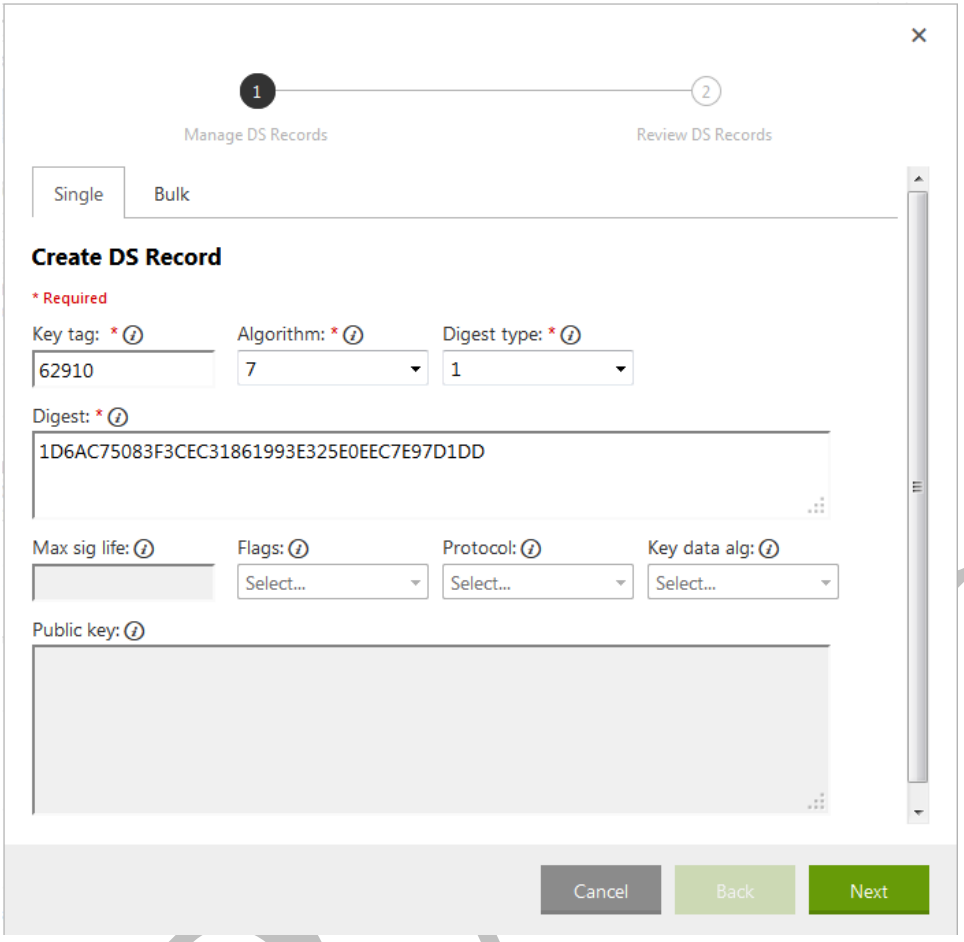

Figure 8. Another registrar's interface for registrants to create DS Record. It requires separate input of the various DS record components. Note that the numerical value of the Digest Type can be selected from a drop-down field (current selection: 1), whereas the Child DNS Operator's interface used a mnemonic ("SHA256") for this field. The registrant would have to know that the correct choice in this case is 2. – The input fields in the lower half of the screenshot are all irrelevant.

Challenges to the registrant for completing Step 2:

- The registrant would have to understand the processes/interfaces at the registrar. Our example focuses on creating DS records only. Other cases include modification and deletion of DS records. As illustrated in the sample interfaces above, these can be quite complex for a registrant to navigate, even if the registrant understands the need.
- Some registrars allow the registrant to paste the whole DS Record (see Figure 6). Other registrars, in various flavors, require the registrant to copy parameter by parameter and/or selection values from drop down menus (see Figures 7 and 8). Still others offer additional fields in which nothing needs to be entered (see Figure 8). It may be difficult to determine that these fields can be left empty.