DNSSEC Deployment Guidebook for ccTLDs

ICANN Office of the Chief Technology Officer

Yazid Akanho & Paul Muchene

OCTO-029
12 November 2021

TABLE OF CONTENTS

This document is part of ICANN’s Office of the Chief Technical Officer (OCTO) document series. Please see the OCTO publication page for a list of documents in the series. If you have questions or suggestions on any of these documents, please send them to octo@icann.org.

This document supports ICANN’s strategic goal to improve the shared responsibility for upholding the security and stability of the Domain Name System (DNS) by strengthening DNS coordination in partnership with relevant stakeholders. It is part of ICANN’s strategic objective to strengthen the security of the DNS and the DNS root server system (RSS).

1 Introduction

In recent years, security has increasingly become a major issue on the Internet. Specific to the Domain Name System (DNS), several security protocols have been proposed and developed over the years, with the Domain Name System Security Extensions (DNSSEC) being one of the most significant protocols. DNSSEC helps to secure DNS responses by adding data origin authentication and data integrity protection.

The DNS root zone managed by ICANN was first signed with DNSSEC in July 2010. At the time this guidebook is published, all generic top-level domains (gTLDs) will be DNSSEC signed, in part due to contractual obligations with ICANN; while on the other hand, only about 60% of country code top-level domains (ccTLDs) have been signed. One of the reasons that could explain this trend at the ccTLD level is lack of visibility from ccTLD managers in the process of securing their zones with DNSSEC.

ICANN's Office of the Chief Technology Officer (OCTO) has therefore published this guidebook to assist ccTLD registry operators in their ability to better take ownership of the process that can help them sign their zones with DNSSEC. It does not cover the second aspect of DNSSEC which is validation that occurs primarily on DNS recursive resolvers usually located at Internet Service Providers (ISPs), in large public cloud operators, or in corporate networks.

1.1 Target audience for this document

This guidebook is primarily intended to provide ccTLD registry managers, staff, stakeholders, particularly registrars, registrants, and anyone else an overview of DNSSEC and how it can be implemented by a registry in zone signing. This document does not go into technical configuration details; rather it serves as a guide for a basic understanding of the DNSSEC protocol, the prerequisites, and deployment considerations in signing a ccTLD's zones.

Even if you are already operating a DNSSEC-signed TLD, this document could help you identify items to improve on, such as the algorithmic best current practices or the proper documentation of the overall DNS service. If you are a ccTLD operator, or administer zones under a signed ccTLD, this guide can help you get started. Notwithstanding that all gTLD operators have already signed their zones, they could also glean insights on DNSSEC operational best practices described in this document and use it as a baseline to raise awareness on DNSSEC for their registrars and registrants.

There are a plethora of documents on the theoretical, technical, and operational aspects of DNSSEC, and this guide cites several as reference. Readers are, therefore, invited to go through these cited documents if they wish to deepen their knowledge or understanding of either one or the other aspects related to DNSSEC.

2 DNSSEC and its value to the DNS

The Domain Name System (DNS) is a hierarchical, distributed, and decentralized naming system for the Internet. Similar to a phone book which translates names to phone numbers, the DNS helps convert domain name information into IP addresses and vice-versa. The DNS is considered a critical service on the Internet, but it was not originally designed with strong security mechanisms to provide integrity and authenticity of its data. Over the years, a number of vulnerabilities that threaten the reliability and trustworthiness of the DNS have been discovered, and DNSSEC helps to address some of them.

DNSSEC is primarily defined and specified in three Internet standards documents: RFC 4033, *DNS Security Introduction and Requirements*; RFC 4034, *Resource Records for the DNS Security Extensions;* and RFC 4035, *Protocol Modifications for the DNS Security Extensions*. DNSSEC uses public key cryptography (the generation of public and private key pairs) to add data origin authentication, data integrity, verification, and authenticated denial-of-existence capabilities to the DNS. Specifically, it adds digital signatures and a new set of resource record types and message header bits (flags) to the DNS, which can be used to verify DNS responses from a signed zone. It is worth noting that DNSSEC does not encrypt any DNS message data and, therefore, does not provide confidentiality.

Once a domain is signed with DNSSEC, digital signatures are generated by the zone administrator using a private key and published as a Resource Record Signature (RRSIG) record into the zone file as part of the domain's zone data. When a security-aware recursive resolver, also known as a validating resolver, sends a DNS query to an authoritative server of the signed domain, the DNS response contains the resource record in cleartext or in an unencrypted format and its associated digital signature. The resolver then uses the digital signature it has received to validate this DNS response. For this purpose, the validating resolver also requests other DNSSEC related information, such as the public key that is stored in the DNSKEY record and published by the domain administrator in the zone data.

3 Prerequisites and requirements to DNSSEC deployment

3.1 Documentation of the existing system

Due to the critical role the DNS plays on the Internet and the need to prevent service disruption in all circumstances, it is important to maintain up-to-date documentation that describes the DNS infrastructure, operations and processes. DNSSEC on the other hand, adds a certain additional level of complexity to existing DNS infrastructure and operations. Therefore, keeping up-to-date documentation is crucial to ensuring that a clear picture of the existing system is available for reference. This also allows for operational continuity in the event of staff turnover or infrastructure upgrades.

We recommend that the documentation summarize two major aspects: the governance policies of the ccTLD as well as the operational and technical aspects of the service.

Furthermore, it is advisable that the document contains the maximum information possible, omitting any sensitive or confidential data like usernames and passwords which may be used to conduct attacks against the registry.

Governance aspects of the document could cover topics such as:

- The general overview and the structure of the ccTLD
- Registration model(s): 3R (registry, registrar, and registrant), 2R (just registry and registrant), or other models

- Technical and administrative contacts of the registry
- Human resources, roles, responsibilities, and contacts of people involved in the registry technical decision making process
- A list of registrars with their respective contact information

Technical and operational aspects of the document could cover subjects such as:

- Number of primary and secondary authoritative name servers (NSes) with their respective IP addresses, TLD contact information (phone and email addresses), protocols in use between the registry and registrars such as the Extensible Provisioning Protocol (EPP), the software and hardware implementing the registry functions including, the registry database, Registration Data Access Protocol (RDAP) and/or Whois servers amongst other pieces of technical information
- User access and privilege inventory only to be strictly available to a limited number of authorized people
- Backups and restoration procedures
- Security: physical access, logs management, access controls, password management, firewalls, zone file integrity, and zone transfer security to name a few.
- Monitoring systems: hardware, software, zone synchronization (between NSes)
- Maintenance strategy
- Business continuity and disaster recovery plan

3.2 Audit the existing infrastructure

As previously stated, deploying DNSSEC adds a level of complexity to the existing DNS infrastructure and operations. Therefore, a good and safe practice would be to perform an audit against the current system infrastructure, operations, and processes either through an external or an internal party who possesses the requisite skills and autonomy to identify and share their findings and any gaps. It’s essential to fix any shortcomings in the current system before or in parallel prior to the signing of the zone. Doing so reduces the risk of things getting unmanageable following the DNSSEC implementation.

3.3 Write a DNSSEC Policy and Practice Statements

3.3.1 What is a DNSSEC Policy and DNSSEC Practice Statement?

There are several considerations to take into account and parameters to define when signing a domain, for example, DNSSEC signing algorithms, key sizes, the signature validity period, and the frequency of signature refreshes. In the case of a zone with a lot of delegations, a good practice is to fully document and keep up to date those sets of parameters applicable to the zone and avail them publicly. Therefore, two concepts should be taken into consideration here:

- The **DNSSEC Policy** (DP): Sets forth the security requirements and standards to be implemented for a DNSSEC signed zone. The DP constitutes a basis for an audit, an accreditation, or an assessment of an entity, for example a registry. Each entity can be assessed against one or more DPs that it claims to implement. In short, the DP states what needs to be done.
- The **DNSSEC Practice Statement** (DPS): This is an operational practice disclosure document that may support and be a supplemental document to the DNSSEC Policy (if it

exists). It states at a high level how a zone operator and its partners in the management of a zone, if any, implement procedures and controls to meet the requirements of the applicable DP. In contrast to a DP, the DPS states what is actually being done.

A DP provides general principles while a DPS provides a description of procedures and controls therefore, making it more detailed than a DP. On the other hand, policy is usually written by a policy authority (TLD manager or regulatory authority) and may be applicable to one or more zones in the DNS hierarchy while a practice statement specific to a single zone is written by the zone operator describing how it meets the requirements of a particular policy or a set of policies.

For example, in the context where both the administrative contact and technical contact of a ccTLD are different entities, the administrative contact could publish a policy outlining the standards and requirements to follow while also requiring the technical contact to publish a practice statement detailing how those standards and requirements will be met.

Alternatively, a zone operator or manager who is not governed by any external policy, may publish a DPS.

Publication of a DPS is most relevant for entities operating a zone that contains a significant number of delegations such as a TLD. Publishing a DPS helps provide a level of transparency which increases community trust in the TLD operations but as previously stated, a DPS should not contain sensitive operational information.

RFC 6841, *A Framework for DNSSEC Policies and DNSSEC Practice Statements,* is a document that provides a deep understanding of a comprehensive list of topics that a TLD operator should consider when defining both a DP and a DPS respectively.

3.3.2 How to write a DP and a DPS

Writing a DPS is an important step in the journey to signing a ccTLD. The DPS may be short and simple or long and complex, but it should assist people to understand the DNSSEC operations framework and how they can trust the process of signing the ccTLD.

The following table is a summary of the set of provisions consisting of eight components outlined in RFC 6841, which could be considered when drafting a DP or DPS. Not all components in RFC 6841 are mandatory to implement, and you are thus at liberty to pick those (sub)components suited to your needs.

Title	Description	Subcomponents
Introduction	Identifies and introduces the set of provisions, and indicates the types of entities and applications for which the policy or practice statement is targeted towards.	• Overview • Document name and identification • Community and applicability • Specification administration

Publication and Repositories	Describes the requirements for an entity to publish information regarding its practices, public keys, the current status of such keys together with details relating to the repositories in which the information is held.	• Repositories • Publication of public keys
Operational Requirements	Describes the operational requirements when operating a DNSSEC-signed zone.	• The meaning of domain names • Identification and authentication of a child zone manager • Registration of DS resource records • Methods to prove possession and ownership of a private key • Removal of DS resource records
Facility, Management, and Operational Controls	Describes non-technical security controls i.e., physical, procedural, and personnel to securely perform the DNSSEC related functions. Such controls include physical access, key management, disaster recovery, auditing, and archiving. These non-technical security controls are critical for trusting DNSSEC generated signatures.	• Physical controls • Procedural controls • Personnel controls • Audit logging procedures • Compromise and disaster recovery • Entity termination
Technical Security Controls	Defines the security measures taken to manage the cryptographic keys and activation data e.g., PIN numbers, passwords, or manually held key shares relevant to DNSSEC operations.	• Key pair generation and installation • Private key protection and cryptographic module engineering controls • Activation data
Zone Signing	Covers all aspects of zone signing, including the cryptographic specification surrounding the signing keys, the signing scheme, the methodology for key rollover, and the actual zone signing. Child zones and other relying parties may depend on the information in this section to understand the	• Key lengths, key types, and algorithms • Authenticated denial of existence • Signature format • Key rollover • Signature lifetime and re-signing frequency

	expected data in the signed zone and determine their own behavior.	
Compliance Audit	Describes how audits are to be conducted by the zone operator and possibly by other involved entities.	● The frequency of an entity compliance audit ● The identity/qualifications of the auditor ● Topics covered by the audit ● Post-audit actions
Legal Matters	Indicates under which jurisdiction the registry is being operated and provides references to any associated agreements that are in force. The Legal Matters section may inform of any identified implications on the protection of personally identifiable private information.	● Mention of the applicable jurisdiction ● Contractual obligations and compliance with national, transnational, or international laws and regulations ● Data protection and handling of personally identifiable information

Additional components may be added below this framework to address the specific needs of the ccTLD. Examples of DNSSEC Practice Statements are available in Annex A of this guidebook.

3.3.3 Selection of cryptographic algorithms for a zone

The field of cryptography evolves continuously. Newer algorithms replace existing ones when they are found to be less secure than previously believed. Therefore, algorithm implementation requirements and usage guidance are updated regularly to reflect new realities.

Implementing DNSSEC requires choosing an appropriate cryptographic algorithm. At the time of publishing this guidebook, RFC 8624, *Algorithm Implementation Requirements and Usage Guidance for DNSSEC* provides both algorithmic implementation guidelines and signing parameter requirements pertinent to DNSSEC.

The following table lists some (non-exhaustive) recommendations drawn from RFC 8624. Individual operators may have specific requirements and may want to adjust accordingly.

Item	**Recommendation**
DNSKEY Algorithm	**Algorithm 13 (ECDSAP256SHA256)** provides cryptographic strength and is currently recommended for use in new DNSSEC deployments due to its shorter keys and signature size, resulting in smaller DNS packets.

	However, **algorithm 8 (RSASHA256)** may also be used for it is widely deployed and has been the default algorithm for a number of years because of its cryptographic strength.
Delegation Signer (DS) Algorithms	SHA-256 is widely used and is a strong hash algorithm, and therefore recommended for new and existing deployments for the DS.
DNSSEC Security Algorithm (composed of the cryptographic algorithm and the hash algorithm)	The current recommendation is algorithm 13 (ECDSAP256SHA256). Otherwise, algorithm 8 (RSASHA256) could also be used.
Zone Signing Key (ZSK) and Key Signing Key (KSK) key size	Algorithm 13 (ECDSAP256SHA256) will always generate keys of 256 bit. The key size of algorithm 8 (RSASHA256) can be set between 2048 and 4096 bits.
ZSK and KSK effectivity period	There is no good way to estimate individual needs as operators adjust key effectivity periods based on their prior experiences. Several operators have been using a ZSK for one to three months and a KSK for one to five years before performing a rollover.
Private key storage	Off-line, non-network-connected, physically secure machines such as hardware security modules (HSMs)

Note: larger keys will increase the sizes of the RRSIG and DNSKEY records and will therefore increase the chance of DNS UDP packet overflow. Furthermore, the time it takes to validate and create resource record signatures (RRSIGs) increases with larger keys, so avoid needlessly increasing the key sizes.

3.3.4 Denial of existence: NSEC or NSEC3

Denial of existence or proof that something does not exist is a mechanism that informs a resolver that a certain domain name is non-existent (NXDOMAIN). Conversely, a domain name exists but it does not have the specific resource record (NODATA) being requested. Authenticated denial of existence uses cryptography to sign a negative response. In DNSSEC, this is either achieved using NSEC (Next Secure) or NSEC3 (Next Secure v3), respectively.

NSEC is used to describe an interval between names. It indirectly tells a resolver which names do not exist in a zone by providing in canonical order the name before it and the name after it. This mechanism implemented in NSEC forms the basis for authenticated denial of existence in DNSSEC and faces two issues:

- NSEC records are susceptible to zone walking, and this weakness can allow an attacker to traverse all the names in a zone. It is therefore possible to reconstruct the entire zone, and therefore defeat any attempts to administratively block zone transfers.

- The second issue facing NSEC in a delegation-centric zone such as a TLD, is that every name in that zone gets an NSEC record and its associated RRSIG. Once a zone is signed, this adversely leads to an increase in its size. The overheads generated by this increase could negatively impact the performance of the authoritative DNS servers such as constraining hardware storage resources or lengthening the duration to perform zone transfers.

NSEC3 on the contrary, mitigates the zone walking issue in NSEC, through the hashing of domain names with the possibility of hardening them using a salt feature. Furthermore, thanks to a specific feature called “opt-out,” unsigned domain names delegated in a zone (insecure delegations) do not require an NSEC3 record. This implies that when the opt-out feature is activated on a TLD zone, NSEC3 cannot prove or deny the existence of the unsigned domains registered under that TLD. Nevertheless, one pitfall exhibited by NSEC3 is that DNS responses are larger than those of NSEC.

There is no single appropriate answer when it comes to choosing between NSEC and NSEC3. If one prefers to use NSEC3 in order to prevent zone walking, implementing NSEC3 with no extra iterations and an empty salt is generally recommended. However, for smaller zones, the use of opt-out based NSEC3 records is not recommended. For very large and sparsely signed zones, in which the majority of the records are insecure delegations, the NSEC3 opt-out feature could be used. Other than these aforementioned considerations, NSEC is easier to troubleshoot than NSEC3.

3.4 Registrar involvement

Involving registrars at the preliminary stages of signing a ccTLD is highly recommended. Not only because they are the intermediary between the registry and the registrants, but deploying DNSSEC at the ccTLD level is the starting point for securing the namespace of that ccTLD. Once the ccTLD is signed, second level domain holders can start securing their respective domains. This will require registrars to be able to collect and send a new type of record from those second or subsequent level domain holders to the registry. This new type of record is called the Delegation Signer (DS).

Technically speaking, the DS is a hash of the Key Signing Key (KSK) and helps to establish the chain of trust between a parent zone and a child zone in the DNS name space. Having the DS record in the parent creates a secure linkage that an external attacker would have to overcome to forge key material in the child.

Typically, the ccTLD shares its DS record with the Internet Assigned Numbers Authority (IANA) for publication into the root zone. Registrants and administrators of domain names under the ccTLD share their DS records with the ccTLD directly or via a registrar. Registrars have two important roles to play here:

- **Provide a secure and reliable interface or mechanism to collect DS records** from the registrants; if such an interface or mechanism does not exist yet, registrars who are willing to offer DNSSEC support to their registrants should work as soon as possible to implement it. There is no standardized way to move the DS record between the customer and the registrar. Different registrars have different mechanisms, ranging from simple web interfaces to various APIs.

- **Pushing the DS to the registry**. An automated solution of publishing the DS into the parent zone is recommended instead of manual intervention. In the registry-registrar model, it is possible to use the DNSSEC extensions to EPP for transfer of DS resource record sets (RRsets) and optionally DNSKEY RRsets. In any case, tests should be conducted between the ccTLD operator and the registrars to ensure that new transactions are feasible using the existing infrastructure in place.

Another mechanism for a child's zone to automatically manage the DS with its parent is to use a CDS (Child DS) or a CDNSKEY (Child DNSKEY) RRset if the parent has an acceptance policy for these records. They can be used for the following three use cases:

- An initial DS publication
- A key rollover
- Returning to the insecure state

Put simply, the CDS/CDNSKEY is an instruction to the parent to modify the DS resource RRset if the CDS/CDNSKEY and DS differ. RFC 8078, *Managing DS Records from the Parent via CDS/CDNSKey* elaborates in detail on the automated management of DS records between the child and parent zones.

Lastly, involving registrars early in the process also allows them to benefit from DNSSEC training and hands-on programs delivered by ICANN and its partners within the Internet community.

4 Timeline

There are no specific timelines for DNSSEC deployment. The duration can range from a few weeks to months or years depending on several factors. Nonetheless, consider the following suggestions to avoid long delays:

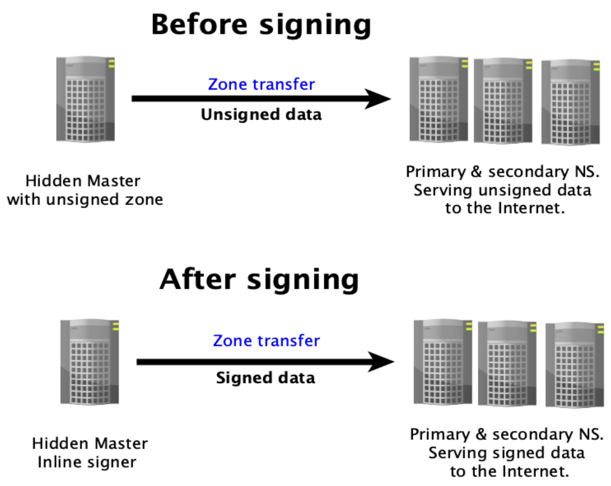
- Define the process as a project with a clear start date, an expected end date, and milestones to reach. In addition, assign a project manager or a technical lead with the appropriate resources
- Define, manage, and engage with stakeholders, including among others, the regulator(s), registrars, technical and administrative parties, contractors, backend operator(s) or any other party. Communication among the different ccTLD stakeholders also needs to be addressed
- Identify and properly manage risks

5 DNSSEC deployment scenarios

Whether you decide to manage the zone solely using your own resources and infrastructure or through hiring a contractor such as a backend operator, it is likely that the technical deployment architecture follows one of the two major scenarios described here.

5.1 Primary Server for in-line signing (hidden primary)

In this configuration scenario, a hidden primary name server usually unknown to the Internet, serves the zone to a set of DNS authoritative servers, usually to one public primary and a number of secondary servers. A hidden primary name server is not a specific DNSSEC requirement, but rather a DNS operation best practice that suggests having an out-of-band authoritative name server that is inaccessible and unknown to the public and where all the zone updates can be done. This server should also implement tighter security and audit procedures. The architecture resembles the figure below:

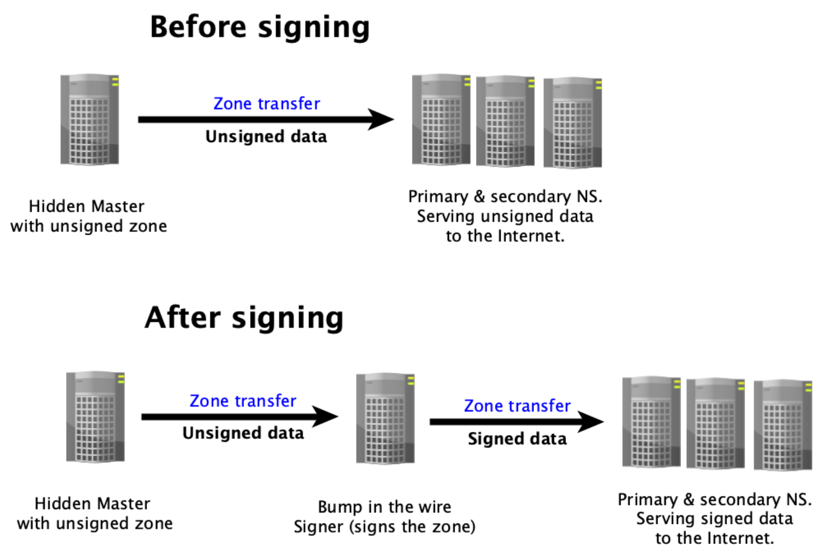

Before signing
Zone transfer
Unsigned data
Hidden Master with unsigned zone
Primary & secondary NS. Serving unsigned data to the Internet.
After signing
Zone transfer
Signed data
Hidden Master Inline signer
Primary & secondary NS. Serving signed data to the Internet.

This hidden primary server will be set up to recognize the created keys, and use them to generate and run a signed zone, following the process provided by the DNS software it runs. On completion, the signed version of the zone file will be transferred and kept synchronized with all appropriate publicly visible authoritative name servers.

Except for the configuration changes in the hidden primary name server, there are no further additional configuration or software changes to perform in this architecture.

5.2 Bump-in-the-wire in-line signing

In this configuration scenario, a new name server, the signer, is inserted into the existing architecture, and placed between the hidden primary server and the public name servers serving the zone to the Internet. This new server acts as a bump-in-the-wire (BITW) device. It takes the unsigned zone file from the hidden primary, signs the data and sends out the signed zone file to be distributed to the public name servers of the domain.


Before signing
Zone transfer
Unsigned data
Hidden Master with unsigned zone
Primary & secondary NS. Serving unsigned data to the Internet.
After signing
Zone transfer
Unsigned data
Zone transfer
Signed data
Hidden Master with unsigned zone
Bump in the wire Signer (signs the zone)
Primary & secondary NS. Serving signed data to the Internet.

To achieve this, the following steps may be considered:

1. The hidden primary should not be listed in the name server (NS) resource RRset for the domain to avoid getting conflicting answers i.e., unsigned answers from the hidden primary and signed answers from the other name servers.
2. The hidden primary server configuration should be updated to allow only the signer to perform a zone transfer.
3. The signer uses the unsigned zone file received from the hidden primary server and a private key to sign the resource records. Finally, it distributes the signed zone file to the primary and secondary name servers using zone transfer mechanisms.
4. The name servers' configuration should also be updated accordingly to receive zone transfer only from the signer.

Regardless of the deployment scenario, verification of the signed zone is advisable before distributing it across all servers. This verification exercise can be conducted on the domain name, using online tools such as those described in Section 10 of this document. It is also good practice to run several queries searching for DNSSEC resource record signatures and signature expiration dates, among other parameters on each of the zones being administered. Include these test runs as part of validating your implementation.

In summary, let us compare the steps in the zone production chain between normal DNS versus DNSSEC:

- **Normal DNS**: Create → Validate → Publish → Monitor
- **With DNSSEC**: Create → Validate → Sign → Validate → Publish → Monitor

6 Signing a TLD

As with any other major change, it is strongly advisable to conduct a deep testing phase before planning a deployment in the production system. It is even more necessary in the case of DNSSEC, because new changes are introduced into the zone. The steps described below provide a general overview for signing a TLD zone but, based on your environment and specific requirements, you may need to adjust them.

In some cases, signing the whole zone file at once would be inappropriate for large ccTLDs; instead, it would be safer to establish an incremental approach to signing the zone. Other ccTLDs may also opt for signing a child zone on a test bed first, before adding a DS record into the parent zone. This is a preparatory test prior to signing the actual ccTLD zone.

In any case, to successfully complete the signing process, it would be wise to proceed with caution by adopting appropriate deployment and test plans accompanied by a rigorous validation methodology.

1. Deploy and configure a DNSSEC testbed environment. Depending on the deployment model, the testbed could contain the following elements:
    - One test signing server, a name server (test NS), which is capable of signing a zone file. The server should be able to either generate the signing keys or receive keys generated from a different server or from a Hardware Security Module (HSM), sign the zone and distribute the signed zone file among other functions.
    - One test secondary authoritative name server will receive the signed zone and serve it.
    - One test resolver should be configured to perform DNSSEC validations locally for the signed zone.
2. Copy the unsigned zone file from the hidden primary server into the test signing server. The distribution of the unsigned zone file to the test signing system could be automated later on in the testing phase.
3. Generate the KSK and ZSK keys. It is recommended to generate the keys out of the signing system. Store the KSK on a HSM or off-line and use it just to sign the DNSKEY resource records.
4. Sign the zone and publish it to the test secondary name server(s).
5. Generate and import the DS as trusted keys into the test resolver. In the real world, the DS will not be distributed to recursive resolvers around the world directly by the TLD administrators; rather it will be sent to IANA for publishing into the root zone. Any validating recursive resolver around the world will then fetch the DS corresponding to that TLD from the root zone.
6. Run some tests by performing User Acceptance Testing (UAT) based on defined test cases. The tests should cover topics such as retrieving the keys and signatures, checking the signatures expiration, query response time, and size. Perform DNSSEC validation among other tasks.
7. If the test signing process is automated, observe signature expiration and the automatic generation of new signatures. Otherwise, you should perform manual rollovers until the test signing process is automated.
8. Perform ZSK rollover and KSK rollover. For a KSK rollover, generate the new DS and simulate sharing with the parent zone by adding it to the test resolver for local validation. Depending on various timing parameters such as the key effectivity period in the test environment, you will need to remove the old DS from the test resolver and the old KSK from the zone to complete the KSK rollover.
9. Iteratively conduct new tests and refine steps 1 to 8.
10. Once confident with the testing methodology, focus on going live and choosing the appropriate environment and deployment scenario of choice: BITW or hidden primary in-line signing. Conduct a new UAT to confirm that all the domain authoritative name servers correctly serve the zone and its corresponding DNSSEC resource records.
11. Finally, publish the DS to the root zone following the delegation management guidelines for TLDs as defined by IANA at https://www.iana.org/domains/root/manage. Once added to the root zone, it announces to the world that the ccTLD is DNSSEC signed and that

any security aware resolver should perform DNSSEC validation against the DNS records originating from that zone. It is highly recommended to use a tool like https://dnsviz.net/ (see Section 10 for more details) to actually validate the DNSSEC chain of trust for the zone.

12. Once the ccTLD is "officially signed," plan to open access for registrars to publish registrants' DS records into the registry in order for the chain of trust to be effective.

7 Key and algorithm rollover

When a zone is secured with DNSSEC, the zone manager must be prepared to replace (or "roll over") the keys used to secure the zone, whether it is done periodically for security or for operational concerns or in the event of an emergency. To implement a key or algorithm rollover, new keys need to be introduced and old keys discarded from the zone. It is critical to take into account that data published for a zone lives in various resolver caches. Ignoring data that may be in caches could lead to loss of service for clients. For example, consider zone data signed with an old key, which is being validated by a resolver that does not have the old zone key in its cache. If the old key is no longer present in the current zone, validation will fail and the corresponding zone data will be marked as bogus.

On the other hand, a resolver trying to validate data which is signed with a new key while the old key still lives in the resolver cache, also results in data being marked as bogus. Various types of key and algorithmic rollover techniques such those described in RFC 6781, *DNSSEC Operational Practices, Version 2* as well as RFC 7583, *DNSSEC Key Rollover Timing Considerations* exist. Examples of these techniques are *pre-publish*, *double-RRSIG ZSK rollover*, *double-KSK, double-DS, and double-RRset,* to name a few.

In the specific case in which an emergency key rollover is required because a ZSK or KSK key pair is suspected of being compromised, it is advisable to already have a documented procedure in place.

7.1 ZSK rollover

During a ZSK rollover, it is essential to ensure that any caching validator that has access to a particular signature also has access to the corresponding valid ZSK. RFC 6781, *DNSSEC Operational Practices, Version 2* documents three methods of conducting a ZSK rollover: pre-publication, double-signature, and double-RRSIG.

In this document, we will describe the pre-publication method only, as it keeps the zone and response sizes to a minimum during the whole rollover process. In this method, the new ZSK is introduced into the DNSKEY RRset and after sufficient time has elapsed to ensure that any cached DNSKEY RRsets contain both keys. The zone is then signed using the new ZSK and the old signatures are removed. Finally, when all signatures created with the old ZSK have expired from caches, the old key is removed. The steps below describe the process.

1. The new ZSK A is introduced into the zone and appears in the DNSKEY RRset, but will not yet be used to sign records in the zone. The current active KSKs resigns the DNSKEY RRset, which is then re-signed with the currently active KSKs. At this stage, the new ZSK is said to be published.

2. After a certain period of time, ZSK A becomes ready to sign records in the zone. This period corresponds to the zone propagation delay plus the time to live for the DNSKEY records in the zone. In other words, it is the maximum time taken for existing DNSKEY records to expire from caches. ZSK A becomes active and effectively starts signing records for the zone.
3. ZSK A will continue signing and refreshing records for the zone until a point of time when a new ZSK B needs to be published. The publication time of key B depends on the activation time of A and the ZSK lifetime set for the zone in the key management policy. ZSK B gets ready and could be used to sign records, but ZSK A is still active.
4. When the ZSK lifetime is reached for key A, it is retired. Key B becomes active and is used to sign the zone. However, the retired key must be retained in the zone for some time (for a "retire interval") to allow the RRSIG to be generated using that key to continue being validated by resolvers. The retire interval corresponds to the time needed for all existing RRsets to be re-signed with the key B, plus the zone propagation delay, and the maximum TTL of all the RRSIG created with the old key in the zone.
5. After a certain period of time, the signatures created with the retired key disappear from resolver caches, and the old key is said to be dead.
6. Once the old key is dead, it can be removed from the DNSKEY RRset, which must be re-signed with the current zone's KSKs. At this stage, key A is declared removed.
7. After some time, a new key will be published, and the whole process will be repeated.

7.2 KSK rollover

In a KSK rollover, the main challenge lies in ensuring that a trusted KSK exists for the zone at all times even during the rollover process. RFC 6781, *DNSSEC Operational Practices, Version 2* also documents three methods of rolling a KSK: double-KSK, double-DS, and double-RRset. The Double-RRset method is the most efficient of these, as both the new DS records and DNSKEY RRsets propagate in parallel.

In this method, the new DNSKEY and DS records are published simultaneously in the appropriate zones. Once enough time has elapsed for the old DNSKEY and DS RRsets to expire from caches, they are removed from their respective zones. The rollover steps are described as follows:

1. The DS and DNSKEY records are present in their respective zones. Their corresponding KSK A is active and securing the zone.
2. Once the current KSK A lifetime approaches, a new KSK B is introduced in the zone and used to sign the DNSKEY RRset. The DS of the KSK B is sent to the parent for publication in the parent zone.
3. The parent could proceed with verifying the new DS and then publishing it into the parent zone.
4. After a period of time has transpired, the new DS or DNSKEY have already propagated into the caches of validating resolvers. At the same time, ZSK A is removed from the zone.
5. Later, the DS and DNSKEY records associated with ZSK A are also removed.
6. After some period of time, a new key will be published, and the whole process will be repeated.

8 Other considerations for signed zones

It is worth considering the following elements in preparation for a DNSSEC deployment strategy:

- **Define a capacity development program**: Identify and participate in workshops, webinars, hands-on training and any other capacity building activities that could help increase knowledge and develop new skills in DNSSEC operations. ICANN's Technical Engagement delivers such trainings including DNSSEC workshops, which usually take place during ICANN meetings.
    - Besides ICANN, the Network Startup Resource Center (NSRC), the Internet Society and Regional Internet Registries (RIRs) also provide engagement activities related to DNSSEC.
    - Lastly, participating in forums, webinars and workshops in which both experienced operators and newcomers meet, present, and discuss current and future DNSSEC deployments, can greatly enhance your knowhow of DNSSEC operational practices.
- **Subscribe to NOG mailing lists**: These are good forums where people discuss and share their technical experiences and expertise as well as seek support and assistance in technical matters. So think of it as a community that could assist you when needed.
- **Key generation and management:** Hardware Security Modules (HSMs) tend to provide a good facility for generating and storing private keys. However, the costs of buying, securing and maintaining the HSM are worth considering. Depending on its features, the costs of HSMs may vary from a few hundred to thousands of dollars. HSMs can also add training overheads, as learning any new hardware comes with its challenges. Using HSMs is a good practice but it is not the only method for generating keys. Another possibility worth considering is to generate, store and use the private keys in an off-line, non-network-connected, physically secure machine. RFC 6781, *DNSSEC Operational Practices, Version 2* provides more details on key generation and management.
- **Time considerations**: DNSSEC introduces the notion of absolute time in the DNS. Signatures in DNSSEC have a validity period from their inception date to their expiration date after which the signature is marked as invalid and the signed data is considered bogus. It is critical to ensure that time is managed well so that signatures are generated with the correct validity period. Imagine a signed zone whose signature's validity period has transpired; this will result in validation failing at the resolver end. Thus, it is strongly recommended to configure the Network Time Protocol (NTP) server in order to keep accurate time. There are also other considerations like the minimum and maximum zone time to live (TTL), signature publication period and signature validity period as described in RFC 6781, *DNSSEC Operational Practices, Version 2*.
- **Software, hardware and network requirements**: Both open source and commercial DNSSEC implementations are currently supported including Berkeley Internet Name Domain (BIND), PowerDNS, NLnet Labs Name Server Daemon (NSD), and Knot DNS to name a few. OpenDNSSEC is a signing solution that remains widely used as it automates the process of keeping track of DNSSEC keys and the signing of zones. If your plan is to deploy DNSSEC on an authoritative server, you will need to generate the cryptographic signing keys on the system. The amount of time required to generate the keys depends on the source of randomness (entropy) in the system. Systems such as virtual machines with insufficient entropy may take a much longer time to generate keys.
    - Hardware resources such as the CPU, system storage, and memory are also areas worth considering for possible optimizations. This is because enabling DNSSEC increases system storage, memory usage, and CPU load in part due to key

generation and signing. A signed zone is always accompanied by a sizable increase in the zone file.

- Concerning network security policies, verify that firewall and ACL rules for example permit both large DNS UDP packets and DNS over TCP on port 53. In addition, the Extension Mechanism for DNS (EDNS0) needs to be activated on both the DNS servers and in the network configuration, respectively.

9 Unsigning the TLD if necessary

DNSSEC, like many things in this world, is not without problems. When adding more complexity into the DNS, it increases the likelihood that things can break or go wrong. For instance, either the KSK or ZSK or both may get lost or compromised. An unanticipated hardware or software bug could prevent a zone from being signed and distributed, therefore, affecting a zone from being properly served.

In the worst case scenario, you may prefer to unsign the zone in order to fix any issues and bugs before signing again. However, unsigning a domain incurs the cost of reverting it to an insecure state.

The world knows whether a zone is signed or unsigned by the presence of a DS record in the parent zone. If no DS record exists, the trust chain is not assured. Therefore, reverting to an unsigned state is technically as easy as removing all DS records from the parent zone. In the case of a ccTLD, this implies requesting IANA to remove the corresponding DS record(s) from the root zone.

Upon resolving all issues, the TLD operator should consider generating new keys and re-signing the zone. Furthermore, the operator should ensure that the newly signed zone is well distributed and available in all the name servers before publishing a new DS record into the root zone. Once the DS is published by IANA, all of the Internet becomes aware that the TLD zone has once again been signed and that any validating resolver will verify all resource records served by that zone.

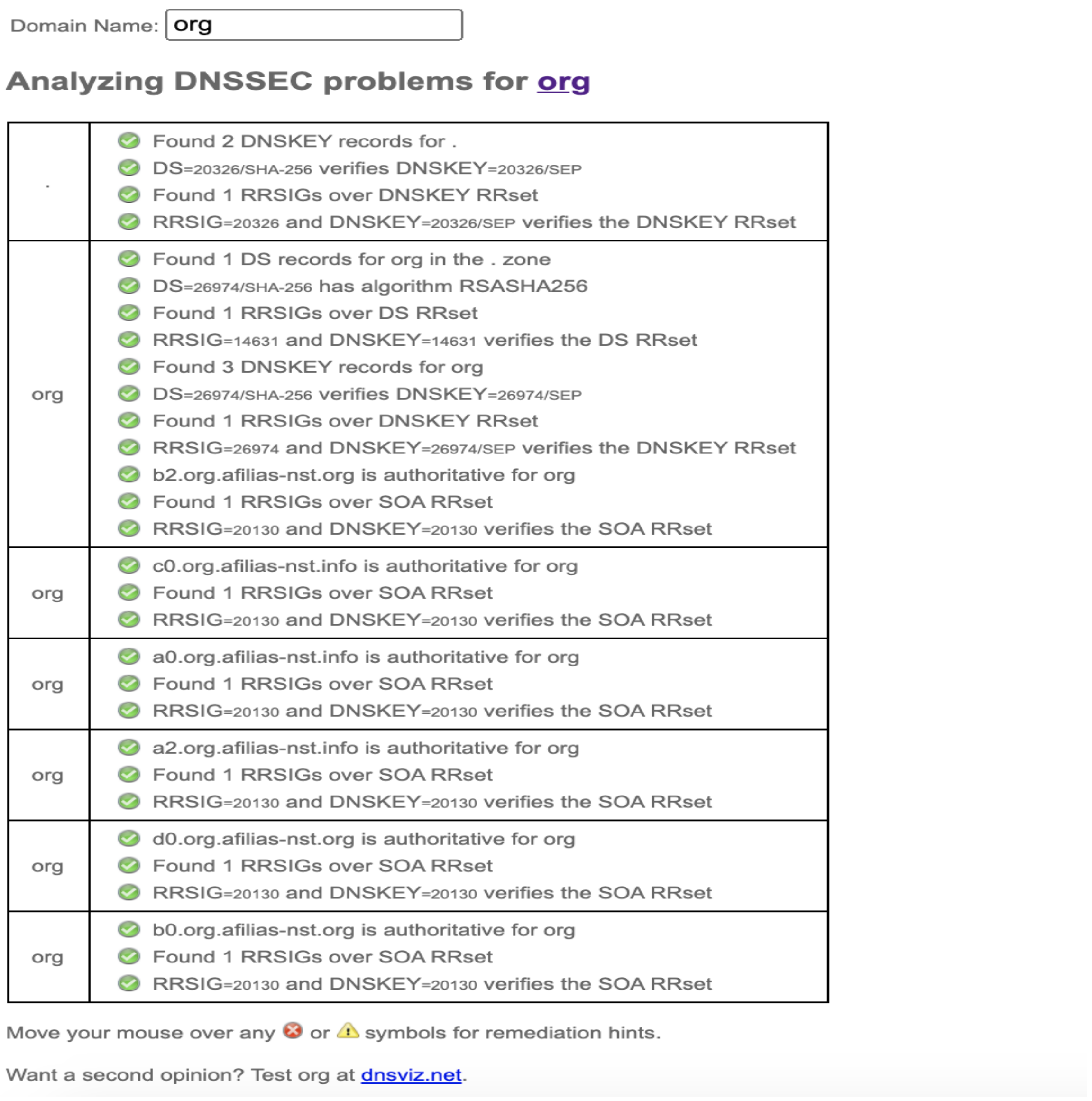
10 Useful DNSSEC tools

The following tools can be useful to troubleshoot DNSSEC issues.

10.1 Verisign DNSSEC debugger

This DNSSEC debugger, found at https://dnssec-debugger.verisignlabs.com/, is a web-based tool, which helps to ensure that the "*chain of trust*" is intact for a particular DNSSEC-enabled domain name. It displays a step-by-step validation of a given domain name and highlights any problems found.

Below is an example of what the output looks like:


Domain Name: org

Analyzing DNSSEC problems for org

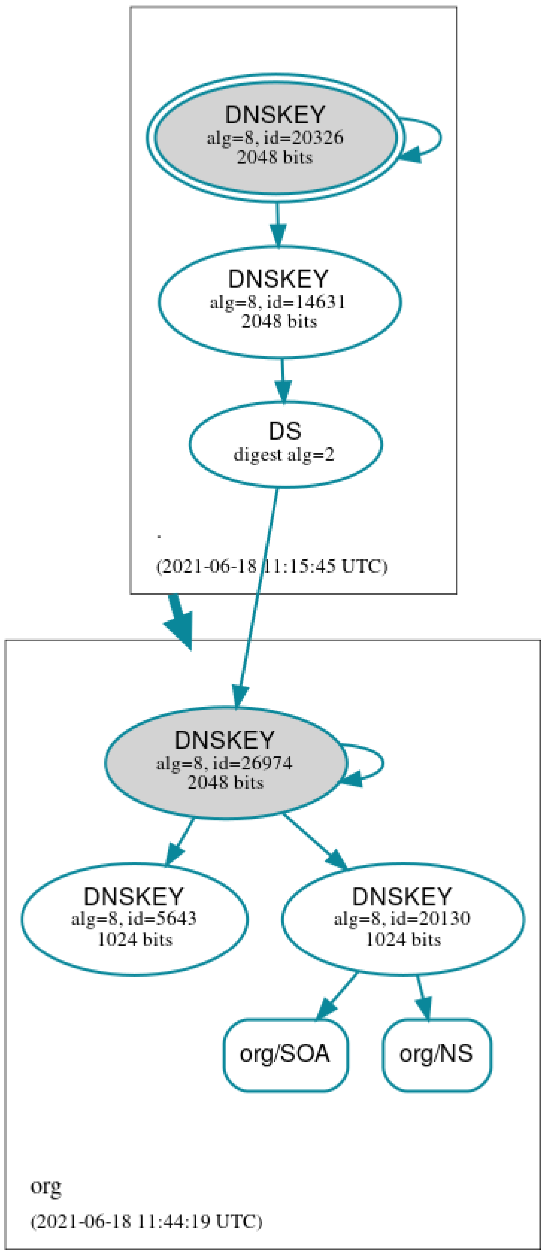
.	✓ Found 2 DNSKEY records for . ✓ DS=20326/SHA-256 verifies DNSKEY=20326/SEP ✓ Found 1 RRSIGs over DNSKEY RRset ✓ RRSIG=20326 and DNSKEY=20326/SEP verifies the DNSKEY RRset
org	✓ Found 1 DS records for org in the . zone ✓ DS=26974/SHA-256 has algorithm RSASHA256 ✓ Found 1 RRSIGs over DS RRset ✓ RRSIG=14631 and DNSKEY=14631 verifies the DS RRset ✓ Found 3 DNSKEY records for org ✓ DS=26974/SHA-256 verifies DNSKEY=26974/SEP ✓ Found 1 RRSIGs over DNSKEY RRset ✓ RRSIG=26974 and DNSKEY=26974/SEP verifies the DNSKEY RRset ✓ b2.org.afilias-nst.org is authoritative for org ✓ Found 1 RRSIGs over SOA RRset ✓ RRSIG=20130 and DNSKEY=20130 verifies the SOA RRset
org	✓ c0.org.afilias-nst.info is authoritative for org ✓ Found 1 RRSIGs over SOA RRset ✓ RRSIG=20130 and DNSKEY=20130 verifies the SOA RRset
org	✓ a0.org.afilias-nst.info is authoritative for org ✓ Found 1 RRSIGs over SOA RRset ✓ RRSIG=20130 and DNSKEY=20130 verifies the SOA RRset
org	✓ a2.org.afilias-nst.info is authoritative for org ✓ Found 1 RRSIGs over SOA RRset ✓ RRSIG=20130 and DNSKEY=20130 verifies the SOA RRset
org	✓ d0.org.afilias-nst.org is authoritative for org ✓ Found 1 RRSIGs over SOA RRset ✓ RRSIG=20130 and DNSKEY=20130 verifies the SOA RRset
org	✓ b0.org.afilias-nst.org is authoritative for org ✓ Found 1 RRSIGs over SOA RRset ✓ RRSIG=20130 and DNSKEY=20130 verifies the SOA RRset

Move your mouse over any ✗ or ⚠ symbols for remediation hints.

Want a second opinion? Test org at dnsviz.net.

10.2 DNSVIZ

DNSViz (https://dnsviz.net) is a tool for visualizing the status of a DNS zone. It provides a visual analysis of the DNSSEC authentication chain for a domain name and its resolution path in the DNS namespace. It also lists configuration errors detected by the tool. Below is a visualization of the .ORG zone:



11 Conclusion

DNSSEC is a robust protocol, which provides authentication and integrity of the DNS data. Signing a domain name with DNSSEC changes a lot of things at an operational level by introducing new concepts and tasks that did not exist with normal DNS.

This document is a guide to help ccTLD registry operators and any other party to understand what constitutes signing a ccTLD. There is much to talk about regarding DNSSEC, and this document covered some of the most important aspects of a deployment.

Like any other security solution, it is advisable to follow a proper process and get prepared to avoid breaking things. All parties must evaluate their own environment and the associated threats and vulnerabilities to determine the level of risk they are willing to accept when relying on DNSSEC to protect their zone and the domains in them.

Essentially, coordinated efforts and active collaboration from all stakeholders remains key to the successful deployment of DNSSEC.

A Example DNSSEC Policies and DNSSEC Practice Statements

- ZACR DNSSEC Policy and Practice Statement Framework, version 001, Sept. 2016, ZACR: https://www.registry.net.za/downloads/u/zacr-dps-signed.pdf
- DNSSEC Practice Statement .fr, version 1.2, June 2013, Afnic: https://www.afnic.fr/wp-media/uploads/2020/12/dps-english-fr.pdf
- CIRA DNSSEC Practice Statement for .CA, version 1.5, August 2016, CIRA: https://www.cira.ca/cira-dnssec-practice-statement-ca
- DNSSEC Practice Statement for the JP Zone (JP DPS), version 1.4, Oct. 2015, JPRS: https://jprs.jp/doc/dnssec/jp-dps-eng.v1.4.html
- Verisign DNSSEC Practice Statement for TLD/GTLD Zone, version 1.8, dec. 2019, Verisign Inc.: https://www.verisign.com/assets/20191111_CTLD_VerisignDNSSECPracticeStatement_v1.8_finalized.pdf. The list of zones to which this applies can be found at https://www.verisign.com/assets/20190430-Verisign-Operated-TLD-GTLD-Zones-v1.04-Converted.pdf

B Example of unsigned and signed zone

B.1 Unsigned zone

```
example.                    86400  IN    SOA   a.nic.dns.blablabla.
hostmaster.dns.blablabla. (
                2017072300 ; serial
                1800       ; refresh (30 minutes)
                900        ; retry (15 minutes)
                2419200    ; expire (4 weeks)
                300        ; minimum (5 minutes)
                )

example.                    86400  IN    NS    a.nic.dns.blablabla.
example.                    86400  IN    NS    b.nic.dns.blablabla.
example.                    86400  IN    NS    d.nic.dns.blablabla.
example.                    86400  IN    NS    e.nic.dns.blablabla.
example.                    86400  IN    NS    f.nic.dns.blablabla.
aaa.example.                86400  IN    NS    ns1.reg.zzzz.
aaa.example.                86400  IN    NS    ns2.reg.zzzz.
```

```
bbb.example.                86400  IN     NS     ns1.reg.zzzz.
bbb.example.                86400  IN     NS     ns2.reg.zzzz.
ccc.example.                86400  IN     NS     ns1.reg.zzzz.
ccc.example.                86400  IN     NS     ns2.reg.zzzz.
ddd.example.                86400  IN     NS     ns3-l2.nic.zzzz.
```

B.2 Signed zone

```
example.                    86400  IN     SOA    a.nic.dns.blablabla.
hostmaster.dns.blablabla. (
                2017072305 ; serial
                1800       ; refresh (30 minutes)
                900        ; retry (15 minutes)
                2419200    ; expire (4 weeks)
                300        ; minimum (5 minutes)
                )

example.                    86400  IN     RRSIG  SOA 8 1 86400 20170724231821
20170618015310 660 example. nQ8H8StSRDoQgzwBNQ0k9+E1LGrV0tsCinoB6KxcyuHfGT4ehWsj5JI6
N01WpXqy/q1S/XlhqtjVoiti4zSOwIjFlSloug3W09eJnH9biwmb6U8B
JQoHf3edGvZtWNZdtcOKY1CFBI2ApceFn8KOYvT0qzpygOlF5lMrJvnO J5c=
example.                    86400  IN     NS     a.nic.dns.blablabla.
example.                    86400  IN     NS     b.nic.dns.blablabla.
example.                    86400  IN     NS     d.nic.dns.blablabla.
example.                    86400  IN     NS     e.nic.dns.blablabla.
example.                    86400  IN     NS     f.nic.dns.blablabla.
example.                    86400  IN     RRSIG  NS 8 1 86400 20170730192856
20170617025305 660 example. KNaF2jTPuCGq5FIzspbJL+TDBx/6z01E7+tkkzYRNh0xAKDnutcfb1It
D7XrNWPEbXsaafFyZ/M5DaDGzTzsVNm1h9h3md6o0vZNH07q8nmm+fYX
do8sx9aFxCgl9NsmG0cyrBbVnyrPKxDlAx69HJCh0kBb7PFKhr1hpnYY xGA=
example.                    86400  IN     DNSKEY 256 3 8
AwEAAasHjitdDurpevNLojD4Sp3609P+C9uOTR42DJel0NSSva/x38Ba
7gs0b4Q+tmKPI5cmxDhECiUfdzaRRA8vxPZK8x5LL/V1WZ5q6egFmH4x
eLxWaxlftFotev/T8kVe7jZUk7Hh3x7LPgGLajpjNNFELj42Xe6XBkkN 9FY1lQkB
example.                    86400  IN     DNSKEY 257 3 8
AwEAAY6HLDY5M5kjlrjVV9HQyWUkkrYZ2eB8KeJjUMN9qDM6FsA57pbS
5tmbGV1zxxqGon0pO7HYV06GZIgFOLBqDvgGsnKDQ5A2iktYNuSmTh+w
fd8ixgbYigtoBMBnNeqFozMK58c1yf7amui2cCOg9ibGZMpLQvjKOSyV
Jnlh0l8e3OE7Ul1GEa39XpVez2wkjImhsG0e7KAZPlFjEUpvwie8HEQV
jz3PK7Zr6SZVLLyet0rnN3prCHfvhNh6DycN/rt6/PopLvPQM8SaW+u8
zn6Z4S4AoTPTxKm5udzb7mWf7lT83PAbOvLu/WIRY6nqye+4SkJsrmjI xnLdk/Q54E8=
example.                    86400  IN     RRSIG  DNSKEY 8 1 86400 20170703000000
20170613000000 54322 example. B2riGYos+/q5RqXVBQKrrkVUuruDBH8ANNa8J6sMHUjf0MPZOuICd2kZ
PlAGMpZpp8LoaRoG2zaTVILZ8Vhi90FsyLsZVpPooAvmKlTFOrWoJoPo
XScLhb3ISRLOzKEnyLt5Ds3TxuabHLPlf8jpTXaHMFZCzYYtTJJQb+M3
BLeK+Lx4uCwU1pvxNkuR9StKa5tJquByIZCWZsSx5nKWPyrsGLtFJKrg
DXe8XlA8LxeER69OQgSZ1VXvK8Kd4p3wyvzUHCcsPYZzebxHXPqDrYB7
BU7eqsDUjCfThqbkC0Ju7koHROYRjGdoY/4f6nDOJEOICIFEgEDHJg2t w1nENQ==
example.                    0      IN     NSEC3PARAM 1 0 3 00FF
example.                    0      IN     RRSIG  NSEC3PARAM 8 1 0 20170716213640
20170606005304 660 example. a6Mp1NjW2/nnn+5i98AWzVrOX0yUvu/urP1cqY6zZjISReZOSLx6aorJ
lM9Nnx1fNVr2COtD71UVJI7kFUC5jVbmAitWdHHH/zyzK6WyyaN5Nsaf
cKW0Su8lLkctCHiqpKmuhOhnK1Dqmigx8YhyhPbN5nCzoST6lcnNjtV0 TwQ=
aaa.example.        86400  IN     NS     ns1.reg.zzzz.
aaa.example.        86400  IN     NS     ns2.reg.zzzz.
```

```
bbb.example.        86400  IN    NS    ns1.reg.zzzz.
bbb.example.        86400  IN    NS    ns2.reg.zzzz.
ccc.example.        86400  IN    NS    ns1.reg.zzzz.
ccc.example.        86400  IN    NS    ns2.reg.zzzz.
ddd.example.        86400  IN    NS    ns3-l2.nic.zzzz.
0KPQJ71AL5RHRST9HM8LEFLK0I0QN5N7.example.     3600 IN      NSEC3 1 1 3 00FF
464L7A368JEOCPKU9G34B9RQADEPKA14 NS DS RRSIG
0KPQJ71AL5RHRST9HM8LEFLK0I0QN5N7.example.     3600 IN      RRSIG NSEC3 8 2 3600
20170703210235 20170602012306 660 example.
H+qdaHqnAgUa66VSKmMmfKWopeZQM0ridMUN2YN4rncHeWD8b0yA6O6N
hLF/ojpZoGrQN+G+p4SWJVb/pj2CkLk00E2AhloXXV0KaQIzUwPVNm7p
J9es7ohi5ErGtM1ClLpGggz05qNWboejbrXtS8TFdoTtn6Z2Omk4RNmj hG0=
464L7A368JEOCPKU9G34B9RQADEPKA14.example.     3600 IN      NSEC3 1 1 3 00FF
MLTMB5J4Q7T5R3GJBSBTMVD2LBMFU3KA NS DS RRSIG
464L7A368JEOCPKU9G34B9RQADEPKA14.example.     3600 IN      RRSIG NSEC3 8 2 3600
20170715005821 20170610062309 660 example.
dk6WScB3zmJYig0w8LxFXoc9vj1leqFRBlET4YAVVmeAwcGf0ixa41T+
pKKcMHbXDsw+PHYZHaRLmA9lEgs+4lJMdA3fRroNSXyV2usHMdFaKUoG
UZKehVGdgrBRx4vx+o4w1ztdumY6MsD0ART6IrhUbr+cvGHAlxNSviCI BbE=
MLTMB5J4Q7T5R3GJBSBTMVD2LBMFU3KA.example.     3600 IN      NSEC3 1 1 3 00FF
0KPQJ71AL5RHRST9HM8LEFLK0I0QN5N7 NS SOA RRSIG DNSKEY NSEC3PARAM
MLTMB5J4Q7T5R3GJBSBTMVD2LBMFU3KA.example.     3600 IN      RRSIG NSEC3 8 2 3600
20170706043605 20170604225320 660 example.
Ndq6p+Y8ztlgNN1vH12o5rxxh7QM8GLY3E1FPCX4h7N4RtnuoPpvEpsl
/K4XQ1p/8Uehe6Izg0BpvQ7A256/+UW3lkwlonR7UaOX/+gkEdxuxlC/
41nX5fI9G5QFrV7H8B7ezlVF/uLz4nXyH4mzz496x4iTMEoHfoAdMinL C7A=
example.                    86400  IN     SOA    a.nic.dns.blablabla.
hostmaster.dns.blablabla. 2017072305 86400 14400 2592000 3600
```

C DNSSEC deployment checklist

C.1 Initiation and preparation

- ❑ Define the DNSSEC deployment as a project

  Managing the whole process as a project with a start date, a targeted end date, and clear deliverables using a project management approach

- ❑ Document the existing system.

  An up-to-date documentation describing the system and the processes in use

- ❑ Audit the existing infrastructure

  Fixing any shortcomings in the current system prior to the DNSSEC deployment

- ❑ Involving stakeholders

  Preparing their understanding and readiness of DNSSEC deployment

- ❑ Define and follow a training plan

  Provide your staff and stakeholders with the necessary knowledge and skills for DNSSEC deployment

C.2 Deployment and monitoring

- ❏ Write a DP or DPS

  Publishing the DNSSEC operations framework applicable in the zone
- ❏ Write a DNSSEC deployment plan

  Documenting the global deployment strategy and steps. Several items of the checklist can be addressed in this document.
- ❏ Write and validate the DNSSEC processes and operations procedures

  Documenting the procedures for your requirements and environment
- ❏ Choose a DNSSEC deployment scenario

  Identifying the DNSSEC implementation model
- ❏ Identify and purchase new material and equipment

  Acquiring new equipment and material
- ❏ Select DNSSEC signing parameters

  Assigning values to a set of DNSSEC parameters
- ❏ Build a DNSSEC test bed

  Writing, running and validating test cases for the zone signing to be familiar with DNSSEC processes and operations
- ❏ Plan to go-live and prepare a fallback

  Preparing and documenting the go-live in the production environment as well as monitoring and fall back procedures
- ❏ Go-live and monitor

  Implementing and monitoring the DNSSEC signing in the production environment
- ❏ Plan for registrants' DS publication

  Promoting DNSSEC, preparing to receive and process sub-domains DS records
- ❏ Publish registrants DS

  Announcing sub-domains DS in the ccTLD zone
- ❏ Document the lessons learned

  Compiling the lessons and experiences learned at each step in the process

D Further reading

- *Major DNSSEC Outages and Validation Failures*, IANIX: https://ianix.com/pub/dnssec-outages.html
- *DNSSEC: The long and bumpy road of algorithm deployment*, APNIC, https://blog.apnic.net/2020/12/01/dnssec-the-long-and-bumpy-road-of-algorithm-deployment/
- *DNSSEC Infrastructure Audit Framework*, NLnet Labs, https://nlnetlabs.nl/downloads/publications/dns-audit-framework-1.0.pdf

- *Root DNSSEC information*, IANA, https://www.iana.org/dnssec
- *Frequently Asked Questions about DNSSEC*, SIDN, https://www.sidn.nl/en/faq/dnssec
- RFC 6781, *DNSSEC Operational Practices v2,* https://www.rfc-editor.org/rfc/rfc6781.html

1. RFC 6841, *A Framework for DNSSEC Policies and DNSSEC Practice Statements,* https://www.rfc-editor.org/rfc/rfc6841.html
2. RFC 8078, *Managing DS Records from the Parent via CDS/CDNSKey,* https://www.rfc-editor.org/rfc/rfc8078.html
3. RFC 8624, *Algorithm Implementation Requirements and Usage Guidance for DNSSEC,* https://www.rfc-editor.org/rfc/rfc8624.html