DNSOP Working Group
Internet-Draft
Intended status: Standards Track
Expires: 28 April 2022

P. Thomassen
deSEC, Secure Systems Engineering
N. Wisiol
deSEC, Technische Universität Berlin

25 October 2021

Automatic DNSSEC Bootstrapping using Authenticated Signals from the Zone's Operator
draft-thomassen-dnsop-dnssec-bootstrapping-02

Abstract

This document introduces an in-band method for DNS operators to publish arbitrary information about the zones they are authoritative for, in an authenticated fashion and on a per-zone basis. The mechanism allows managed DNS operators to securely announce DNSSEC key parameters for zones under their management, including for zones that are not currently securely delegated.

Whenever DS records are absent for a zone's delegation, this signal enables the parent's registry or registrar to cryptographically validate the CDS/CDNSKEY records found at the child's apex. The parent can then provision DS records for the delegation without resorting to out-of-band validation or weaker types of cross-checks such as "Accept after Delay" ([RFC8078]).

[Ed note: Text inside square brackets ([]) is additional background information, answers to frequently asked questions, general musings, etc. They will be removed before publication. This document is being collaborated on at https://github.com/desec-io/draft-thomassen-dnsop-dnssec-bootstrapping/ (https://github.com/desec-io/draft-thomassen-dnsop-dnssec-bootstrapping/). The most recent version of the document, open issues, etc. should all be available there. The authors gratefully accept pull requests.]

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 28 April 2022.

Copyright Notice

Table of Contents

1. Introduction

Securing a DNS delegation for the first time requires that the Child's DNSSEC parameters be conveyed to the Parent through some trusted channel. How exactly this is done depends on the relationship the Child has with the Parent.

In general, the communication has to occur between the DNSSEC key holder and the Parent registry. It is often the case that the key is held by the Child DNS Operator. Furthermore, depending on the circumstances, the communication may also involve the Registrar, possibly via the Registrant (for details, see [RFC7344], Appendix A).

As observed in [RFC7344], this is often a manual process -- and not an easy one. Any manual process is susceptible to mistakes and/or errors. In addition, due to the annoyance factor of the process, involved parties may avoid the process of getting a DS record set published in the first place.

To alleviate these problems, automated provisioning of DS records has been specified in ([RFC8078]). It is based on the parental agent (registry or registrar) fetching DNSSEC key parameters in the form of CDS and CDNSKEY records ([RFC7344]) from the Child zone's apex, and validating them somehow. This validation can be done using DNSSEC itself if the objective is to update an existing DS record (such as during key rollover). However, when bootstrapping a DNSSEC delegation, the Child zone has no existing DNSSEC validation path, and other means to ensure the CDS/CDNSKEY records' legitimacy must be found.

For lack of comprehensive DNS-innate solution, either out-of-band methods have been used so far to complete the chain of trust, or cryptographic validation has been entirely dispensed with, in exchange for weaker types of cross-checks such as "Accept after Delay" ([RFC8078] Section 3.3). An in-band validation method for enabling DNSSEC has been missing.

This document aims to close this gap by introducing an in-band method for DNS Operators to publish arbitrary information about the zones they are authoritative for, in an authenticated manner and on a per-zone basis. The mechanism allows managed DNS Operators to securely announce DNSSEC key parameters for zones under their management. The Parent can then use this signal to cryptographically validate the CDS/CDNSKEY records found at an insecure Child zone's apex, and upon success immediately secure the delegation.

Readers are expected to be familiar with DNSSEC, including [RFC4033], [RFC4034], [RFC4035], [RFC6781], [RFC7344], and [RFC8078].

1.1. Terminology

This section defines the terminology used in this document.

Child The entity on record that has the delegation of the domain from the Parent.

Parent The domain in which the Child in registered.

Child DNS Operator The entity that maintains and publishes the zone information for the Child DNS.

Parental Agent The entity that has the authority to insert DS records into the Parent zone on behalf of the Child. (It could the the registry, registrar, a reseller, or some other authorized entity.)

Signaling Domain A hostname from the Child's NS record set, prefixed with the label _boot. There are as many Signaling Domains as there are distinct NS targets.

Signaling Zone The zone which is authoritative for a given Signaling Domain.

Signaling Name The labels that are prefixed to a Signaling Domain in order to identify a Child zone's name (see Section 2.2).

Signaling Record A DNS record located at a Signaling Name under a Signaling Domain. Signaling Records are used by the Child DNS

Operator to publish information about the Child.

CDS/CDNSKEY This notation refers to CDS and/or CDNSKEY, i.e., one or both.

Base32hex Encoding "Base 32 Encoding with Extended Hex Alphabet" as per [RFC4648].

1.2. 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.

2. Signaling

When setting up initial trust, the Child zone's CDS/CDNSKEY RRsets need to be authenticated. This is achieved using an authentication signal from the Child DNS Operator that the Parent can discover and validate, thus transferring trust from the Child DNS Operator to the Child zone.

2.1. Chain of Trust

If a Child DNS Operator implements the protocol, each Signaling Zone MUST be securely delegated, i.e. have a valid DNSSEC chain of trust.

For example, when performing DNSSEC bootstrapping for a Child zone with NS records ns1.example.net and ns2.example.net, the Child DNS Operator needs to ensure that a valid DNSSEC chain of trust exists for the zone(s) that are authoritative for the Signaling Domains _boot.ns1.example.net and _boot.ns2.example.net.

2.2. Signaling Names

To publish a piece of information about the Child zone in an

authenticated fashion, the Child DNS Operator MUST publish one or more Signaling Records at the Child's Signaling Name under each Signaling Domain.

Signaling Records MUST be accompanied by RRSIG records created with the corresponding Signaling Zone's key(s). The type and contents of these Signaling Records are detailed in Section 3.1.

The Signaling Name MUST consist of the following two labels:

1. the first label of the Child name;

2. a label equal to the SHA-256 hash digest of the fully qualified domain name of the Child's immediate ancestor in the DNS tree (one level up), using wire format for the hash input and "Base 32 Encoding with Extended Hex Alphabet" as specified in [RFC4648] for the output. Trailing padding characters ("=") MUST be dropped.

Note that the "fully qualified domain name of the Child's immediate ancestor in the DNS tree" coincides with the Parent's FQDN only when the delegation is directly (one level) below the Parent's apex. For deeper delegations, it also contains the labels between the Parent and the Child.

For example code, see Appendix B.

[The purpose of the hash function is to avoid the possibility of exceeding the maximum length of a DNS name, and to normalize the number of labels in a Signaling Name. The encoding choice is like in NSEC3, except that SHA-256 is used instead of SHA-1. This is to make it harder for other tenants in shared hosting environments to create hash collisions.]

[Prefixing the first label verbatim minimizes the number of hash calculations that need to be performed by the Child DNS Operator and the Parental Agent, and also facilitates discovery of unprocessed Signaling Records by the Parental Agent by means of NSEC walking the Signaling Domain. (If the first label was part of the hash, the Parental Agent would not be able to infer the Child's name.)]

3. Bootstrapping a DNSSEC Delegation

Child DNS Operators and Parental Agents who wish to use CDS/CDNSKEY records for DNSSEC bootstrapping SHOULD support the protocol described in this section.

3.1. Signaling Consent to Act as the Child's Signer

To confirm its willingness to act as the Child's delegated signer and authenticate the Child's CDS/CDNSKEY RRsets, the Child DNS Operator MUST co-publish them at the corresponding Signaling Name under each Signaling Domain as defined in Section 2.2.

Existing use of CDS/CDNSKEY records is specified at the Child apex only ([RFC7344], Section 4.1). This protocol extends the use of these record types to non-apex owner names for the purpose of DNSSEC bootstrapping. To exclude the possibility of semantic collision, there MUST NOT be a zone cut at a Signaling Name.

Unlike the CDS/CDNSKEY records at the Child's apex, Signaling Records MUST be signed with the corresponding Signaling Zone's key(s). Their contents MUST be identical to the corresponding records published at the Child's apex.

3.1.1. Example

For the purposes of bootstrapping the Child zone example.co.uk with NS records ns1.example.net and ns2.example.net, the required Signaling Domains are _boot.ns1.example.net and _boot.ns2.example.net.

In the zones containing these domains, the Child DNS Operator authenticates the CDS/CDNSKEY records found at the Child's apex by co-publishing them at the names

example.bge2bvlnqt4ei2oq3v9nr8a0lh9nkf6b4lh6c3j51k5kd67helmg._boot.ns1.example.
example.bge2bvlnqt4ei2oq3v9nr8a0lh9nkf6b4lh6c3j51k5kd67helmg._boot.ns2.example.

where example.bge2bvlnqt4ei2oq3v9nr8a0lh9nkf6b4lh6c3j51k5kd67helmg is derived from the Child zone's name example.co.uk as described in Section 2.2. The records are accompanied by RRSIG records created using the key(s) of the respective Signaling Zone.

3.2. Steps Taken by the Parental Agent

[TODO Should this be phrased as an update to [RFC8078], Section 3?]

To validate a Child's CDS/CDNSKEY RRset, the Parental Agent, knowing both the Child zone name and its NS hostnames, MUST execute the following steps:

1. verify that the Child is not currently securely delegated;

2. query the CDS/CDNSKEY records at the Child zone apex directly from each of the authoritative servers listed in the delegation's NS record set;

3. query the CDS/CDNSKEY records located at the Signaling Name under each Signaling Domain using a trusted validating DNS resolver;

4. check (separately by record type) that all record sets retrieved in Steps 2 and 3 have equal contents;

If the above steps succeed without error, the CDS/CDNSKEY records are successfully validated, and the Parental Agent can proceed with the publication of the DS record set under the precautions described in [RFC8078], Section 5.

If, however, an error condition occurs, in particular:

* in Step 1: the Child is already securely delegated;

* in Step 2: any failure during the retrieval of the CDS/CDNSKEY records located at the Child apex from any of the authoritative nameservers, with an empty record set qualifying as a failure;

* in Step 3: any failure to retrieve the CDS/CDNSKEY RRsets located at the Signaling Name under any Signaling Domain, including failure of DNSSEC validation, unauthenticated data (AD bit not set), or an empty record set;

* in Step 4: inconsistent responses;

the Parental Agent MUST abort the procedure.

3.2.1. Example

To verify the CDS/CDNSKEY records for the Child example.co.uk, the Parental Agent (assuming that the Child delegation's NS records are ns1.example.net and ns2.example.net)

1. checks that the Child zone is not yet securely delegated;

2. queries CDS/CDNSKEY records for example.co.uk directly from ns1.example.net and ns2.example.net;

3. queries the CDS/CDNSKEY records located at (see Section 2.2)

example.bge2bvlnqt4ei2oq3v9nr8a0lh9nkf6b4lh6c3j51k5kd67helmg._boot.ns1.example.
example.bge2bvlnqt4ei2oq3v9nr8a0lh9nkf6b4lh6c3j51k5kd67helmg._boot.ns2.example.

4. checks that the CDS/CDNSKEY record sets retrieved in Steps 2 and 3 agree across responses.

If all these steps succeed, the Parental Agent can proceed to publish a DS record set as indicated by the validated CDS/CDNSKEY records.

3.3. Triggers

[Clarity of this section needs to be improved.]

Parental Agents SHOULD trigger the procedure described in Section 3.2 once one of the following conditions is fulfilled:

* The Parental Agent receives a new or updated NS record set for a Child;

* The Parental Agent encounters Signaling Records for its Children during a scan (e.g. daily) of known Signaling Domains (derived from the NS records found in the Parent zone).

  To perform such a scan, the Parental Agent iterates over some or all of its delegations and strips the first label off each one to construct the set of immediate ancestors of its children. (For delegations one level below the Parent's apex, such as second-level domain registrations, this will simply be the name of the Parent zone.) The Parental Agent then uses these names to compute the second label of the Signaling Names as described in Section 2.2. The scan is completed by either

- performing a targeted NSEC walk starting one level below the Signaling Domain, at the label that encodes the Child's ancestor; or by

- performing a zone transfer of the zone containing the (relevant part of the) Signaling Domain, if the Signaling Zone operator allows it, and iterating over its contents.

The Child's name is constructed by prepending the first label of the encountered Signaling Names to the ancestor name from which the Signaling Name's second label was computed;

* The Parental Agent encounters Signaling Records during a proactive, opportunistic scan (e.g. daily queries for the Signaling Records of some or all of its delegations);

* Any other condition as deemed appropriate by local policy.

One of the inputs of the bootstrapping algorithm in Section 3.2 is the NS record set of the Child's delegation. It is therefore necessary to establish knowledge of the delegation's NS record set before firing the trigger.

In some cases, the trigger context contains reliable information about the Child's delegation, such as when bootstrapping is triggered by the registrant changing their NS record set, or during a daily scan of existing delegations. In such cases, the delegation's NS RRset can be used as is.

In cases where the trigger context does not provide sufficient knowledge of the NS record set, the Parental Agent MUST fetch the delegation's NS record set and ensure that the proper NS record set is fed to the bootstrapping algorithm (Section 3.2).

In particular, when discovering Signaling Names by means of an NSEC walk or zone transfer, the Parental Agent MUST NOT assume that the nameserver(s) under whose Signaling Domain(s) a Signaling Name is discovered is in fact authoritative for the corresponding Child. Before firing the trigger for a particular candidate Child, the Parental Agent MUST ascertain that the Child's delegation actually contains the nameserver hostname under whose Signaling Domain the scan occurred.

4. Operational Recommendations

4.1. Child DNS Operator

Signaling Domains SHOULD be delegated as zones of their own, so that the Signaling Zone's apex coincides with the Signaling Domain (such as _boot.ns1.example.net). While it is permissible for the Signaling Domain to be contained in a Signaling Zone of fewer labels (such as example.net), a zone cut ensures that bootstrapping activities do not require modifications of the zone containing the nameserver hostname.

In addition, Signaling Zones SHOULD use NSEC to allow efficient discovery of pending bootstrapping operations by means of zone walking (see Section 3.3). This is especially useful for bulk processing after a Child DNS Operator has enabled the protocol.

To keep the size of the Signaling Zones minimal and bulk processing efficient (such as via NSEC walks or zone transfers), Child DNS Operators SHOULD remove Signaling Records which are found to have been acted upon.

4.2. Parental Agent

It is RECOMMENDED to perform queries within Signaling Domains (Section 3.2) with an (initially) cold resolver cache as to retrieve the most current information regardless of TTL. (When a batch job is used to attempt bootstrapping for a large number of delegations, the cache does not need to get cleared in between.)

[It is expected that Signaling Records have few consumers only, so that caching would not normally have a performance benefit. On the other hand, perhaps it is better to RECOMMEND low TTLs instead?]

5. Implementation Status

Note to the RFC Editor: please remove this entire section before publication.

5.1. Child DNS Operator-side

* Knot DNS supports manual creation of non-apex CDS/CDNSKEY/DNSKEY

records.

* PowerDNS supports manual creation of non-apex CDS/CDNSKEY/DNSKEY records.

* Proof-of-concept Signaling Domains with several thousand Signaling Names exist at _boot.ns1.desec.io and _boot.ns2.desec.org. Signaling Names can be discovered via NSEC walking. Some other operators are planning an experimental implementation.

* A tool to automatically generate signaling records for bootstrapping purposes is under development by the authors.

5.2. Parental Agent-side

* A tool to retrieve and process Signaling Records for bootstrapping purposes, either directly or via zone walking, is available at https://github.com/desec-io/dsbootstrap (https://github.com/desec-io/dsbootstrap). The tool outputs the validated DS records which then can be added to the Parent zone.

* Some registries/registrars are planning exerimental implementations of the protocol.

6. Security Considerations

The protocol adds authentication to the CDS/CDNSKEY-based bootstrapping concept of [RFC8078], while removing nothing. The security level is therefore strictly higher than existing approaches described in that document (e.g. "Accept after Delay"). Apart from this general improvement, the same Security Considerations apply as in [RFC8078].

In case of a hash collision in the second label of the Signal Names, two distinct Child zones may be associated with the same Signaling Name. However, CDS/CDNSKEY mix-up is prevented by the requirement to check signaling records against the "original copy" at the Child's apex. A collision thus produces a mismatch and will impede bootstrapping, but it won't allow an attacker to inject unauthorized key material. The situation is thus equivalent to the traditional bootstrapping model, in that it requires fall-back to another provisioning method. Other mitigations such as salt are thus not

considered necessary.

The level of rigor in Section 3.2 is needed to minimize the risk of publishing a rogue DS RRset. In particular, the various checks ensure that an operator in a multi-homed setup cannot enable DNSSEC unless all other operators agree. [TODO Should this be phrased as a general update to [RFC8078]?]

[Thoughts on the Chain of Trust:

Actors in the chain(s) of trust of the Signaling Zone(s) (the DNS Operator themselves, plus entities further up in the chain) can undermine the protocol. However,

* that's possible with CDS/CDNSKEY, too (new method is not weaker);

* if the Child DNS Operator doesn't trust the zones in which its NS hostnames live (including their nameservers' A records) because their path from the root is untrusted, you probably don't want to trust that operator as a whole;

* when bootstrapping is done upon receipt of a new NS record set, the window of opportunity is very small;

* mitigation exists by diversifying e.g. the nameserver hostname's TLDs, which is advisable anyways;

* correct bootstrapping is easily monitored by the Child DNS Operator.

]

7. IANA Considerations

TODO: reserve _boot?

This document has no IANA actions.

8. Acknowledgements

Thanks to Brian Dickson, Ondřej Caletka, John R. Levine,

Christian Elmerot, and Oli Schacher for reviewing draft proposals and offering comments and suggestions.

Thanks also to Steve Crocker, Hugo Salgado, and Ulrich Wisser for early-stage brainstorming.

9. Normative References

[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>.

[RFC4034] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "Resource Records for the DNS Security Extensions", RFC 4034, DOI 10.17487/RFC4034, March 2005, <https://www.rfc-editor.org/info/rfc4034>.

[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>.

[RFC4648] Josefsson, S., "The Base16, Base32, and Base64 Data Encodings", RFC 4648, DOI 10.17487/RFC4648, October 2006, <https://www.rfc-editor.org/info/rfc4648>.

[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>.

[RFC7344] Kumari, W., Gudmundsson, O., and G. Barwood, "Automating DNSSEC Delegation Trust Maintenance", RFC 7344, DOI 10.17487/RFC7344, September 2014, <https://www.rfc-editor.org/info/rfc7344>.

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

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

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

* draft-thomassen-dnsop-dnssec-bootstrapping-02

| Reframed as an authentication mechanism for RFC 8078.
|
| Removed multi-signer use case (focus on RFC 8078 authentication).
|
| Triggers need to fetch NS records (if not implicit from context).
|
| Improved title.
|
| Recognized that hash collisions are dealt with by Child apex check.

* draft-thomassen-dnsop-dnssec-bootstrapping-01

| Add section on Triggers.
|
| Clarified title.

|
| Improved abstract.
|
| Require CDS/CDNSKEY records at the Child.
|
| Reworked Signaling Name scheme.
|
| Recommend using cold cache for consumption.
|
| Updated terminology (replace "Bootstrapping" by "Signaling").
|

| Added NSEC recommendation for Bootstrapping Zones.
|
| Added multi-signer use case.
|
| Editorial changes.

* draft-thomassen-dnsop-dnssec-bootstrapping-00

| Initial public draft.

Appendix B. Example Code for Computing Signaling Names

Python, with dnspython package:

```
from base64 import b32encode
from hashlib import sha256

import dns.name
from dns.rdtypes.ANY.NSEC3 import b32_normal_to_hex

def compute_signaling_name(child_name):
    child = dns.name.from_text(child_name)
    suffix_wire_format = child.parent().to_wire()
    suffix_digest = sha256(suffix_wire_format).digest()
    suffix_digest = b32encode(suffix_digest).translate(b32_normal_to_hex).rstri
    return dns.name.Name([child[0], suffix_digest.lower()])

signaling_name = compute_signaling_name('example.co.uk.')
print(signaling_name)
# >>> 'example.bge2bvlnqt4ei2oq3v9nr8a0lh9nkf6b4lh6c3j51k5kd67helmg'
```

Authors' Addresses

Peter Thomassen
deSEC, Secure Systems Engineering
Berlin
Germany

Email: peter@desec.io

Nils Wisiol
deSEC, Technische Universität Berlin
Berlin
Germany

Email: nils@desec.io