Workgroup: dnsop
Internet-Draft:
draft-ietf-dnsop-dnssec-validator-
requirements-04
Published: 25 January 2023
Intended Status: Informational
Expires: 29 July 2023
Authors: D. Migault E. Lewis D. York
Ericsson ICANN ISOC

Recommendations for DNSSEC Resolvers Operators

Abstract

The DNS Security Extensions (DNSSEC) define a process for validating received data and assert them authentic and complete as opposed to forged.

This document clarifies the scope and responsibilities of DNSSEC Resolver Operators (DRO) as well as operational recommendations that DNSSEC validators operators SHOULD put in place in order to implement sufficient trust that makes DNSSEC validation output accurate. The recommendations described in this document include, provisioning mechanisms as well as monitoring and management mechanisms.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on 29 July 2023.

Copyright Notice

Copyright (c) 2023 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents

(https://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Revised BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Revised BSD License.

Table of Contents

1. Introduction

The purpose of a DNSSEC Resolver Operator (DRO) is to provide DNSSEC validation in to their users.

By validating with DNSSEC a received Resource Record Set (RRset), the resolver provides a high level of certainty that the information carried by the RRset is effectively as published by the legitimate

owner of the RRset. The act of DNSSEC validation [RFC4033][RFC4035] can be broken into two part: A *Signature Validation* which binds the RRset to a private key as well as *Trust* in the owner of the private being the legitimate owner.

Signature Validation consists in checking the cryptographic signature of a RRset and involves among other parameters a DNSKEY Resource Record (RR) and RRSIG RR and the RRset itself. The signature validation process results in designating the owner of the RRset as the owner of the corresponding private part of the public key contained in the DNSKEY RR. Cryptography provides a high level of confidence in the binding to the private key. In that sense, a rogue RRset likely results from the private key being exposed or guessed -- as opposed to signature or key collisions for example. As such differ the confidence into the Trust to designate which DNSKEY RR is legitimate.

Trust implicitly assumes the private keys used to sign are not shared and as such can be associated their respective owners. The purpose of Trust is to put significant confidence into designating the legitimate DNSKEY RR - which also characterizes the corresponding private key. Such trust is provided by a Trust Anchor (TA), and the chain of trust established between the TA and the DNSKEY RR. The chain of trust is obtained by recursively validating the DNSKEY RRs.

As a result, such trust results from the trust placed in the TA as well as the delegation mechanism provided by DNSSEC and the Signature Validation. As TAs need to be managed over time, the trust also concerns the management procedure of the TA. This is the main concern of this document.

Data's authenticity and integrity is tied to the operator of the key that generates the signature. It is conceivable that a validator could "know" the keys of each data source, but this is not practical at large scale. To counter this, DNSSEC relied on securely chaining keys in a manner isomorphic to the way names are delegated. Keys for a name will "vouch for" keys at a name delegated via the signing of a DS resource record set.

Using keys to vouch for keys, recursively, works when a manageable set of key to name associations are determined to be "trusted" - and are called trust anchors. In DNSSEC, a validator needs one or more Trust Anchors from which to grow chains of verified keys.

With operational experience, a twist has emerged. More often, to date, failed validation is due to operator error or software bug [ENT] and not an attempt to forge data. In general badly signed RRsets or zone badly delegated are out of scope of the DRO's

responsibility. However, the DRO may reflect this operational error with a temporary solution designated as Negative Trust Anchors (NTA) [RFC7646]. A NTA instructs a validator to ignore the presence of keys for a name, reacting as if the name is unsigned.

Once accurately validated the RRset is assumed to be accurately validated and trusted during the time indicated by its TTL.

The responsibilities of a DRO are limited to the management of TAs as well as providing the necessary infrastructure to perform the signature validation, e.g. appropriated libraries and time. More specifically, badly signed zones or insertion of malicious DNSKEY fall out of the DRO's responsibilities. Even though these threats fall out of these responsibilities, a DRO may collaborate with authoritative servers to limit the damage of their operational errors.

This document is focused on operational recommendations that a DRO SHOULD put in place in order to implement sufficient trust that makes DNSSEC validation output accurate. The recommendations described in this document include provisioning mechanisms as well as monitoring and management mechanisms.

The mechanisms provided are designed in accordance of the DNSSEC trust model as to meet the current operations of DNSSEC. Such trust model is briefly recapped in Section 4 so operators understand the limits and motivations for such mechanisms.

2. Requirements Notation

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

3. Terminology

This document uses the following terminology:

DNSSEC validator: the entity that performs DNS resolution and performs signature validation.

Accurate validation: validation that avoids false positives and catches true negatives.

Trust Anchor Data Store: a module (of code) implementing functions related to the trust anchors used by the validator. This is essentially a database allowing access, monitoring of, and changes to trust anchors.

DNSSEC Resolver Operator (DRO): The operator providing DNSSEC validation service and managing DNSSEC validators

4. DNSSEC Validator Description

This is a conceptual block diagram of the elements involved with DNSSEC validation. This is not meant to be an architecture for code, this is meant to be a framework for discussion and explanation.

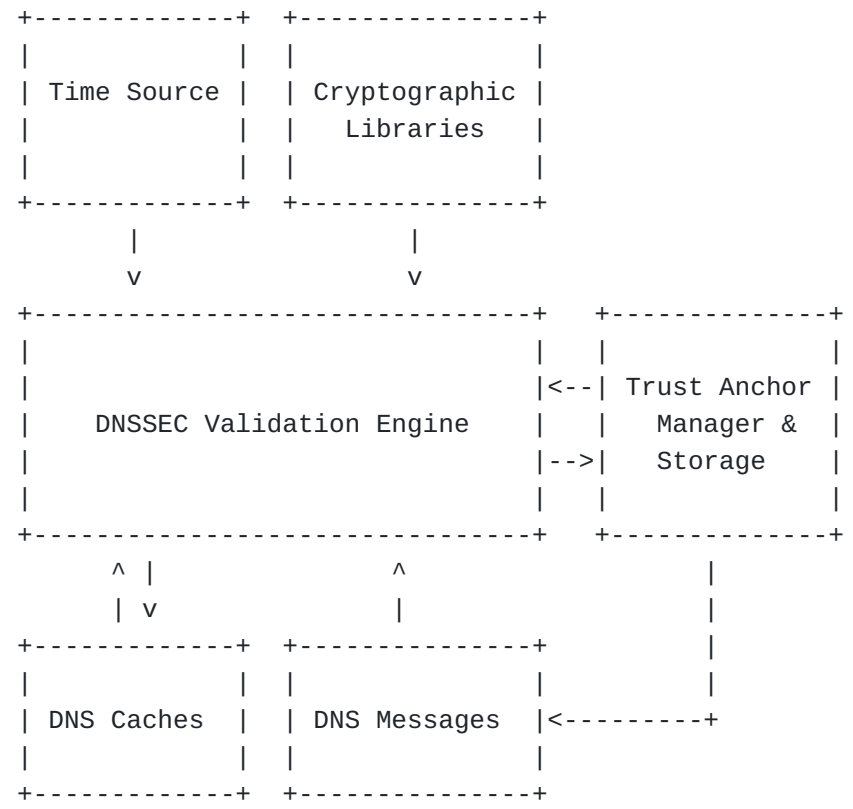

Figure 1: DNSSEC Validator Description

Time Source: The wall clock time provides the DNSSEC Validation Engine the current time. Time is one input used to validate the RRSIG Signature and Inception Fields to provide some protection against replay attacks.

Cryptograhic Libraries: The code performing the cryptographic functions providing the DNSSEC Validation Engine the ability to check the Signature Field that contains the cryptographic signature covering the RRSIG RDATA.

DNS Message: DNS responses are used to carry the information from the DNS system. The receiver of the DNS message can be any kind of application including DNS-related application such as in the case of automated Trust Anchor update performed by the Trust

Anchor Manager & Storage. The DNSSEC Validator Engine accurately validates the DNS responses before caching them in the DNS Cache and forwarding them to the DNS receiver. In case of validation failure, an error is returned and the information may be negatively cached.

DNS Caches: Include positive and negative caches. The DNSSEC Validation Engine fills DNS Caches with validated DNS RRsets. The DNSSEC trust model considers that once a RRset has been accurately validated by the DNSSEC Validator Engine, the RRset is considered trusted (or untrusted) for its associated TTL. DNS Caches contain RRsets that may contain information requested by the application (RRset of type AAAA for example) as well as RRset necessary to accurately validate the RRsets (RRsets of type DNSKEY or RRSIG for example). This can also include RRsets that were not DNSSEC signed at validation time.

Trust Anchor Manager: The database of trust anchors associated to database management processes. This function provides the DNSSEC Validation Engine Trust Anchor information when needed. When a TA needs to be updated, the Trust Anchor Manager is also responsible for handling the update procedure. This includes sending DNS Messages as well as treating appropriately the DNS responses that have been accurately validated by the DNSSEC Validator Engine. This will require the DNSSEC Validator to update Trust Anchor information, whether via methods like Automated Updates of DNSSEC Trust Anchors [RFC5011], management of Negative Trust Anchors, or other, possibly not yet defined, means.

DNSSEC Validation Engine: This follows local policy to approve data. The approved data is returned to the requesting application as well as to the DNS Caches. While the cryptographic computation of the RRSIG signature may be the most visible step, the RRSIG record also contains other information intended to help the validator perform its work, in some cases sanity checks are performed. For instance, the original TTL (needed to prepare the RR set for validation) ought to be equal to or higher than the received TTL.

Not shown - Name Server Process Management interfaces to elements, handling of Checking Disabled request, responses, as well as all API requests made of the name server.

5. Recommendations Categories

A DRO needs to be able to enable DNSSEC validation with sufficient confidence they will not be held responsible in case their resolver does not validate the DNSSEC response. The minimization of these risks is provided by setting automated procedures, when a resolver

is started or while it is operating, as well as some on-demand operations that enable the DRO to perform a specific operation. The recommendations do not come with the same level of recommendations and some are likely to be required. Others are optional and may be followed by more advanced DROs. It is up to the DRO to determine their applicability.

Regarding recommendations on the behavior of the resolver, some recommendations may already be applied by default by the operated resolver software, in which case the DRO does not have to perform any action. Some recommendations may require to be activated via configuration by the DRO. Some recommendations may simply not be provided by the operated software, in which case we recommend the DRO to contact the software vendor for further discussion.

The recommendations are voluntary restrictive to prevent side effects of interventions that will be hard to predict, debug and understand.

The RECOMMENDATIONS in this document are subdivided into the following categories:

Start-up recommendations (STARTUP): which describes RECOMMENDED operations the DRO is expected to perform when the resolver is started. These operations typically include health checks of the infrastructure the resolver is instantiated on as well as configuration check.

Run time recommendations (RUNTIME): which describes RECOMMENDED operations the DRO is expected to perform on its running resolvers. These operations typically include health checks of the infrastructure as well as the resolvers.

On demand recommendations (ONDEMAND): which describes the RECOMMENDED operations a DRO may perform. This includes the ability to operate health checks at a given time as well as specific operations such as flushing the cache. The reason this document mentions these recommendations is to enable DROs to have the appropriates tools as well as to restrict their potential interventions.

6. Time deviation and absence of Real Time Clock Recommendations

With M2M communication some devices are not expected to embed Real Time Clock (Raspberry Pi is one example of such devices). With Raspberry Pi for example, when these devices are re-plugged their time is reset to some initial values like (January 1, 1970 for example) until they get re synchronized via NTP. Other devices that have clocks may suffer from time deviation. These devices cannot rely on their time estimation to perform DNSSEC validation.

Time synchronization may be performed manually, but for the sake of operations it is strongly RECOMMENDED to automate the time synchronization on each resolver.

STARTUP:

- A DRO MUST provide a mean to update the time without relying on DNSSEC when the DNSSEC validator is started. The resolver MUST NOT start if the time synchronization does not succeed at start time.

Note that updating time in order to be able to perform DNSSEC validation may become a form of a chicken and egg problem when the NTP server is designated by its FQDN. The update mechanism must consider the DNSSEC validator may not able to validate the DNSSEC queries. In other words, the mechanisms may have to update the time over an unsecure DNSSEC resolution.

RUNTIME:

- While operating, a DRO MUST closely monitor time derivations of the resolvers and maintain the time synchronized.

ONDEMAND:

- A DRO SHOULD be able to check and synchronize, on demand, the time of the system of its resolver.

Note that time synchronization is a sensible operation. A DRO MUST update time over an authenticated and secure channel.

For all recommendations, it is strongly RECOMMENDED that recommendations are supported by automated processes.

7. Trust Anchor Related Recommendations

A TA store maintains associations between domain names and keys (whether stored as in a DNSKEY resource record or a DS resource record) and domain names whose key are to be ignored (negative trust anchors). The TA store is essentially a database, storing the (positive) trust anchors. Management of the TA can be done manually or in an automated way. Automatic management of the TA is RECOMMENDED and can be subdivided into the following sub-categories:

1. **Initial TA provisioning:** that is the ability, for a DRO, to ensure a starting resolver is automatically provisioned with an up-to-date configuration. This includes defining a trust model and ensuring the associated TAs values are valid at the time the resolver is started. # This includes the TA associated to the trust model established by the DRO.

2.

TA update over time: while the trust model may not evolve, the cryptographic keys associated to the security entry points are subject to change and thus a TA needs to be updated dynamically over time by a resolver. In this case, the DRO needs to check that the resolver has properly updated the TAs.

3. **TA reporting:** The reporting of the TA used by the resolver is made to the DRO as well as the authoritative servers which are hold responsible for making their zone validated by DNSSEC resolvers.

Note that TA update and TA reporting only concerns running resolvers. This section is only considering TA and not NTA. The handling of NTA is detailed in Section 8.

7.1. Trust Anchor Configuration

When a DRO starts a DNSSEC resolver, the DNSSEC resolver is provisioned with the TAs as part of its configuration. As these TAs change over time, the DRO MUST ensure the resolvers are always provisioned with up-to-date TAs and detect deprecated configuration.

To do so, the TA configuration is considered as a trusted model instantiated as follows:

1. **Trust model definition:** The DRO defines the domain names that constitutes security entry point (TA) -- see Section 7.1.1 for more details.

2. **Trust model bootstrapping:** A bootstrapping procedure is applied to each TA to retrieve their TA values - that is the corresponding value of the key - in a trusted way. Such TA MAY have a specific format not understood by the resolver -- see Section 7.1.2 for more details.

3. **DNSSEC resolver configuration generation:** The TA with associated values are formatted appropriately for the DNSSEC resolver that will be instantiated. In many cases the appropriated format is the DNS RRset format - see Section 7.1.3 for more details.

4. **DNSSEC resolver instantiation:** The DNSSEC resolver is started, performs some checks before becoming operational, that is checking compability between provisioned TAs and those found online. These checks are implemented by the resolver and not really in scope of the DRO -- see Section 7.1.4 for more details.

While these steps may require some specific development for complex trust models, no additional deployment is required when using the default model where the root zone is defined as the only secure entry point. In such cases, the trusted model bootstrapping is performed by software-update that relies on the code signing key of the software provider. The software provider also provides the configuration to the appropriated format and the checks at instantiation -- see Section 7.1.2.1.

7.1.1. Definition of the Trust model

The DRO defines its trust model by explicitly mentioning the domain name that constitutes security entry point as well as domain name that are known to be unsecured.

This document does not provide recommendations regarding the number of TAs a DRO needs to configure its DNSSEC resolver with. There are many reasons a DRO may be willing to consider multiple TAs as opposed to a single Root Zone Trust Anchor. In fact it is not always possible to build a trusted delegation between the Root Zone and a sub zone. This may happen, for example, if one of the upper zones does not handle the secure delegation or improperly implements it. Typically, a DS RRset may not be properly filled or its associated signature cannot be validated. As the chain of trust between a zone and the root zone may not be validated, the DNSSEC validation for the zone requires a TA. Such DNS(SEC) resolutions may be critical for infrastructure management. A company may, for example, address all its devices under the domain example.com and may not want disruption to happen if the .com delegation cannot be validated for any reason. Such companies may operate DNSSEC with a TA for the zone example.com in addition to the regular DNSSEC delegation. Similarly some domains may present different views such as a "private" view and a "public view". These zones may have some different content, and may use a different KSK for each view.

The domain name chosen as the TA security entry point may be a child zone from another TA security entry point. For example, the DRO may use seperate TAs for example.com, .com, and the root zone. The TA associated with a domain name is determined by the longest match. However, the trust model MUST at least ensure that any domain name in the DNS be covered by at least one TA. As the number of top level domains is evolving overtime, it remains safe to keep the root zone as a security entry point in order to cover the full domain name space.

The default trust model remains rooted in the root zone as a security entry point, and no zones being considered as unsecured. By default, the instantiation of this trusted model is delegated by the DRO to the software vendor that is responsible to update the the

default trust model with the up-to-date value of the TA. In such cases, the trusted model is implemented by software-update that relies on the code signing key of the software provider.

The use of a different trust model must be carefully considered and NOT RECOMMENDED for DRO without a strong DNSSEC expertise. A DRO implementing another trust model MUST be able to ensure that the trust model is properly implemented - that is with up-to-date TAs. The DRO MUST carefully balance the implications, the risks, the necessary involved mechanisms to achieve this goal with the potential benfits. More specifically, it is expected updating the trust model's instantiation is performed in an automated way with strong validation validation and checks being put in place. It is also likely that the DRO and the owner of the TA will have a specific relationship and TA update will be provided via a specific out-of-band channel. More specifically, [RFC5011] that we recommend to update TA by a running instance of resolver, is likely not to be sufficent, as it is completly automated and relies on in band signaling which comes with risks detailled in [RFC5011], Section 8. A DRO need to understand these risks. Note also that even when [RFC5011] the owner of the TA needs to respect the [RFC5011] timings -- see Section 7.2.1.

7.1.2. Trust Model Bootstrapping

The purpose of the boostrapping step is clearly to securely retrieve DNSKEY as well as DS RRsets with a valid and authentic RDATA to implement the trust model (see Section 7.1.1). Authentic data includes data that are up-to-date at the time these are requested, provided with securely and verified. In particular the TA MUST NOT be retrieved from a local source that is not known to be up to date. This typically includes software or any local store of data for which there is not a dedicated and automated updating system. Similarly, TA MUST NOT be retrieved from untrusted communication such as a DNS resolution that cannot be verified or validated.

The authenticity of the RDATA is usually based on the authentication of the source which can take multiple forms, but the principle is that a chain of signature ends up being validated by a trusted key. For TA that are not the root zone KSK, DNSSEC may be used to retrieve and validate the TA. Note that such means to validate the authenticity, the TA as a subdomain mostly prevents potential disruptions of the parent domains. In many cases, an alternative trusted source will be preferred such as TLS which is likely to rely on the Web PKI, or the signature of a file.

Although some bootstrapping mechanisms to securely retrieve publish [RFC7958] and retrieve [UNBOUND-ANCHOR] the Root Zone Trust Anchor have been defined, it is believed these mechanisms should be

extended to other KSKs or Trust Anchors. Such bootstrapping process enables a DRO to start a DNSSEC resolver from a configuration file, that reflects the trust model of the DRO.

STARTUP:

* DRO SHOULD only rely on TA associated with a bootstrapping mechanism.

7.1.2.1. IANA Trust Anchor Bootstrapping

For validators that may be used on the global public Internet (with "may be" referring to general purpose, general release code), handling the IANA managed root zone KSK trust anchor is a consideration.

The IANA managed root zone KSK is an operationally significant trust point in the global public Internet. Attention to the trust anchor for this point is paramount. Trust anchor management ought to recognize that the majority of operators deploying DNSSEC validators will need to explicitly or implicitly rely on this trust anchor. Trust anchor management needs to recognize that there may be other trust anchors of interest to operators. Besides deployments in networks other than the global public Internet (hence a different root), operators may want to configure other trust points.

The IANA managed root zone KSK is managed and published as described in "DNSSEC Trust Anchor Publication for the Root Zone" [RFC7958]. That document is written as specific to that trust point. Other trust points may adopt the technique describe (or may use other approaches).

This represents a consideration for implementations. On one hand, operators will place special emphasis on how the root zone DNSSEC KSK is managed. On the other hand, implementations ought to accommodate trust anchors in a general manner, despite the odds that other trust anchors will not be configured in a specific deployment.

Because of this, it is RECOMMENDED that implementations make the root zone trust anchor obvious to the operator while still enabling configuration of general trust points.

7.1.3. Configuration Generation

The generation of a configuration file associated to the TA is expected to be implementation independent. The necessity of tweaking the data depending of the software implementer or eventually the software version introduces a vector for configuration errors.

7.1.4. DNSSEC Resolver Instantiation

STARTUP:

* DNS resolver MUST validate the TA before starting the DNSSEC resolver, and a failure of TA validity check MUST prevent the DNSSEC resolver to be started. Validation of the TA includes coherence between out-out band values, values stored in the DNS as well as corresponding DS RRsets.

7.2. Trust Anchor Update

Updating the TA reflects the evolution of the trust. It is important to understand that this section does not consider the trust model to be updated by the DRO. This has been defined in section Section 7.1. Instead, it considers the evolution over time of the instantiation of the trust model, that is the update of the values associated to the TA, by a running instance of resolver. Such updates need to be operated in a reliable and trusted way. Note that you may not need to proceed to such TA update if the DRO can guarantee that any instance has been instantiated with a configuration the reflects the current value of the TAs. Given that the roll-over phases for a TA update are relatively long, for DRO that can ensure a fast deployment of the latest release of resolver, with the defult trust model being implemented by the software vendor, TA update is likely to be achieved via software updates.

The remaining of this section is left to DRO willing to ensure their instance have the capacity to update their TAs while running. This may be a specific feature or a complementary feature to software update. The value associated to the TA may be updated over time which is part of the maintenance of the configuration and needs to be performed by the DNSSEC resolver without any intervention of the DRO. This is the purpose of this section.

TA update is expected to be transparent to the DRO (see Section 7.2.1. However, a DRO MAY wish to ensure its resolvers operate according to the provisioned configurations and are updated normally (see Section 7.2.2. This includes for a DRO the ability to check which TA are in used as well as to resolve in collaboration of authoritative servers and report the used TAs.

7.2.1. Automated Updates to DNSSEC Trust Anchors

Trust is inherently a matter of an operations policy. As such, a DRO will need to be able to update the list of Trust Anchors. TA updates are not expected to be handled manually. This introduces a potentially huge vector for configuration errors, due to human intervention as well as potential misunderstanding of ongoing operations. Instead DRO will rely on "Automated Updates to DNSSEC

Trust Anchors" [RFC5011] when the TA signers is known to respect the [RFC5011] timing.

STARTUP:

- *DRO SHOULD check TA signers commits to respect "Automated Updates to DNSSEC Trust Anchors" [RFC5011] timings.

- *DRO SHOULD enable "Automated Updates to DNSSEC Trust Anchors" [RFC5011] [I-D.ietf-dnsop-rfc5011-security-considerations].

Note that while automation prevents the risks associated to manual intervention, but managing TA on its own - that is via other means than software updates, does not come without any risks. A DRO need to understand these risks.

7.2.2. Automated Trust Anchor Check

A DRO SHOULD regularly check the trust anchor used by the DNSSEC resolver is up-to-date and that values used by the resolvers are conform to the ones in the configuration (see Section 7.1). Such check is designated as TA health check.

Note that retrieving in an automated way the value of the TA removes old values from the configuration and ensures that resolvers are always started with up-to-date values. In the case of a key roll over, the resolver is moving from an old value to an up-to date value. This up-to-date value does not need to survive reboot, and there is no need to update the configuration file of the running instances - configuration is updated by a separate process. To put it in other words, the updated value of the TA is only expected to be stored in the resolver's memory. Avoiding the configuration file to be updated prevents old configuration file to survive to writing error on read-only file systems.

The TA used by a resolver may be part of a configuration parameter as well as part of an internal state of the resolver. It is NOT RECOMMENDED a DRO accesses configuration or internal state of a resolver as it may open the resolver to other vulnerabilities and provides privileged access to a potential attacker.

STARTUP:

- *DRO SHOULD enable "Signaling Trust Anchor Knowledge in DNS Security Extensions (DNSSEC)" [RFC8145] to provide visibility to the TA used by the resolver. The TA can be queried using a DNSKEY query. The channel MAY be protected and restricted to the DRO.

Note also that [RFC8145] does not only concern Trust Anchor but is instead generic to DNSKEY RRsets. As a result, unless for the root

zone, it is not possible to determine if the KSK/ZSK or DS is a Trust Anchor or a KSK/ZSK obtained from regular DNSSEC resolutions.

TA health check includes validating DNSKEY RRsets and associated DS RRsets in the resolver, on the DNS authoritative servers as well as those obtained out-of-band. TA health check results MUST be logged. The check SHOULD evaluate if the mismatch resulted from an ongoing normal roll over, a potential emergency key roll over, failed roll over or any other envisioned cases. Conflicts are not inherently a problem as some keys may be withheld from distribution via the DNS. A failed key roll over or any other abnormal situation MUST trigger an alarm.

RUNTIME:

*A DRO SHOULD regularly run TA health checks.

If the mismatch is due to a failed key roll-over, this SHOULD be considered as a bug by the DRO. The DRO MUST restart the resolver with updated TA.

ONDEMAND:

*A DRO SHOULD be able to check the status of a TA as defined in Section 3 of [RFC7583].

8. Negative Trust Anchors Related Recommendations

When the DNSSEC Resolver is not able to validate signatures because a key or DS has been published with an error, the DNSSEC Operator MAY temporarily disable the signature check for that key until the time the error is addressed. This is performed using NTA[RFC7646]. NTA represents the only permitted intervention in the resolving process for a DRO.

NTA are considered as temporary fix for a known unsecured domain, which is different from an TA that would not be trusted. The designation of NTA might be misleading, but NTA are not expected to be part of the trust model. This does not prevent a DRO to provision NTA as a configuration parameters NTA. The management of the configuration SHOULD be automated as described in Section 7.1.

Handling NTA is described in [RFC7646] and a DRO SHOULD follow these guidelines. The intent of this section is to position these guidelines toward the operational recommendations provided in this document.

STARTUP:

- DRO SHOULD be able to automatically configure NTA when starting DNSSEC resolvers.

ONDEMAND:

- DRO SHOULD set automated procedures to determine the NTA of DNSSEC resolvers.

- DRO SHOULD be able to handle NTA as defined in [RFC7646].

Note that adding a Negative Trust Anchor only requires the domain name to be specified. Note also that NTA can disable any sort of DNSKEY and is not restricted to TA.

A failure in signaling validation is associated to a mismatch between the key and the signature. DNSKEY/DS RRsets for TA have a higher level of trust then regular KSK/ZSK. In addition, DRO are likely to have specific communication channel with TA maintainer which eases trouble shooting.

A signature validation failure is either an attack or a failure in the signing operation on the authoritative servers. The DRO is expected to confirm this off line before introducing the NTA. This is likely to happen via a human confirmation. As a result here are the following recommendations:

RUNTIME:

- DRO SHOULD monitor the number of signature failure associated to each DNSKEY. These number are only hints and MUST NOT trigger automated insertion of NTA.

- A DRO MAY collect additional information associated each DNSKEY RRSets. This information may be useful to follow-up roll over when these happen and evaluate when a key roll over is not performed appropriately on the resolver side or on the authoritative server. It would provide some means to the DRO to take action with full knowledge without necessary asking for a confirmation. In other cases it could prevent invalidation to happen. These check may be performed for a limited subset of domains or generalized.

9. ZSK / KSK (non TA) Related Recommendations

KSK / ZSK are not part of the DNSSEC validator configuration. Their values in the DNS Caches may not reflect those published by the authoritative servers or may be incoherent with the RRset in the DNS Cache they are validating. However, such incoherence primary results

from error in the management of the authoritative servers. As a result, it is not expected that the DNSSEC validator provides complex management facilities to address these issues as this will modify the DNS architecture and add complexity that is not proved to be beneficial. As a result, recommendations always belong to the run time or on demand recommendations. The main difference between TA and KSK/ZSK is that the DRO does not necessarily have an out of band mechanism to retrieve the RRsets. As a result, the DRO has less information to determine and confirm what is happening. The default recommendation is to let things go.

A number of reasons may result in inconsistencies between the RRsets stored in the cache and those published by the authoritative server.

An emergency KSK / ZSK rollover may result in a new KSK / ZSK with associated new RRSIG published in the authoritative zone, while DNSSEC validator may still cache the old value of the ZSK / KSK. For a RRset not cached, the DNSSEC validator performs a DNSSEC query to the authoritative server that returns the RRset signed with the new KSK / ZSK. The DNSSEC validator may not be able to retrieve the new KSK / ZSK while being unable to validate the signature with the old KSK / ZSK. This either results in a bogus resolution or in an invalid signature check. Note that by comparing the Key Tag Fields, the DNSSEC validator is able to notice the new KSK / ZSK used for signing differs from the one used to generate the received generated signature. However, the DNSSEC validator is not expected to retrieve the new ZSK / KSK, as such behavior could be used by an attacker. Instead, ZSK / KSK key roll over procedures are expected to avoid such inconsistencies.

Similarly, a KSK / ZSK roll over may be performed normally, that is as described in [RFC6781] and [RFC7583]. While the KSK / ZSK roll over is performed, there is no obligation to flush the RRsets in the cache that have been associated with the old key. In fact, these RRsets may still be considered as trusted and be removed from the cache as their TTL timeout. With very long TTL, these RRsets may remain in the cache while the ZSK / KSK with a shorter TTL is no longer published nor in the cache. In such situations, the purpose of the KSK / ZSK used to validate the data is considered trusted at the time it enters the cache, and such trust may remain after the KSK / ZSK is being rolled over. Note also that even though the data may not be associated to the KSK / ZSK that has been used to validate the data, the link between the KSK / ZSK and the data is still stored in the cache using the RRSIG. Note also that inconsistencies between the ZSK / KSK stored in the cache and those published on the authoritative server, may lead to inconsistencies to downstream DNSSEC validators that rely on multiple cache over time.

Typically, a request for the KSK / ZSK may have been provided by a cache that is storing the new published value, while the data and associated signatures may be associated to the old KSK / ZSK.

Incoherence between RRsets and DNSKEYs is not the responsibility of the DRO. Instead, it is the responsibility of authoritative server publishing these data. This includes insuring the coherence between TTLs and signature validation periods as well as small variations of the resolvers clocks. Section 4.4.1 of [RFC6781] provides some recommendations that can be implemented by the authoritative server which puts the responsibility of failure of signature validation under the responsibility of the authoritative server. A DRO MAY however limit the risks for these inconsistencies to happen by configuring the DNSSEC validator with generic rules that applies to the validation process. Typically, the TTL associate to the DNSKEY is an engagement from the authoritative server that the DNSKEY will remain valid over this period. As this engagement supersedes the validation of any RRSIG and by extension to any RRset in the zone, this TTL value may be used as the maximum value for the TTL associated to FQDNs in the zone. Section 8.1 of [RFC4033] mention the ability by the resolver to set the upper bound of the TTL to the remaining signature validity period. In addition, the DNSSEC validator should also be able to provide a maximum values for TTLs. These values MAY also consider the small inaccuracy of the local clock.

RUNTIME:

* To limit the risks of incoherent data in the cache, it is RECOMMENDED DRO enforce TTL policies of RRsets based on the TTL of the DS, KSK and ZSK. RRsets TTL SHOULD NOT exceed the DS, KSK or ZSK initial TTL value, that TTL SHOULD trigger delegation revalidation as described in [I-D.ietf-dnsop-ns-revalidation]. TTL SHOULD NOT exceed the signature validity.

10. DNSKEY Related Recommendations

This section considers the recommendations that are common to TA as well as non TA DNSKEY RRsets.

10.1. Automated Reporting

A DRO MAY regularly report the Trust Anchor used to the authoritative server. This would at least provide insight to the authoritative server and provide some context before moving a key roll over further.

The purpose of reporting the currently used Trust Anchor for a domain name is to establish an informational channel between the resolver and the authoritative server. This data may not directly be

useful for the DNSSEC Resolver, but instead to the authoritative server. In return it is likely the authoritative server will take the appropriate steps in operating the authoritative server and consider this information. This results in the following recommendation:

RUNNING:

* A DRO SHOULD enable TA reporting to the authoritative server as specified in "Signaling Trust Anchor Knowledge in DNS Security Extensions (DNSSEC)" [RFC8145]

10.2. Interactions with the cached RRsets

The purpose of automated checks is to enable early detection of failed operations, which provides enough time to the DRO to react without any consequences. On the other hand, these checks MAY reveal as well that a rogue TA has been placed and that the resolver is corrupted. Similarly, a DRO may be informed by other channel a rogue or unwilling DNSKEY has been emitted.

In such situation, the DRO SHOULD be able to remove the RRsets validated by the rogue DNSKEY.

ONDEMAND:

* A DRO MUST be able to flush the cached data subtree associated to a DNSKEY

11. Cryptography Deprecation Recommendations

As mentioned in [RFC8247] and [RFC8221] cryptography used one day is expected over the time to be replaced by new and more robust cryptographic mechanisms. In the case of DNSSEC signature protocols are likely to be updated over time. In order to anticipate the sunset of one of the signature scheme, a DNSSEC validator may be willing to estimate the impact of deprecating one signature scheme.

Currently, interoperability and security are enforced via cryptographic recommendations [RFC8624] that are followed by both resolvers and authoritative servers. The implementation of such guidance is ensured by the software vendors and the compliance of their releases.

To safely deprecate one signature scheme, the DNSSEC validator operator is expected to follow the recommendation below:

STARTUP: * DRO MUST ensure recent software releases that comply with the most recent cryptographic guidances are being used

RUNTIME:

- *A DRO SHOULD regularly request and monitor the signature scheme supported by an authoritative server.

- *A DRO SHOULD report a "Unsupported DNSKEY Algorithm" as defined in [RFC8914] when a deprecated algorithm is used for validation.

One inconvenient to such strategy i sthat it does not let one DRO to take advantage of more recent cryptographic. While currently not being widely used, a DRO MAY share information with authoritative server in the hope that future deployment of authoritative servers will be able to leverage it. [RFC6975] provides the ability for a DNSSEC validator to announce an authoritative server the supported signature schemes. However, a DNSSEC validator is not able to determine other than by requesting and monitoring DNSKEY RRsets as well as RRSIG. These RRsets are received by enabling DNSSEC validation by default which is obviously the case for DNSSEC validator.

12. Invalid Reporting Recommendations

A DNSSEC validator receiving a DNS response cannot make the difference between receiving an non-secure response versus an attack. Dropping DNSSEC fields by a misconfigured middle boxes, such as DS, RRRSIG is considered as an attack. A DNSSEC validator is expected to perform secure DNS resolution and as such protects its stub client. An invalid response may be the result of an attack or a misconfiguration, and the DNSSEC validator may play an important role in sharing this information with the authoritative server or domain name owner.

RUNTIME:

- *DRO SHOULD monitor and report DNSSEC validation error. Reporting may take various means but the DRO SHOULD implement [RFC8914] to inform the DNS client.

13. Transport Recommendations

DNSSEC validation requires that the validator is able to reliably obtain necessary records, especially DNSKEY records. This should be done at initial configuration, and tested periodically.

This means the validator MUST ensure it is configured so that the UDP and TCP transports, and DNS resolver components, are compatible with the network paths that the majority of DNS queries traverse - which includes compatibility between DNS and transport parameters with the Maximum Transmission Unit (MTU).

In other words, make sure that: 1. DNS UDP bufsize (EDNS parameter) is set to a value compatible with network MTUs the queries and responses will encounter. If the validator advertises a bufsize >> MTU, responses with the IPv4 Don't Fragment (DF) bit set whose size R where MTU < R <= bufsize exceeds the MTU will be dropped by the router with MTU < R.

1. The validator's OS TCP configuration has its advertised Maximum Segment Size (MSS) set to a value compatible with network MTUs the queries and responses will encounter. * Having an advertised MSS set to a value < MTU ensures that Path MTU Discovery is not required * If PMTUD fails for any reason, or if the server responding does not maintain or use PMTUD, and advertised MSS > MTU at any point in the path, TCP may encounter problems caused by IP fragmentation and reassembly. * This is particularly relevant if there are any NAT type devices in the path, as those may not properly handle fragmentation and reassembly * If all TCP segments are smaller than the path MTU, TCP will work reliably.

The avoidance of fragmentation in order to address known fragmentation-related security issues with DNS (leading to cache poisoning, for example) has resulted in the need to set the DF bit on UDP. Validators will need to ensure their local environment can reliably get any critical DNSSEC records (notably DNSKEY) over UDP, or reliably get responses with TC=1 if overly large responses cannot be sent over UDP due to answers not fitting within the advertised bufsize payload. Validators also need to ensure TCP works if it is needed, for the same situations.

STARTUP: * DRO MUST ensure UDP and TCP transport are enabled. * DRO MUST ensure DNS and TCP parameters will not exceed the expected MTU

RUNTIME: * DRO SHOULD regularly discover MTU and ensure the DNS and TCP parameters are aligned wit this value. * DRO SHOULD closely monitor the absence of responses, or ICMP PTB message as a potential incompatibility between configured parameters and the MTU.

14. IANA Considerations

There are no IANA consideration for this document.

15. Security Considerations

The recommendations listed in this document have two goals. First ensuring the DNSSEC validator has appropriated information to appropriately perform DNSSEC validation. Second, monitoring the necessary elements that would enable a DNSSEC validator operator to ease a potential analysis. The recommendations provide very limited ability for a DNSSEC validator operator to alter or directly

interfere on the validation process and the main purpose in providing the recommendations was to let the protocol run as much as possible. Providing inappropriate information can lead to misconfiguring the DNSSEC validator, and thus disrupting the DNSSEC resolution service. As a result, enabling the setting of configuration parameters by a third party may open a wide surface of attacks. In addition, such changes may lead to unexpected corner cases that would result in making analysis and trouble shooting very hard.

As an appropriate time value is necessary to perform signature check, an attacker may provide rogue time value to prevent the DNSSEC validator to check signatures.

An attacker may also affect the resolution service by regularly asking the DNSSEC validator to flush the KSK/ZSK from its cache. All associated data will also be flushed. This generates additional DNSSEC resolution and additional validations, as RRSet that were cached require a DNSSEC resolution over the Internet. This affects the resolution service by slowing down responses, and increases the load on the DNSSEC validator.

An attacker may ask the DNSSEC validator to consider a rogue KSK/ ZSK, thus hijacking the DNS zone. Similarly, an attacker may inform the DNSSEC validator not to trust a given KSK in order to prevent DNSSEC validation to be performed.

An attacker (cf. Section 7) can advertise a "known insecure" KSK or ZSK is "back to secure" to prevent signature check to be performed correctly.

As a result, information considered by the DNSSEC validator should be from a trusted party. This trust party should have been authenticated, and the channel used to exchange the information should also be protected and authenticated.

The software used for DNSSEC validator is not immune to bugs and may become vulnerable independently of how it is operated. As a result a DRO SHOULD NOT depend on a single implementation or version of a given software and SHOULD instead run at least two independent pieces of software.

16. Acknowledgment

The need to address DNSSEC issues on the resolver occured during multiple discussions including among others Ted Lemon, Ralph Weber, Normen Kowalewski, Mikael Abrahamsson, Jim Gettys, Paul Wouters, Joe Abley, Michael Richardson, Vladimír Čunát, James Gannon, Andrew McConachie, Peter Thomassen, Florian Obser and Brian Dickson.

We also appreciated the support of the DNSOP chairs Tim Wicinski, Suzanne Woolf and Benno Overeinder.

17. References

17.1. Normative References

[I-D.ietf-dnsop-ns-revalidation] Huque, S., Vixie, P. A., and R. Dolmans, "Delegation Revalidation by DNS Resolvers", Work in Progress, Internet-Draft, draft-ietf-dnsop-ns-revalidation-03, 6 September 2022, <https://www.ietf.org/archive/id/draft-ietf-dnsop-ns-revalidation-03.txt>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

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

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

[RFC5011] StJohns, M., "Automated Updates of DNS Security (DNSSEC) Trust Anchors", STD 74, RFC 5011, DOI 10.17487/RFC5011, September 2007, <https://www.rfc-editor.org/info/rfc5011>.

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

[RFC6975] Crocker, S. and S. Rose, "Signaling Cryptographic Algorithm Understanding in DNS Security Extensions (DNSSEC)", RFC 6975, DOI 10.17487/RFC6975, July 2013, <https://www.rfc-editor.org/info/rfc6975>.

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

[RFC7646] Ebersman, P., Kumari, W., Griffiths, C., Livingood, J., and R. Weber, "Definition and Use of DNSSEC Negative

Trust Anchors", RFC 7646, DOI 10.17487/RFC7646, September 2015, <https://www.rfc-editor.org/info/rfc7646>.

[RFC8145] Wessels, D., Kumari, W., and P. Hoffman, "Signaling Trust Anchor Knowledge in DNS Security Extensions (DNSSEC)", RFC 8145, DOI 10.17487/RFC8145, April 2017, <https://www.rfc-editor.org/info/rfc8145>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[RFC8221] Wouters, P., Migault, D., Mattsson, J., Nir, Y., and T. Kivinen, "Cryptographic Algorithm Implementation Requirements and Usage Guidance for Encapsulating Security Payload (ESP) and Authentication Header (AH)", RFC 8221, DOI 10.17487/RFC8221, October 2017, <https://www.rfc-editor.org/info/rfc8221>.

[RFC8247] Nir, Y., Kivinen, T., Wouters, P., and D. Migault, "Algorithm Implementation Requirements and Usage Guidance for the Internet Key Exchange Protocol Version 2 (IKEv2)", RFC 8247, DOI 10.17487/RFC8247, September 2017, <https://www.rfc-editor.org/info/rfc8247>.

[RFC8624] Wouters, P. and O. Sury, "Algorithm Implementation Requirements and Usage Guidance for DNSSEC", RFC 8624, DOI 10.17487/RFC8624, June 2019, <https://www.rfc-editor.org/info/rfc8624>.

[RFC8914] Kumari, W., Hunt, E., Arends, R., Hardaker, W., and D. Lawrence, "Extended DNS Errors", RFC 8914, DOI 10.17487/RFC8914, October 2020, <https://www.rfc-editor.org/info/rfc8914>.

17.2. Informative References

[ENT] Levigneron, V., "ENT was here !!!", n.d., <https://indico.dns-oarc.net/event/25/contributions/403/attachments/378/647/AFNIC_OARC_Dallas.pdf>.

[I-D.ietf-dnsop-rfc5011-security-considerations] Hardaker, W. and W. A. Kumari, "Security Considerations for RFC5011 Publishers", Work in Progress, Internet-Draft, draft-ietf-dnsop-rfc5011-security-considerations-13, 16 July 2018, <https://www.ietf.org/archive/id/draft-ietf-dnsop-rfc5011-security-considerations-13.txt>.

[RFC7958] Abley, J., Schlyter, J., Bailey, G., and P. Hoffman, "DNSSEC Trust Anchor Publication for the Root Zone", RFC

7958, DOI 10.17487/RFC7958, August 2016, <https://www.rfc-editor.org/info/rfc7958>.

[UNBOUND-ANCHOR] "unbound-anchor - Unbound anchor utility", n.d., <https://nlnetlabs.nl/documentation/unbound/unbound-anchor/>.

Authors' Addresses

Daniel Migault
Ericsson
8275 Trans Canada Route
Saint Laurent, QC 4S 0B6
Canada

Email: daniel.migault@ericsson.com

Edward Lewis
ICANN
United States of America

Email: edward.lewis@icann.org

Dan York
ISOC
United States of America

Email: york@isoc.org