SAC126
DNSSEC Delegation Signer (DS) Record Automation

A Report from the ICANN Security and Stability Advisory Committee (SSAC)
12 August 2024

Preface

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

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

The deployment of Domain Name System (DNS) Security Extensions (DNSSEC) has been hindered by a number of obstacles. This report focuses on one: the management of Delegation Signer (DS) records, which connect a child zone's DNSSEC public key and signatures to the chain of trust provided by its parent zone (e.g., a zone corresponding to a top-level domain).

DNSSEC is not simply enabled by signing a delegated domain's DNS zone with DNSSEC signatures. It is also necessary to configure (and later maintain) appropriate DS records, which involves coordinated actions by the DNS operator, registrant, registrar, and registry.

In the case where the domain's DNS service is operated by the registrar, this process can be reduced to a simple internal operation by the registrar. If the functions are separated, this is not possible. This report is therefore focused on when the domain's DNS service is *not* operated by the registrar, but by a third-party DNS operator.

In such a scenario, current practice holds the registrant responsible for coordinating DS maintenance. The registrant (or someone appointed by them) needs to first obtain DNSSEC public key parameters from the DNS operator, and convey these parameters to the registrar (potentially via a reseller). The registrar will then need to relay these DNSSEC public key parameters to the registry, who will use them to create and publish the DS record in the parent zone. This process often involves idiosyncratic interfaces for each combination of DNS operator and registrar, requiring a level of engagement and time investment, awareness, and understanding that often do not match with what the registrant knows or expects. The complexity of the process further introduces opportunity for error.

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, without human involvement, obtain all information needed for keeping DS records up to date. Various approaches to achieve this are possible, such as a scheme where the registry or registrar actively contacts the Child DNS operator, or vice versa. The different approaches come with different challenges with respect to authentication, timing, and efficiency.

The IETF has standardized specifications around the first approach, where the parent pulls information from the Child DNS operator, and operational experience has been gained over recent years. However, some standardization gaps remain (such as to improve efficiency and error handling). In addition, the industry could benefit from further development of best practices in deploying the technology.

The SSAC believes that automated DS maintenance should be a goal for the domain name industry. To make this a reality, the SSAC makes several recommendations with the goal to spur industry players and ICANN towards an industry best practice for DNSSEC DS automation.

1 Introduction

Since signing the root zone in 2010, ICANN has been calling for full deployment of domain name system security extensions (DNSSEC).[1,2,3,4] However, DNSSEC adoption has been hindered by a number of obstacles. One such obstacle is the management of Delegation Signer (DS) records. These records connect a child zone's DNSSEC public keys and signatures to the chain of trust provided by its parent zone (e.g., a zone corresponding to a top-level domain), and thus need to be provisioned and maintained properly.

This report considers the automation of this process, focusing on DNS zones that are signed with DNSSEC and require DS records to be included in their parent zone.[5] Today's default method of transferring DNSSEC key information to the parent for this purpose involves up to 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 performs local checks and forwards the information to the registry, and
4. the registry performs local checks and publishes the new DS record set in the TLD zone.

When the domain's DNS service (including signing) and the domain's sponsoring registrar function are operated by the same supplier, DS provisioning can be automated by combining steps 1, 2 and 3 above, since the supplier can coordinate the DNS function and the Registrar function seamlessly. The challenge occurs when the domain's DNS service and the sponsoring registrar function are provided by different, uncoordinated suppliers. In this case, all of the four steps above are needed, and they are performed separately by the suppliers providing the corresponding functions of DNSSEC signing, registrant, registrar and registry.

This report investigates the options available for performing DS provisioning when the DNS operator, registrar and registry functions are provided by different suppliers without opportunity for their services to be integrated or packaged. The goal of this report is to help the domain name industry develop and use best practices for automated management of DS records.

The document is organized as follows: Section 2 provides background for understanding the role of DS records; Section 3 gives a more detailed problem statement; Section 4 analyzes the pros

[1] "ICANN Calls for Full DNSSEC Deployment, Promotes Community Collaboration to Protect the Internet," *ICANN*, February 22, 2019, https://www.icann.org/en/announcements/details/icann-calls-for-full-dnssec-deployment-promotes-community-collaboration-to-protect-the-internet-22-2-2019-en.
[2] Yazid Akanho and Paul Muchene, "OCTO-029 - DNSSEC Deployment Guidebook for ccTLDs," *ICANN*, November 12, 2021, https://www.icann.org/en/system/files/files/octo-029-12nov21-en.pdf.
[3] E.g., Specification 6 of "Registry Agreement gTLD," *ICANN*, February 26, 2015, https://itp.cdn.icann.org/en/files/registry-agreements/aaa/aaa-agmt-html-26feb15-en.htm.
[4] "ICANN to Work with United States Government and VeriSign on Interim Solution to Core Internet Security Issue," *ICANN* June 3, 2009, https://itp.cdn.icann.org/en/files/announcements/release-2-03jun09-en.pdf.
[5] The focus of this report is on the issue of DS provisioning and is not concerned with when or how zones should be signed.

and cons of current and proposed ways to coordinate the transmission of DS record parameters; Section 5 summarizes the findings; Section 6 provides recommendations.

The report contains three appendices. Appendix A defines the terms used in this document. Most notably, the exact arrangement of the entities managing a domain's DNS zone is ignored, and subsumed under the term "DNS Operator". This may encompass several operators, and also includes the signing party. Appendix B discusses operational considerations, and Appendix C illustrates the challenges faced when performing DS management without automation.

2 Background: DS Records

This section explains, from a conceptual perspective, how DS records come into play when provisioning DNSSEC for a DNS zone, and how they are affected when the DNSSEC configuration of a zone changes. It also lays out which actors are involved at each step of the process.

2.1 Initial Provisioning (DNSSEC Bootstrapping)

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

1. The child zone's operator signs its authoritative resource records sets (RRsets) using one or more private keys and publishes the signatures and corresponding public keys in DNSKEY records in the zone itself.
2. The child zone's operator identifies a suitable trust anchor for the child zone[6] and arranges for it to be published in the form of a Delegation Signer (DS) RRset in the parent zone.

It is through the presence of these DS records (and their accompanying signatures) in the parent zone that the child's DNSSEC public keys are made part of the global DNSSEC chain of trust,[7] enabling DNSSEC-aware resolvers to cryptographically verify the legitimacy of the DNSSEC signatures found in the child zone. Several illustrated primers on this topic are available.[8]

2.2 Key Changes

There may be situations, either routine or emergency, in which an administrator might want to add or remove keys used for signing a zone. A non-exhaustive list of potential reasons are:

- Adding or removing a signing algorithm (or both in sequence, for changing algorithms);
- Adding or removing a key from another provider (multi-signer, RFC 8901);

[6] The key used to sign authoritative records and the one used as an entry point key may be the same ("Combined Signing Key", CSK); alternatively, two separate keys may be used, with entry point key called the Key Signing Key (KSK) and the key signing the remaining records of the zone called the Zone-Signing Key (ZSK).
[7] 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 root to the lower levels.
[8] "What is DNSSEC?," *Efficient IP*, https://efficientip.com/glossary/what-is-dnssec/ and "How DNSSEC Works," *Cloudflare*, https://www.cloudflare.com/dns/dnssec/how-dnssec-works/.

- Adding or removing a stand-by key for later use (see RFC 6781 Section 4.4);
- As a precaution (e.g., after potentially insecure disposal of key storage hardware);
- Turning off DNSSEC ("going insecure", e.g., for switching DNS operators that don't support signed transfers);
- The old key can no longer be used for signing.

Whenever the DNSSEC keys associated with the secure delegation change, adjustments have to be made in both the child and the parent zone. Conceptually, this involves two steps much alike the ones for initial provisioning:

1. The child zone needs to be updated with new signatures made with the new (set of) keys;
2. In the parent zone, the delegation's public key information needs to be updated by replacing the relevant DS RRset with one that represents the child's new public key(s).

These steps ensure that the parent's delegation is consistently linked with the public keys used by the child zone. Child- and parent-side updates need to be carefully coordinated so that a valid chain of trust remains available at all times.[9]

2.3 Roles and Responsibilities

While signing a zone (Step 1 in the above subsections) involves only the child's DNS operator, Step 2 requires that public key information from the child be 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.

In practice, several variations 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 this text readable, this document 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 terms as appropriate. Further details on how these and related terms are used in this document are found in Appendix A.

3 Problem Statement

The previous section has outlined that the operation of a DNSSEC-secured zone in the global DNS requires the provisioning of DS records in the parent zone, linking the parent's chain of trust with the child's DNSSEC validation public keys.

Automation of this process is mostly absent in cases where the DNS service and the sponsoring registrar function for the domain are provided by different, uncoordinated suppliers. As a result, a human (typically, the registrant) has to get involved and relay the relevant parameters. This

[9] Detailed recipes can be found in Section 4 of "RFC 6781 - DNSSEC Operational Practices, Version 2," *IETF*, https://datatracker.ietf.org/doc/rfc6781/.

registrant-centric process has turned out to be a significant obstacle for the deployment of DNSSEC. For a detailed analysis of these difficulties, see Section 4.1.

The obstacle can be removed by automating DS record provisioning, which eliminates the need for human intervention. This requires answering the following questions:

1. **Authentication:** How does the Child DNS operator establish trust with the Registrar/Registry, enabling them to verify the legitimacy of the desired change?
2. **Trigger:** At what points in time does the parent apply DS record changes? Can the Child DNS operator request this on demand?
3. **Transport channel:** Which is the method used to convey DS provisioning requests?

Ideally, the changes to a DS record set should occur in a smooth, efficient, and faultless fashion. To determine the requirements for such a process, it is worth splitting the problem into its two use cases. For an overview of the dimensions of the problem space, see Table 1.

3.1 Initial DS Provisioning (DNSSEC Bootstrapping)

Once signatures and corresponding keys have been added to the child zone, it is ready to have its delegation secured. This is achieved by the addition of a DS record set alongside the delegation NS records in the parent zone (Step 2 in Section 2). At this point, some process needs to be initiated which will add the desired DS record set[10] to the parent zone.

The first question is how to automatically initiate this process. Considering *who should trigger the process*, there are two primary options and a hybrid variant:

a) The Child DNS operator could trigger the process by contacting some endpoint.
b) The parent registry or registrar could make unsolicited checks to see if the child zone is ready for DS provisioning. For example, when a new child is about to be delegated, the parent may check whether the child desires DS provisioning; if so, the bootstrapping procedure can be executed immediately, securing the delegation from its inception and avoiding any insecure time windows. Similar checks may be done at later points in time.
c) Combining (a) and (b), the Child DNS operator could send a notification to the parent, requesting a parent-side check to happen on demand.

In case (a), the endpoint location would have to be well-known or at least easily discoverable. One complication is that potentially there might be several such endpoints under a registry (e.g., one per registrar); another one is authentication. Option (b) does not require locating an endpoint; however, unsolicited checks come with uncertainties about timing and scalability. An interesting property of option (c) is that the notification does not need to be authenticated, as the parent can verify legitimacy during its subsequent check.

Once the provisioning process has been started, the second question is *how exactly to convey the required information*. As the child zone is lacking a chain of trust, it does not suffice to simply

[10] In general, several different DS records can be used to link the child's validation keys with the parent. For example, a child might have multiple entry point keys and publish DS records for any number of them. Additional factors like the choice of digest type result in different DS records, and may be up to the Parent operator (and not the requestor).

retrieve the child's DNSSEC parameters via DNS queries, because the responses (although signed) cannot be validated, and hence are at risk of manipulation while in transit. Additional authentication is therefore necessary, either in-band (e.g., via the Authenticated DNSSEC Bootstrapping protocol[11]) or using an out-of-band method.

3.2 DS Updates

Unlike in the case of initial provisioning, the child domain already is securely delegated when a change of keys necessitates a DS update. One option for authenticating such DS change requests (addition or removal of DS records) is thus to make use of the child's existing chain of trust, by transmitting the new parameters in a signed DNS message that can be validated by the recipient.

However, there might be other possibilities: At least in principle, any authentication mechanism suitable for DNSSEC bootstrapping might be suitable for authenticating subsequent DS updates.

Regarding the mechanism used to trigger the DS update process, the same basic considerations apply as for initial DS provisioning: The Child DNS Operator (or some other party might) contact an endpoint to initiate the update process, the parent registry or registrar could look for updates themselves, or a polling trigger could be used.

	Initial DS Provisioning	**DS Updates**
Trigger	• Parent-side notification endpoint (push) • Opportunistic readiness checks by parent registry or registrar (pull) • Polling trigger, combining a push notification with the pull process	
Authentication	• In-band (RFC 9615) • Out-of-band	• In-band (RFC 7344) • Out-of-band
Transport channel	• DNS • HTTPS, with an API provided either by the parent or child operator	

Table 1. Basic dimensions and high-level considerations of the DS provisioning problem.

4 Approaches to DS Provisioning

In general, the communication path used in DS provisioning involves several parties, with the exact number and their relationships depending on the distribution of technical and business roles (see Section 2.3).[12] This report considers the case where, in particular, the Child DNS Operator is not affiliated with the parent registry or the registrar.

Conceptually, the required communication can either be mediated by a human who is in touch with both the Child DNS Operator and the parent registry or registrar (**registrant-centric approach**, Section 4.1). Alternatively, the communication can occur without human intervention

[11] "RFC 9615 - Automatic DNSSEC Bootstrapping using Authenticated Signals from the Zone's Operator," *IETF*, July 18, 2024, https://datatracker.ietf.org/doc/rfc9615/.

[12] A notable special case is when the signing party and the registrar are the same party. In this case, 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.

(Section 4.2), in which case the parent registry or registrar can collect pending changes from the child domains it is responsible for (**pull-based approach**), or the child side pushes the change to the parent (**push-based approach**). Both of these approaches are plausible for removing human intervention from the process, but each one comes with its own set of shortcomings. Figure 1 gives an illustration of the various approaches. A third, hybrid automation approach is for the Child DNS operator to contact the registry or registrar to trigger a pull process on demand.

The following subsections review usability and automation aspects of these approaches. All of these methods assume that the Child zone has already been signed.

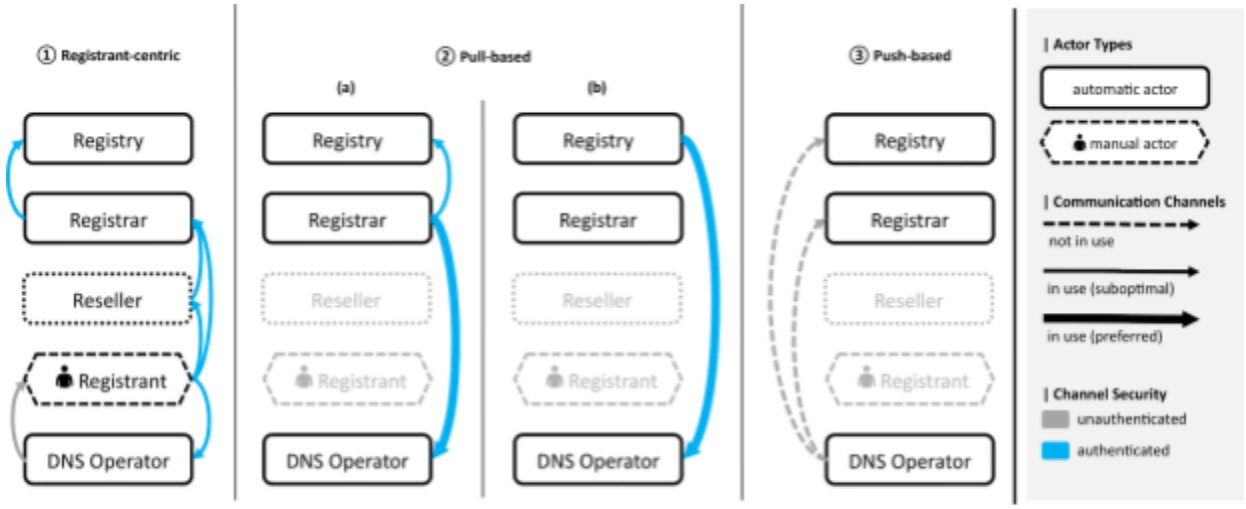

Figure 1. Transmission Channels of various DS Provisioning Methods.

4.1 Registrant-centric

In this approach, the registrant assumes a coordinating role. Typically, the registrant (or someone designated by them) will (1) follow instructions from the domain's DNS operator to retrieve the necessary DNSSEC parameters and (2) forward them to the party where the domain was registered (registrar or intermediary reseller), (3) who then forwards them on to the registry.

Communication in the first two path segments usually occurs in an authenticated fashion through web forms secured via the Transport Layer Security (TLS)[13] protocol. Transmission from registrar to registry typically occurs via Extensible Provisioning Protocol (EPP)[14] or a similar protocol, over an authenticated channel (such as TLS). Figure 2 details these steps:

1. DNS operator signs the child zone with DNSSEC;
2. Registrant obtains DNSSEC public key parameters;
3. Registrant conveys parameters to the registrar (potentially via a reseller);
4. Registrar forwards parameters to registry;

[13] "RFC 5246 - The Transport Layer Security (TLS) Protocol," *IETF*, August 2008, https://datatracker.ietf.org/doc/html/rfc5246.
[14] "RFC 5730 - Extensible Provisioning Protocol," *IETF*, August 2009, https://datatracker.ietf.org/doc/html/rfc5730.

5. Registry sets/updates/removes DS records in the parent zone.

This is the dominant method in use for DS provisioning (unless the registrar or registry also provides DNS service for the child).

This method allows the human operator to see that each step is performed (i.e., whether the DS information was conveyed), and whether any errors occurred. On the other hand, the registrant's direct involvement in the operational maintenance of DNSSEC leads to several drawbacks.

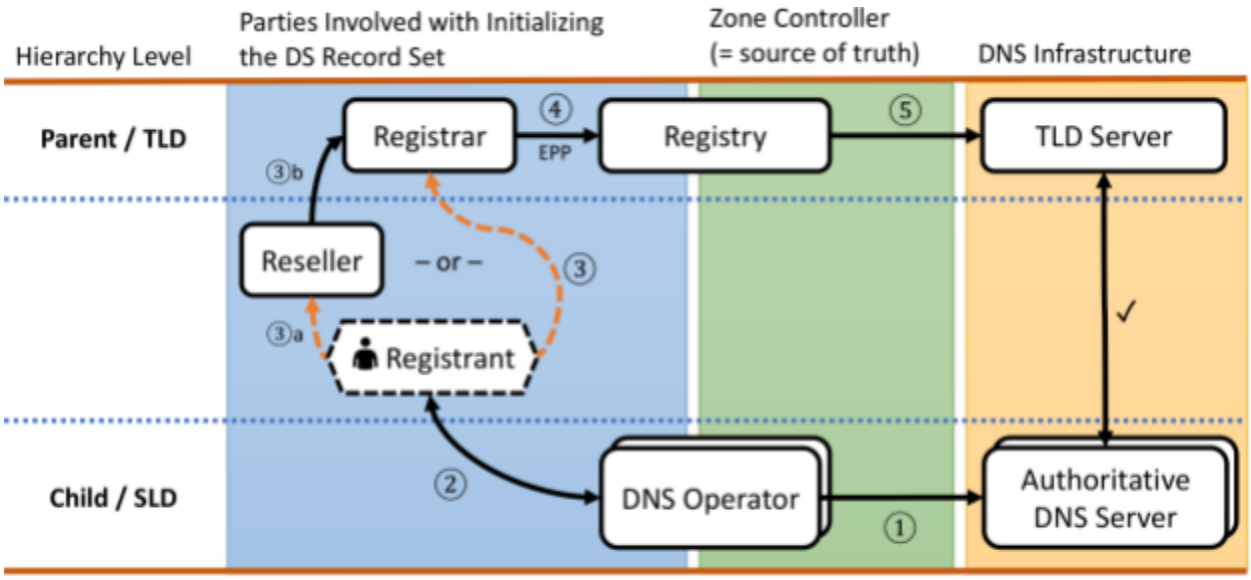

Figure 2. Entities and their relationships during DS provisioning.

To begin with, tasking the DNS operator with signing the zone is not sufficient for activating DNSSEC, which also includes securing the delegation. The registrant has to perform the actual DS initialization (or update) by pulling the DNSSEC information from the DNS operator and pushing it to the Parent zone registry. 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 might 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. 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,[15] as several kinds of values have to be entered. User interfaces vary considerably across different DNS operators and registrars. Some use single-string DS input fields, some use separate fields for the DS record's semantic substructure, and some use drop-down menus for cryptographic

[15] Peter Thomassen, "DS in the Wild: Lessons from a Campaign to Configure DS Records," (presentation, ICANN 78 Tech Day, Hamburg, Germany, October 23, 2023). Refer to slides 6 and 7, https://static.sched.com/hosted_files/icann78/4d/7%202023-10-23%20ICANN78%20Tech%20Day%2C%20DS%20in%20the%20Wild.pdf.

algorithms with numerical values or with mnemonics; see Appendix C for examples. In some cases, DS provisioning forms are not available at all.[16]

The handling of DNSSEC parameters involves technical details that registrants may not be familiar with. For example, the exact type of information which the registrant needs to retrieve and forward is dependent on the registry's policy: Some only accept DNSKEY-style data (containing the public key) and compute the DS record from it, while others will accept DS-style data directly; some accept both (RFC5910).[17] As a result, Child DNS operators – unaware of exactly what the Parent expects – often provide both types of data (see 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.

Lastly, to put a domain name into basic operation, all that is required is a working insecure delegation (NS records). Although DNS resolution remains insecure unless DS records are provisioned, the name is usable even without them.[18] Due to the fact that "it already works," domain owners may not be inclined to spend the extra work to secure it. In fact, doing so would multiply their cost, as (1) DS records have higher complexity than NS records (four fields instead of one), and (2) cannot be reused for other domains in a portfolio (unlike NS records).

Hence, while the act of manually copying or entering the appropriate information may, in itself, not pose a significant challenge, there is a gap between what the average person trying to set up a registration 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 surveying DNSSEC deployment[19] found that using DNSSEC with third-party DNS operators requires the domain owner to take a number of steps that many did not successfully complete, leading to a failure rate of 40% of DNSSEC deployment attempts at a large DNS operator.

4.2 Automatic Methods

In the classic (original) DNS model (which pre-dates the concepts of registries, registrars etc.), such interactions are done in-band, using DNS resource records and implicit signaling. Since

[16] Thomassen, "DS in the Wild," 6-7.
[17] "RFC 5910 - Domain Name System (DNS) Security Extensions Mapping for the Extensible Provisioning Protocol (EPP)," *IETF*, May 2010, https://datatracker.ietf.org/doc/html/rfc5910.
[18] This is reminiscent of when HTTPS was sparsely deployed, because it was much work and the page already worked with HTTP alone. This view was overturned through the advent of automation. Section 2.2 of Josh Aas et al., "Let's Encrypt: An Automated Certificate Authority to Encrypt the Entire Web," (CSS '19: proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security, London, United Kingdom, November 6, 2019). *Association for Computing Machinery*, New York, NY, USA, 2473–2487. https://doi.org/10.1145/3319535.3363192.
[19] Taejoong Chung, et. al., "Understanding the Role of Registrars in DNSSEC Deployment," (IMC '17: proceedings of the 2017 Internet Measurement Conference, London, United Kingdom, November 1, 2017). https://doi.org/10.1145/3131365.3131373.

then, the ecosystem has evolved with the introduction of new actors such as registrar and resellers. These new actors access a shared registry upon the registrant's registration request using an out-of-band mechanism (typically, using the Extensible Provisioning Protocol (EPP)[20]). In this scheme, communication for DS provisioning purposes involves multiple intermediaries handling the delegation information, causing a significant increase in terms of complexity.

Still, the classic entities (parent zone and delegated zone) exist on either side of the communication path. As a result, both direct in-band provisioning as well as out-of-band paths via one or multiple intermediaries can be imagined.

When eschewing DNS signaling completely and instead considering to use out-of-band API methods for automated DS record management, a number of issues would need to be addressed. These include the issue of mutual authentication of the involved parties, the (current) lack of standardization of a single API method to perform a DS update, and the question of "why embark on API development when in-band DNS methods are already available". Of these aspects, the authentication problem deserves special attention: as the registry/registrar in general lacks any information about the child DNS operator except for its NS hostnames, it appears to be an impossibility to establish trust automatically without resorting to a mechanism that relies on mappings securely derived from these hostnames.

Today, no single mechanism is used for DS provisioning; rather, a variety of approaches are deployed by a small number of actors each, typically with limited and non-interoperable automation capabilities.

4.2.1 Pull-based

In this model, the parent side (registrar or registry) takes an active role and collects the DNSSEC parameters from the Child DNS operator. In the general case, the mechanism consists of two components:

1. The DNS operator makes it known which public key information is to be included in the delegation's DS record set in the parent zone.
2. The registry or registrar fetches and validates this information and then updates the DS record set accordingly.

In existing deployments of this scheme, the first step is realized by placing CDS and/or CDNSKEY records[21] at the apex of the child zone (with the same owner name as the zone's SOA record, RFCs 7344, 8078). Next, the information published by the Child DNS Operator is fetched by the parent. In the RRR Model, this can be done by the registrar who then forwards the information to the registry, typically via EPP (see Figure 3). Alternatively, the registry itself can fetch the information and then update the DS record set. If the domain has a registrar, the registry

[20] S. Hollenbeck, "STD 69 RFC 5730 - Extensible Provisioning Protocol (EPP)," August 2009, https://doi.org/10.17487/RFC5730.

[21] 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 to compute DS records. "RFC 7344 - Automating DNSSEC Delegation Trust Maintenance," *IETF*, September 2014, https://datatracker.ietf.org/doc/rfc7344/ and "RFC 8078 - Managing DS Records from the Parent via CDS/CDNSKEY," *IETF*, March 2017, https://datatracker.ietf.org/doc/rfc8078/.

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.[22]

In order to maintain the integrity of the process, the integrity of the information being exchanged needs to be verified, both for DS initialization (DNSSEC bootstrapping) and for DS updates. When relying on the DNS itself for communication, updates can be authenticated via the child's existing chain of trust (RFC 7344), and initial provisioning can be secured using the Authenticated DNSSEC Bootstrapping protocol[23] (RFC 9615).

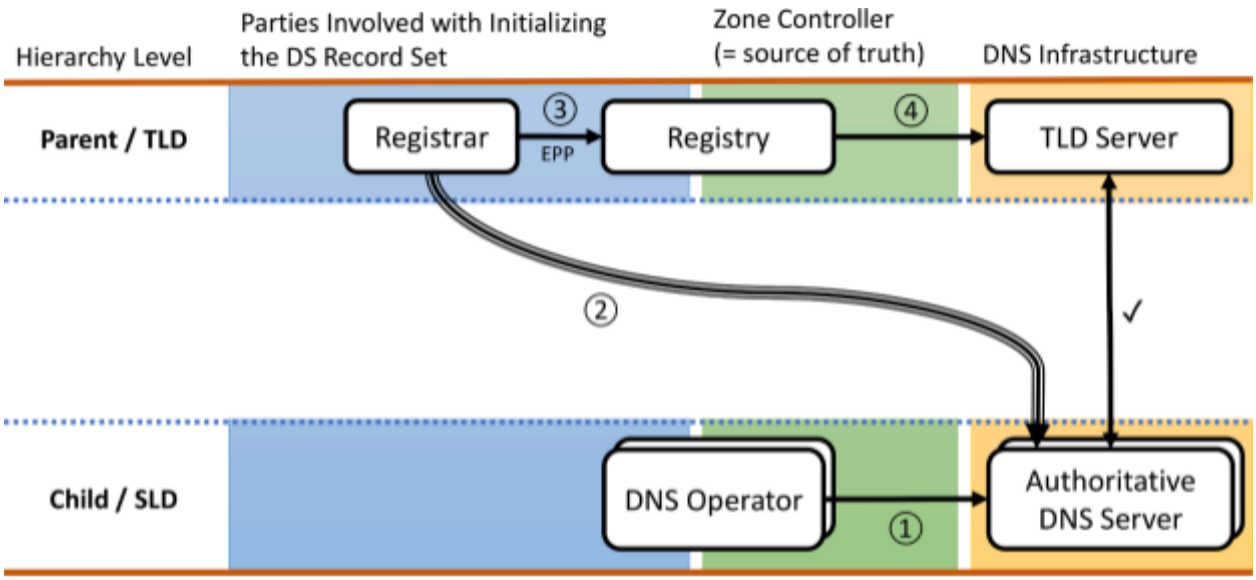

Figure 3. Simplified entity relationships during pull-based DS provisioning, performed by the Registrar.

[22] If the registry and registrar both look 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 making update attempts. See also Section 4.4 and Appendix B.2 in this report.
[23] "RFC 9615 - Automatic DNSSEC Bootstrapping using Authenticated Signals from the Zone's Operator," *IETF*, July 2024, https://datatracker.ietf.org/doc/rfc9615/.

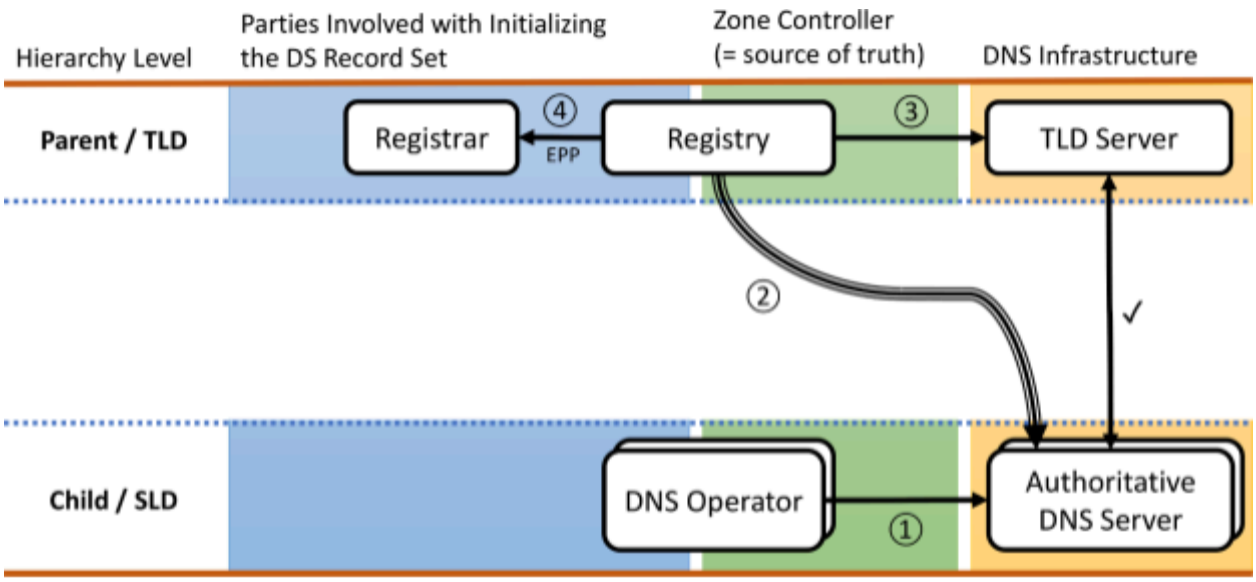

Figure 4. Simplified entity relationships during pull-based DS provisioning, performed by the Registry.

Being the only standardized solution, this set of techniques has been deployed by various registries, registrars and Child DNS operators:[24] Scanning of CDS/CDNSKEY records is currently in use at about 10 ccTLDs and one Regional Internet Registry (RIPE NCC, for reverse DNS zones), and some gTLD registries have expressed interest.[25] On the child side, a number of operators like Cloudflare, DNSimple, and GoDaddy DNS publish corresponding records, so that automation is automatically in effect for supporting parents. Some registries, registrars, and Child DNS operators also have implemented Authenticated DNSSEC bootstrapping.

The pull-based approach avoids most of the issues with the registrant-centric provisioning approach (see Section 4.1). The required information is exchanged directly between the relevant entities; the registrant is not involved as a relaying party, and the role of DNS Operators can be smoothly integrated into the DS update process without any additional exchange of credentials.

The main downside of the pull-centric approach is that it is not known in advance when the Child DNS Operator will publish updated CDS/CDNSKEY records. Therefore, periodic scanning is used to discover whether a change has occurred. Barring blocking reasons (such as registration locks), scanning needs to be done for all domains under management by the registrar or registry: Child zones that already have DS records might have published a new request for

[24] A list is available from: oskar456, "Support for CDS/CDNSKEY/CSYNC Updates," *GitHub,* https://github.com/oskar456/cds-updates.
[25] "Automated DNSSEC Configuration (CDS Scanning)," *CentralNIC Registry*, July 18, 2022, https://centralnic.support/hc/en-gb/articles/5957742209309-Automated-DNSSEC-Configuration-CDS-scanning and Michael Baulnd et. al., "CDNSKEY in TANGO Registry Services CORE Registry Software," (presentation, ICANN76, Cancun, Mexico, March 2023) https://static.sched.com/hosted_files/icann76/fd/4.5%20Bauland%20-%20CDNSKEY%20Support%20in%20TANGO%20Registry%20Services.pdf.

modification of the domain's DS record set, and newly signed zones without DS records might be awaiting DNSSEC bootstrapping as described in the previous section.

When a delegation is initially established, it is preferred from a security perspective for the new zone to be secured immediately. In principle, when a new domain is registered, the registry or registrar could poll for DNSSEC material and bootstrap the domain fully signed on registration.

As noted in section 6.1.1 of RFC 7344, full zone scanning comes with a cost proportional to both its frequency and the number of domains to scan. Although operational experience[26] indicates that it is currently feasible to be performed daily on zones on the order of several hundred million, it is nonetheless inefficient as only very few scans are expected to result in an actual DS update.

Furthermore, 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 a signal and processing it ranges between "almost immediately" (when the scan happens to occur just after the publication) and the maximum interval between scans (e.g., 1 day). This is another kind of "uncertainty and delays," as are also experienced in the registrant-centric approach.

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

4.2.2 Push-based

In the push-based approach, the Child DNS operator would contact the registrar or registry and push the desired DS update directly into the parent-side system. Although no method using this approach has been standardized, several schemes are conceivable.

In-band

One way is to send the new CDS/CDNSKEY value as a DNSSEC-signed RRset payload to an endpoint announced by the registry or registrar, in a manner analogous to DNS UPDATE (RFC 2316), thereby removing the requirement for the parent to poll the child zones for the new CDS/CDNSKEY RRset. This update-like method can be effective when the child zone is operated by a single DNS operator.

However, it does not cleanly apply to the situation of multiple DNS operators using per-operator keys (model 2 of RFC 8901), where there are multiple DS records to be maintained in the parent zone. The issue here is that there is some ambiguity as to which DNS operator's DS value in the RRset is to be updated. Additional information would be needed in the update signal, such as providing the old and new DS values for context.

[26] An SSAC survey of existing implementations revealed that scans are generally performed daily on a very small number of machines. Peter Thomassen, "Scalability of Scanning of CDS/CDNSKEY Records," (presentation, ICANN 77 DNSSEC and Security Workshop, Washington D.C., June 12, 2023) https://tinyurl.com/bkw4msfz.

Out-of-band

Alternatively, out-of-band solutions such as REST APIs could be used. The SSAC acknowledges that customized APIs are in use with some registries/registrars or DNS operators. However, we consider such solutions *out of* scope for the current document for the following reasons.

First, the in-advance credential exchange required by out-of-band APIs changes the authentication problem of DS provisioning into one of credential exchange, reproducing all of the automation problems of initialization. Second, the lack of standardization in APIs means that registry/registrar operators have to develop specific code to interact with each DNS operator, and vice versa.

Solving these requires either resorting to a pre-existing trust channel derived from the Child DNS operator identity (comparable to in-band DNSSEC bootstrapping) or restructuring the governance model between DNS operators and ICANN contracted parties, which this document does not intend.

4.2.3 Hybrid: Combining Aspects of Pull/Push

As an alternative to pull-based scanning, the Child DNS operator may inform the parent when an update to the DS record is desired. This avoids continual polling of the CDS/CDNSKEY resource record to detect changes, and replaces it with an on-demand notification signal to the parent.

The purpose of this notification would be to indicate that a new CDS/CDNSKEY value is available. The parent can then retrieve the CDS/CDNSKEY record using a DNS query (using the same method as the "Pull" approach), and validate the retrieved resource record value as described in Section 4.2.1. This approach is an instance of a "polling trigger" described in section 6.1.2 of RFC 7344.

The signaling method to trigger polling is proposed to be an extension to the DNS NOTIFY method (RFC 1996). The standardization of this method is being undertaken in the DNSOP Working Group of the IETF based on a draft specification titled “Generalized DNS Notifications.” [27] An important component of this work is to standardize a method for the child to discover where to direct such a notification signal.

Moreover, there is the issue of error notification and handling in case the parent is unable to update the DS value with the new value. There currently is no standard defining whether or how such a condition should be signaled to the child zone operator(s), or how much detail in terms of a diagnostic is appropriate. The “Generalized DNS Notifications” draft proposes to use an RFC 9567 Report-Channel option in the notification message to advertise error handling capabilities.

4.3 Discussion

Of the various DS provisioning methods when the DNS operator, registrar and registry functions are performed by separate, non-coordinating entities, the *registrant-centric* method (section 4.1) currently is the most common one. However, there is a gap between what most registrants would

[27] Johan Stenstam et. al., “Generalized DNS Notifications,” *IETF*, July 21, 2024, https://datatracker.ietf.org/doc/draft-ietf-dnsop-generalized-notify/.

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.

Of the automatic methods, the *pull-based* model (section 4.2.1), which allows for fully automatic DS provisioning coordination between the Child DNS operator and a parent-side entity (registrar or registry), sees most of the current adoption by registries and registrars. Although the ongoing use of this method has not surfaced any serious operational problems, the mechanism suffers from some inefficiency and delays related to the scanning-based approach which it employs.

Alternatives that address these disadvantages are based on the push concept where the DNS operator informs the parent of a desired DS update (section 4.2.2). While this approach has the potential to improve efficiency, a number of aspects such as error reporting require standardization before the solution can be considered complete.

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 an industry best practice for DS management without the need for human involvement is a natural extension of this development.

4.4 Operational Considerations for DS Automation

Finally, when deploying DS automation, registries and registrars need to consider a number of operational aspects:

- Should DS automation involve the registrar or the registry, or both?
- What kind of validity checks should be performed on DS parameters? Should those checks be performed upon acceptance, or also continuously when in place?
- How do TTLs and caching impact DS provisioning? How important is timing in a child key change?
- How are conflicts resolved when DS parameters are accepted through multiple channels (e.g. via a conventional channel and via automation)? In case both the registry and the registrar are automating DS updates, how to resolve potential collisions?
- What is the relationship with other registration state parameters, such as registry or registrar locks?
- Should a successful or rejected DS update trigger a notification to anyone?

Not all existing DS automation deployments have made the same choices with respect to these questions, leading to somewhat inconsistent behavior across TLDs.[28] From the perspective of a

[28] .ch and .li: "Automated DNSSEC Provisioning: Guidelines for CDS processing at SWITCH," *SWITCH*, April 25, 2024, https://www.nic.ch/export/shared/.content/files/SWITCH_CDS_Manual_en.pdf;
.cz: "Automated Keyset Management," *FRED*, July 1, 2024, https://fred.nic.cz/documentation/html/Concepts/AKM.html;
.sk: https://centralnic.support/hc/en-gb/articles/5957742209309;
.se: "DNSSEC Practice Statement (DPS)," *Internet Stiftelsen*, September 6, 2022, https://internetstiftelsen.se/app/uploads/2019/02/se-dnssec-dps-eng.pdf;

registrant with domain names under several TLDs, this is unexpected and confusing. It is thus crucial to address the above operational aspects in a uniform manner across TLDs, to obtain predictable behavior.

Where the zone is operated by multiple DNS operators using per-operator keys (model 2 of RFC8901) then there is no standards-defined process to handle CDS/CDNSKEY records. A conservative approach is for the child zone DNS operators to coordinate the CDS/CDNSKEY RRsets from all the zone's DNS operators and each operator to publish the collection of values. A more detailed examination of these issues is described in Internet-Draft “DNSSEC automation”[29] and “Consistency for CDS/CDNSKEY and CSYNC is Mandatory”[30]. It is likely that additional standard specifications and operational procedures may be developed for multiple DNS operators in the future (this is mentioned in Section 7).

Furthermore, registrants modifying DS records on their own (e.g., for migrating their domain to a different DNS operator) should consider turning off DS automation to ensure that their changes are not overwritten. This can be achieved by setting an appropriate lock through the registrar, or – when using CDS/CDNSKEY-based automation – by adjusting or removing those records. It is worth emphasizing that independently of DS automation, maintaining a secure delegation during a provider transition requires additional coordination between operators (in particular the exchange and inclusion of public keys in each other’s DNSKEY record set[31]). DNS operators are thus advised to not request conflicting DS updates while such coordination is ongoing. Supporting transitions through additional automation is part of future work (see Section 7).

5 Findings

Finding 1: The operation of a DNSSEC-secured zone in the DNS requires the management of DS records in the parent zone. Today, when the domain’s DNS service is not operated by the registrar, DS management can involve intermediaries manually handling the delegation information, introducing significant complexity, idiosyncratic interfaces, and chance for error.

Finding 2: The management of DS record sets should occur in a smooth, efficient, and faultless fashion. DS management automation, which is a natural extension for automating DNSSEC operations, can contribute to this goal.

Finding 3: There are two basic types of approaches to automate DS management: pulled-based and pushed-based. The *pull-based* approach is most developed with Internet standards and deployment experience (RFCs 7344, 9615). A third, hybrid approach is being developed and incorporates *push-based* features into the *pull-based* method, by extending it with polling-trigger

---

RIPE NCC: “Configuring Reverse DNS,” *RIPE*, July 4, 2024. https://apps.db.ripe.net/docs/Database-Support/Configuring-Reverse-DNS/.

[29] Ulrich Wisser et. al., “DNSSEC Automation,” *IETF*, October 21, 2013, https://www.ietf.org/archive/id/draft-ietf-dnsop-dnssec-automation-02.txt.

[30] Peter Thomassen, “Consistency for CDS/CDNSKEY is Mandatory,” *IETF*, October 2, 2023, https://www.ietf.org/archive/id/draft-ietf-dnsop-cds-consistency-04.txt.

[31] “RFC 6781- DNSSEC Operational Practices, Version 2,” *IETF*, December 2012, Section 4.3.5.1, https://www.rfc-editor.org/rfc/rfc6781.html#section-4.3.5.1

notifications and error reporting for efficiency and reliability. All three approaches are amenable for removing human intervention from the process.

Finding 4: Enabling DS management automation requires the registry/registrar to make a number of technical decisions. It is important to develop a set of best common practices.

6 Recommendations

Recommendation 1: If a registry or a registrar wishes to implement DS automation for third-party DNSSEC operations, the current recommended interoperable mechanism is CDS/CDNSKEY (RFCs 7344, 9615). This mechanism has limitations as described in this document but has been deployed and there is operational experience in using it.

Recommendation 2: ICANN Org should support registries and registrars who want to implement DS automation using the mechanisms from Recommendation 1, such as by facilitating a Fast-Track RSEP[32].

Recommendation 3: ICANN org should facilitate the development of operational guidance for registries and registrars around the implementation of DS automation, in particular the operational aspects outlined in section 4.4.

7 Future and Related 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.3, the impact of registration locks on DS maintenance requires non-trivial analysis. One practical approach is to allow 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.

- **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 theory of

[32] "Registry Services Evaluation Policy," *ICANN*, July 25, 2006, https://www.icann.org/resources/pages/fast-track-rsep-process-authorization-language-2019-06-14-en / https://www.icann.org/rsep-en.

multi-signer setups is well understood, more work is needed for the development of both automated protocols and practical guidance.

- **Provider transfer without interrupting DNSSEC service.** 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-provider DNSSEC setup is needed to ensure glitch-free operation throughout the transition. While a small number of DNS providers support such transitions via manual configuration, the required protocols would ideally be supported by all DNS providers and registrars that offer signed DNS service. As a side effect, multi-signer automation (see above) will enable this use case as well.

- **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.

- **Extensions to the DNS delegation mechanism.** The IETF DELEG working group is investigating new DNS signaling mechanisms that allow parents to return additional DNS delegation information about their children. Still in an early phase, this effort might lead to protocol extensions that touch upon the problem space addressed in this document and might justify updated advice at a later time. Nevertheless, it seems unlikely that any future extensions would render DS management automation superfluous, as legacy resolvers are expected to continue to rely on DS records as they are defined today.

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.

In the interest of transparency, these sections provide the reader with information about aspects of the SSAC process. The Acknowledgements section lists the SSAC members, outside experts, and ICANN staff who co-authored or contributed directly to this particular documentor who provided reviews. The Disclosures of Interest section points to the biographies of all SSAC members, which disclose any interests that might represent a conflict—real, apparent, or potential—with a member'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.

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
Joe Abley
Steve Crocker (co-chair)
Patrik Fältström
Ondřej Filip
James Galvin
Geoff Huston
Merike Kaeo
Warren Kumari
Jacques Latour
John Levine
Ram Mohan
Russ Mundy
Doron Shikmoni
Peter Thomassen (co-chair)
Jiankang Yao

Invited Guests

Mats Dufberg
Mark Elkins
Shumon Huque
Eric Osterweil
Nicklas Pousette
Johan Stenstam

ICANN Staff
John Emery
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

There were no 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). 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), 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 operationally 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.[33]

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.

EPP: The Extensible Provisioning Protocol (EPP), which is commonly used for communication of registration information between registries and registrars. Currently, it is Internet Standard 69, <https://www.rfc-editor.org/info/std69>.

[33] In a DNSSEC multi-signer setup with multiple signing entities, the zone data 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 zone: a DNS zone that holds a delegation for a Child zone.

Parent (DNS operator): The DNS operator responsible for a Parent zone, and thus involved with the maintenance of its delegations' DNSSEC parameters (in particular, the acceptance and verification 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). In the RRR Model, 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: 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.

TLS: An authentication and encryption protocol widely implemented in browsers and web servers. HTTP traffic transmitted using TLS is known as HTTPS.

Appendix B: Operational Considerations on 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).

No recommendations are given; rather, this Appendix is meant to provide a starting point for future development of additional guidance.

Practical experience with multi-signer setups is currently limited. We would like to highlight that with further experience, some of the below operational considerations may uncover other issues associated with the automation of multi-signer scenarios that are not included in this section.

B.1 Validity Checks and Safety Measures

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

Analysis:

To maintain the basic resolution function, it is a reasonable strategy to avoid the deployment of bad DS record sets in the parent zone. It is therefore desirable for the Parent to verify that the DS record set resulting from an automated (or even manual) update does not break DNSSEC validation if deployed, and otherwise cancel the update. (The CDS/CDNSKEY specification captures this in RFC 7344 Section 4.1.)

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 a brief time period 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[34] such

[34] Petr Špaček and Viktor Dukhovni, "March Towards Shorter DNSSEC Outages," (presented at OARC 41, Danang, Vietnam, September 6, 2023, https://indico.dns-oarc.net/event/47/contributions/1010/attachments/958/1811/DS%20and%20DNSKEY%20TTL%20experiment.pdf.

TTLs to have negligible impact on the overall load of a registry's authoritative nameserver infrastructure.

It thus appears advisable that registries *reduce* the DS record set's TTL to a low value *at the time it is updated*, and restore the normal TTL value after a certain amount of time has passed. Pragmatic values for the reduced TTL value range between 5–15 minutes. The TTL reduction should be in 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 RRset (usually after a few days) provides a convenient opportunity for resetting the TTL. The IETF REGEXT Working Group is finalizing a specification that allows such on-demand adjustments to the DS TTL.[35]

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, given the goal of enabling timely withdrawal of a botched DS RRset, it is not equally important for the previous (functional) DS RRset to be abandoned very quickly. 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 once the Parent received information that a DS update is desired; doing so at this point, however, 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 does not seem justified. On the other hand, *not* reducing the old DS TTL ahead of time has the advantage of providing some resiliency against a potentially 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). Such a reporting mechanism may also be used for notifications in case a problem is detected with an operational (unchanged) DS RRset, such as due to an incompatible change in the Child zone. For details, see Appendix B.4.

B.2 Multiple Submitting Parties

How are conflicts resolved when DS parameters are accepted through multiple channels (e.g. via a conventional channel and via automation)? In case both the registry and the registrar are automating DS updates, how to resolve potential collisions?

Analysis:

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

- The registry can, through some automated method, receive information about intended DS updates directly from the Child DNS Operator, and update the DS records;

[35] Gavin Brown, "Extensible Provisioning Protocol (EPP) mapping for DNS Time-To-Live (TTL) values," *IETF*, July 22, 2024, https://datatracker.ietf.org/doc/draft-ietf-regext-epp-ttl/.

- The registrar can, through some automated method, receive this information 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 offer several considerations in this context:

- When a manual DS update is performed without ensuring that the information informing automated DS maintenance requests (such as CDS/DNSKEY records) are updated as well, the delegation's DS records may be reset to their previous state at the next run of the automation process. (Note that in case of CDNSKEY/CDS processing, these records will only remain valid when signed with keys authorized via the manually deployed DS records. Validity checks described in Appendix B.1 further ensure that updates do not break validation.)

  Some experts have proposed suspending DS automation once a manual DS update has occurred, e.g., 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 comes with a high risk of "timing surprises" – namely, when the SOA serial changes for reasons unrelated to the DNSSEC key configuration. Any arbitrary modification to the zone content would then 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), so that an apparent serial increase might actually be an illusion. It is therefore advised to not follow this practice.

  All things considered, it appears advisable that automated DS updates 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.[36] DS automation should then be suspended so that automatic re-initialization (bootstrapping) does not occur. In all other cases, any properly authenticated DS updates received through an automated method should be considered as the current intent of the domain owner.

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

[36] A cleaner way to disable DNSSEC is to use the automation protocol's designated procedure. In case of CDS/CDNSKEY automation, special records can be configured as described in Section 4 of "RFC 8078 - Managing DS Records from the Parent via CDS/CDNSKEY," *IETF*, March 2017, https://datatracker.ietf.org/doc/html/rfc8078

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

- 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 disappears entirely when using "Generalized DNS Notifications" for triggering DS maintenance through one party's designated endpoint (see Section 4.2.2), and can otherwise 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's DS update requests, such as when CDS/CDNSKEY records are retrieved from different vantage points. This may lead to flapping of DS updates, which is not expected to be harmful as either DS RRset will allow for the validation function to continue to work. The effect subsides as the Child's state eventually becomes consistent (expected to occur within one TTL), and can be further reduced by checking overlapping update requests for consistency.

  As a standard aspect of key rollovers (RFC 6781[37]), the Child DNS operator is expected to monitor propagation of Child zone updates to all authoritative nameserver instances, and only proceed to the next step once replication has succeeded everywhere and the DS record set was subsequently updated (and in no case before the DS RRset's TTL has passed). Any breakage resulting from improper timing on the Child side is outside of the Parent's sphere of influence, and thus out of scope of DS automation considerations.

B.3 Registration Locks

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

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 locks and which impact on DS automation would appear plausible is given in Table 2. Some registries may offer additional types of locks whose meaning and set/unset mechanisms are defined according to a proprietary policy.

[37] "RFC 6781 - DNSSEC Operational Practices, Version 2," *IETF*, December 2012, https://www.rfc-editor.org/rfc/rfc6781.

	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 2. SSAC analysis of registry and registrar locks[38] and considerations on their impact on DS automation. "(allowed)" indicates that it does not appear plausible for this lock alone to prevent DS maintenance, but more restrictive locks may take precedence (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).

When a *serverUpdateProhibited* lock ("registry lock") is in place, there exists an expectation that this lock renders all otherwise updateable registration data immutable.[39] It seems logical to extend this lock to DS updates as well.

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

[38] For more information on locks, see https://www.icann.org/resources/pages/epp-status-codes-2014-06-16-en.
[39] The only exception to this is the registry's out-of-band process for removing this type of lock. See Registry Lock Service, https://www.verisign.com/en_US/channel-resources/domain-registry-products/registry-lock/.

maintenance needlessly difficult. It therefore seems reasonable 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, 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. It is worth recalling that any authenticated DS update request (such as when published via CDS/CDNSKEY records on all nameservers) constitutes a legitimate request in the name of the registrant. Further, the resulting DS update is subject to the parent's acceptance checks, and not applied when incompatible with the DNSSEC keys published in the child zone (see Appendix B.1).

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). Any attack on the registration data that is feasible in the presence of a registrar lock is also feasible regardless of whether DS maintenance is done automatically; in other words, DS automation is orthogonal to the attack vector that a registrar lock protects against. Considering that automated DS updates are required to be authenticated and validated for correctness, it thus appears that honoring such requests, while in the registrant's interest, comes with no additional associated risk. (Automated DS maintenance may be disabled by requesting a registry lock, if so desired.)

Suspending automated DS maintenance at the registrar or registry therefore does not seem justified 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 provides insufficient grounds for suspending DS maintenance.

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

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 is requesting DS initialization through an authenticated, automated method expresses the registrant's intent to have the delegation secured. There would seem to be little reason for the registrant to 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 (to their Child DNS operator) the intent of securing their domain with DNSSEC. It therefore appears 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.

B.4 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 without an update request to the DS record set. 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 may result.

The SSAC therefore suggests that entities performing automated DS maintenance should report on error conditions they encounter. Even when no update was requested, it may be worthwhile to occasionally check whether the current DS contents would be accepted today (see Appendix B.1), and communicate any failures without changing the published DS record set.

The following situations may be of particular interest for 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 under automatic maintenance, see Appendix B.2);

2. The DS record set has been removed
    a. manually, or
    b. automatically via the automation protocol's designated process;

3. A pending DS update cannot be applied due to an error condition. There are various scenarios where an automated DS update might have been requested, but can't be fulfilled. These include:
    a. The new DS record set would break validation/resolution or is not acceptable to the Parent for some other reason (see Appendix B.1);
    b. A lock prevents DS automation (see Appendix B.3);
4. No DS update is due, but it was determined that the Child zone is no longer compatible with the existing DS record set (e.g. lack of DNSKEY algorithm).

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

- Child DNS operator, via the report channel contained in the "Generalized DNS Notification" that triggered the DS update, or via email (from SOA contact data);
- Registrar (if DS automation is performed by the registry), via EPP (or similar channel);
- Registrant (domain holder) or technical contact, via email.

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 advisable to first notify the domain's technical contact and the DNS operator serving the affected Child zone, and only if the problem persists for a prolonged amount of time (e.g., three days), notify the registrant.

When the RRR model is used and the registry performs DS automation, the registrar should always stay informed of any DS changes, e.g., via EPP (RFC 8590[40]).

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, notifications to humans 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, together with the currently active configuration, be made accessible to the registrant (or their designated party) through the customer portal available for domain management.

[40] "RFC 8590 - Change Poll Extension for the Extensible Provisioning Protocol (EPP)," *IETF*, May 2019, https://www.rfc-editor.org/rfc/rfc8590.html.

B.5 Consistency Considerations

Are DS parameters best conveyed via DS format or DNSKEY format (or both)? How are conflicts resolved?

Generic Analysis

DS records can be generated from either DS-style or from DNSKEY-style input format. The former is identical to that of DS records (so can be taken verbatim), while for the latter, generation of a DS record involves computing a hash and other information based on the DNSKEY-format information.

Whether DS or DNSKEY format 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 DS records, enabling the Child DNS operator to deploy (for example) experimental hash digests and removing the need for registry-side changes when new digest types become available.

The need to make a choice in the face of this dichotomy is not 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.

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 with insufficient entropy). Besides, as the DS format contains a field indicating the hash digest type, objectionable ones (such as those outlawed by RFC 8624) can still be rejected even when parameters are accepted as DS-style input, by inspecting that field.

The fact that more than one input type needs to be considered burdens both Child DNS operators and Parents with the need to consider how to handle this dichotomy. 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.

In the meantime, it seems worthwhile for both Child DNS operators and Parents implementing automated DS maintenance to act as to maximize interoperability. How that is best achieved depends on the automation mechanism in use. In any case, it will be helpful to record the registry's practices in the DNSSEC Practice Statement (DPS).

Impact on CDS/CDNSKEY-based Automation

For CDS/CDNSKEY-based automation, the following additional considerations are offered:

- There exists no discovery protocol for the Parent's input format preference. Recognizing that Child DNS operators are generally ignorant about which format the Parent prefers, they are encouraged to publish both CDNSKEY records as well as CDS records. The choice of hash digest type should follow current best practice, currently RFC 8624.[41]

  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 bear the complexity cost of discovering 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 synthesizing them, or programmatically updating them at once.

- Parents, independently of their input format preference, are advised to require publication of both CDS and CDNSKEY records, and to enforce consistency between them, as determined by matching CDS and CDNSKEY records using hash digest algorithms whose support is required according to RFC 8624 Section 3.3. (For CDS records using another, unsupported hash digest, consistency is undefined and thus not required.)

  By rejecting the DS update if either type is found missing or inconsistent with the other, Child DNS operators are held responsible when publishing contradictory information. Registries can retain whatever benefit their choice carries for them, while at the same time facilitating a later revision of their choice. Similarly, this simplifies possible future deprecation of one of the two formats without breakage.

In order to further reduce the risk of wrongful DS deployment such as from nameserver hijacking or negligent multi-signer operation, it seems helpful to ensure that CDS and CDNSKEY responses are consistent across nameservers. This can be achieved by attempting to collect them from all authoritative nameservers listed in the delegation (one query per hostname suffices). When a key is referenced in a CDS or CDNSKEY record set returned by one nameserver, but is missing from another nameserver's answer, the Child's intent is unclear, and DS provisioning should be aborted. A current IETF draft document describes this in detail.[42]

[41] "RFC 8624 - Algorithm Implementation Requirements and Usage Guidance for DNSSEC," *IETF,* June 2019, https://datatracker.ietf.org/doc/html/rfc8624.
[42] Peter Thomassen, "Consistency for CDS/CDNSKEY and CSYNC is Mandatory," *IETF*, April 4, 2024, https://datatracker.ietf.org/doc/draft-ietf-dnsop-cds-consistency/.

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. Figure 5 below shows how the registrant can retrieve such information from one DNS operator's interface.

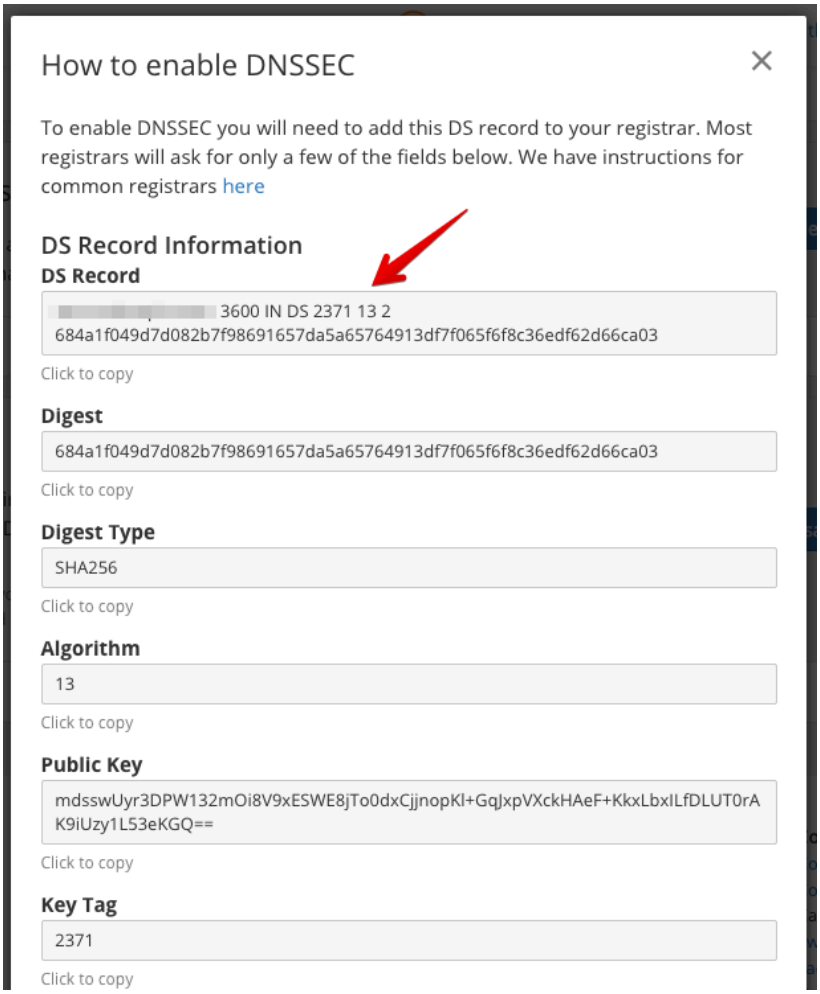

Figure 5. A DNS operator's Interface for registrant to Retrieve DS Record Information

Challenges to registrants for completing Step 1:

- The registrant first needs to understand that in order to have their delegation secured, they need to complete these steps. This may not be aligned with the registrant's expectations. Furthermore, a registrant 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 information.
- 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; typical values are 64 characters (digest type 2) and 96 characters (digest type 4).
    c. *Digest Type:* The hash algorithm 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")[43].
    e. *Public Key:* Public key of the associated DNSKEY record (in base64 encoding).
    f. *Key Tag:* Contains a non-unique identifier for the DNSKEY record referenced by this DS record.

    The registrant may operate under the assumption that the DS record is one long string ("presentation format"), which is a 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 registrar's 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

[43] "Domain Name System Security (DNSSEC) Algorithm Numbers," *IANA*, April 16, 2024, https://www.iana.org/assignments/dns-sec-alg-numbers/dns-sec-alg-numbers.xhtml.

DS parameters in the correct location in the registrar's interface. Figures below show some registrars' interfaces.

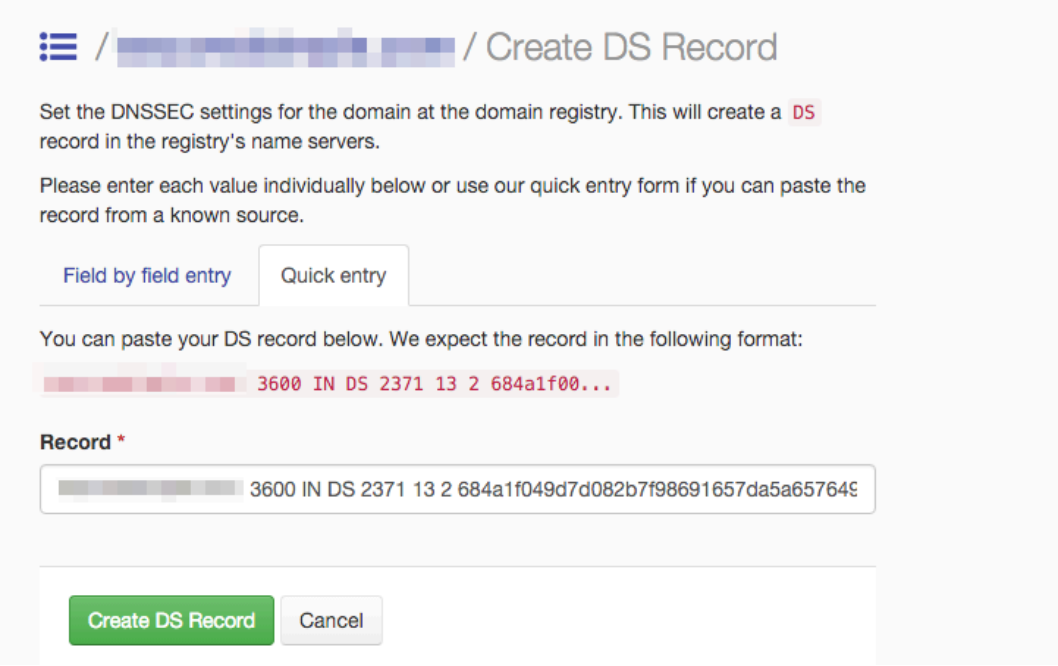

Figure 6. A registrar's interface for registrants to create DS Record. This interface requires direct input of the full DS record in presentation format (including the TTL, which is ignored by the registry and thus normally not included in the Child DNS operator's instructions).

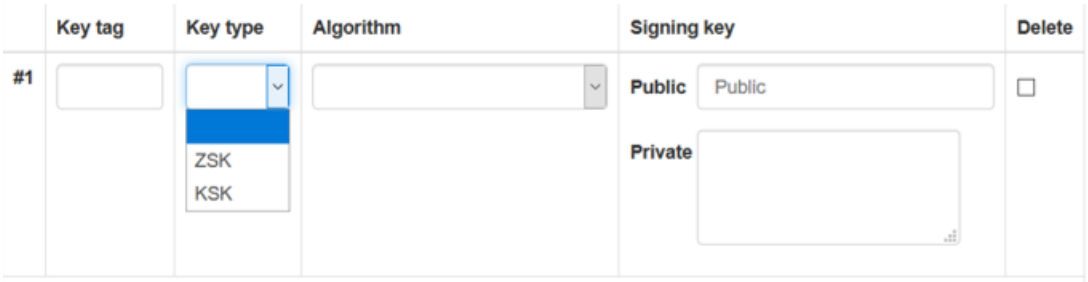

Figure 7. A registrar's interface for registrants to create DS Record. This interface requires entering the various components of the DNSKEY record, namely the Algorithm and the Public Key (labeled as "Signing Key – Public"). The value of "Key type" typically is 257/KSK and therefore normally not included in Child DNS operator's instructions; a non-expert will not know which value to pick. 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" field is superfluous, confusing, and potentially dangerous, as there is no need to input any private key material for DS provisioning.

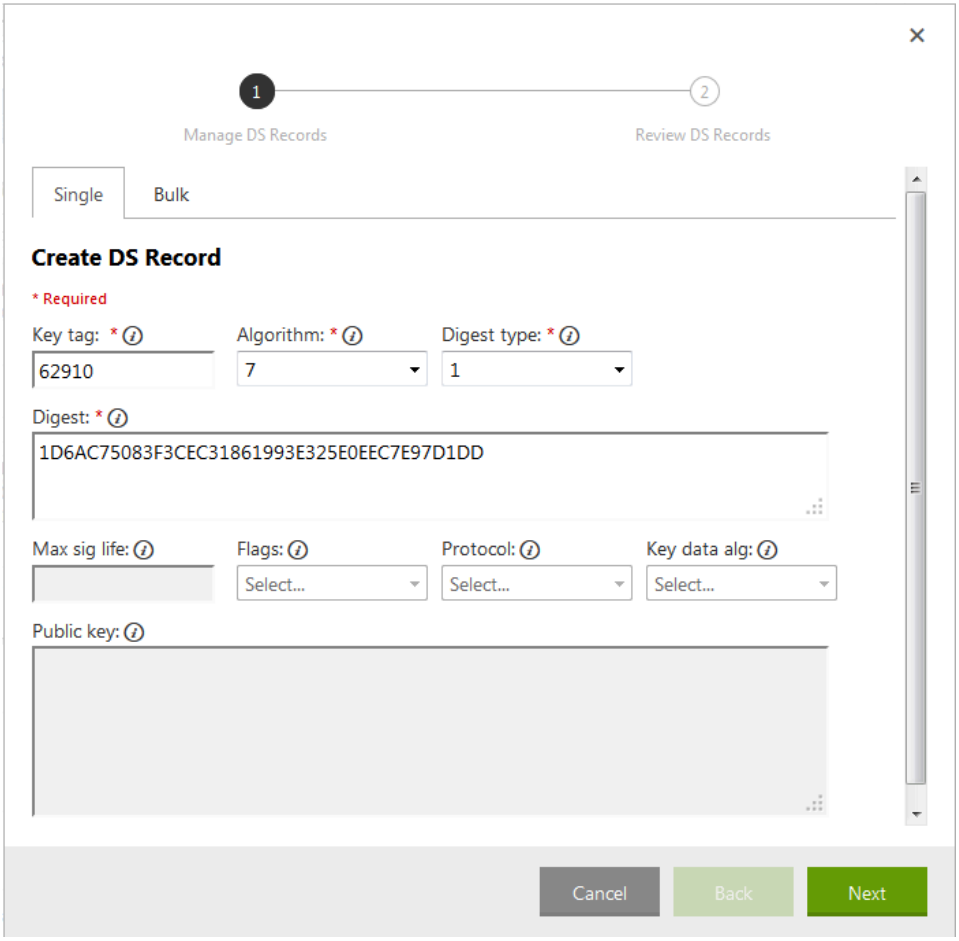

Figure 8. Another registrar's interface for registrants to create DS Record. It requires separate input of the various DS record components. Note that the numerical value of the Digest Type can be selected from a drop-down field (current selection: 1), whereas the Child DNS operator's interface used a mnemonic ("SHA256") for this field (see Figure 5). 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.