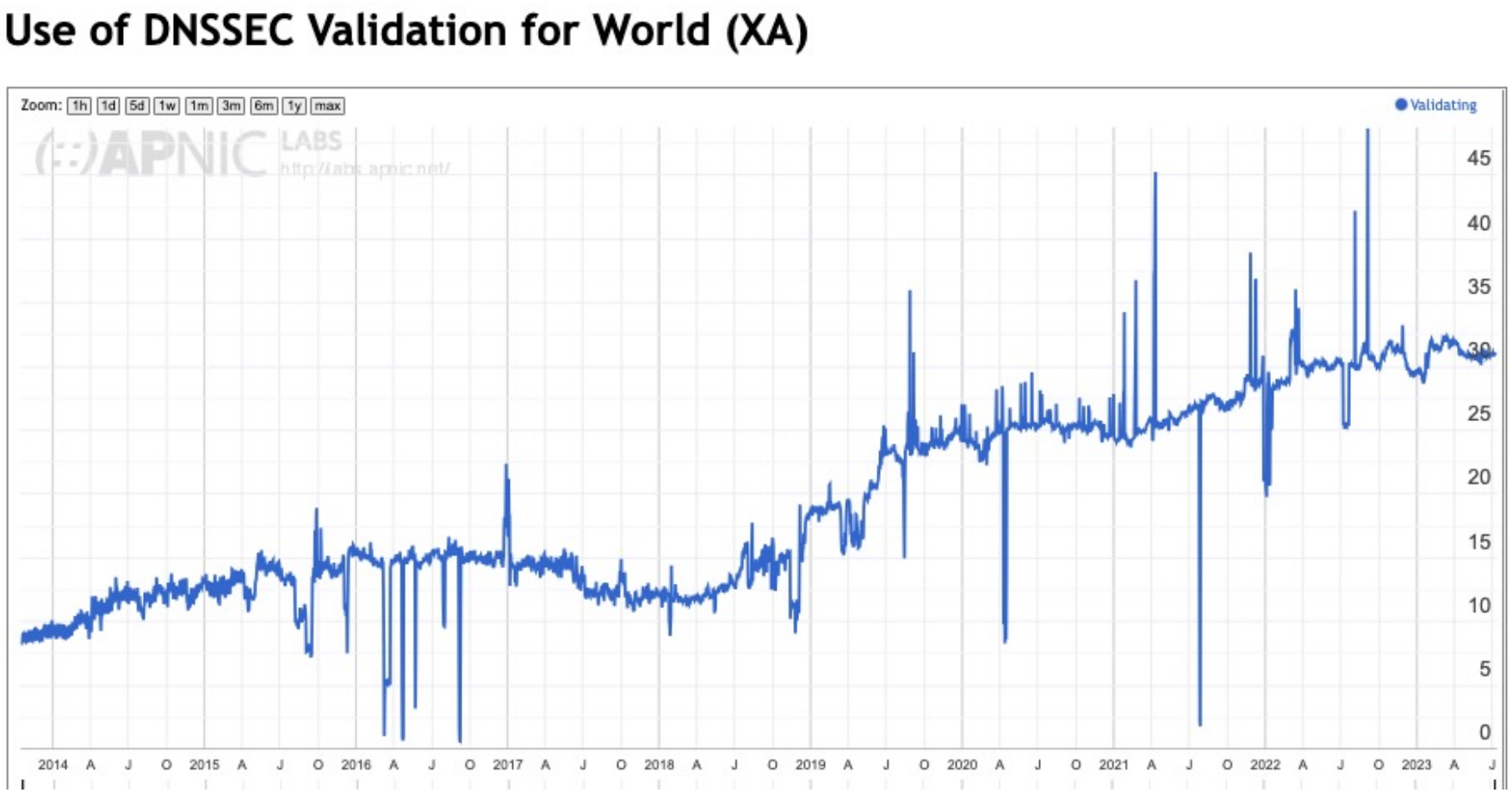
Automating DNSSEC Provisioning

Geoff Huston AM

Chief Scientist, APNIC

DNSSEC adoption is sluggish



Why?

- DNSSEC validation is slow and unreliable
  - Additional DNS queries are necessary to construct the DNSSEC validation chain
  - Signed DNS responses may be large, which can create issues with UDP fragmentation and TCP fallback
- DNSSEC zone signing creates a new set of operational steps
  - Either when using whole-of-zone pre-signing
  - Or front end units that perform signing-on-demand
- DNSSEC provisioning also has an additional delegation step
  - Inserting a DS record in the parent zone

Why?

- DNSSEC validation is slow and unreliable
  - Additional DNS queries are necessary to construct the DNSSEC validation chain
  - Signed DNS responses may be large, which can create issues with UDP fragmentation and TCP fallback
- DNSSEC zone signing creates a new set of operational steps
  - Either when using whole-of-zone pre-signing
  - Or front end units that perform signing-on-demand
- DNSSEC provisioning also has an additional delegation step
  - Inserting a DS record in the parent zone

DS Record?

- The DS record binds the DNSKEY Key Signing Key used in the signed child zone to the parent zone
- The DS record contains the hash of the child zone's DNSKEY KSK value, signed with the parent zone's Zone Signing Key

Example - Parent

NS record is a copy of the delegated zone – not signed

DS record is hash of the child DNSKEY

DS record signature

```
sub.potaroo.net.        6400    IN NS   wattle.rand.apnic.net.
                        6400    DS      2776 15 2 (
                                        502D41D31A14AFD2396D1C8BE67383CCF588
                                        BA8DF19A46603EDAE2D10424CDD5 )
                        6400    RRSIG   DS 13 3 6400 (
                                        20320331235230 20220324225230 41284 potaroo.net.
                                        2NiBkbGrv3vlwQJU3KSekifk6rNk22SPHkrj
                                        xUKpL7XI2C16/M3QLb0PevKTzr/zHS0bDbgF
                                        HAzwHqJjC/te3w== )
                        6400    NSEC    sup.potaroo.net. NS DS RRSIG NSEC
                        6400    RRSIG   NSEC 13 3 6400 (
                                        20320331235230 20220324225230 41284 potaroo.net.
                                        CJodbLauSXGXlkfyIMfMI5O2u9aDGM+4G44N
                                        8VPHUcuupW3LvxD1lzTg+0sjEkU9teklj0zJ
                                        iZ9qe20Lvmrmvg== )
```

Example - Child

```
sub.potaroo.net.  86400    IN SOA   wattle.apnic.net. gih.potaroo.net. (
                                              2021100801 ; serial
                                              10800      ; refresh (3 hours)
                                              3600       ; retry (1 hour)
                                              604800     ; expire (1 week)
                                              10800      ; minimum (3 hours)
                                              )
                           86400    RRSIG    SOA 15 3 86400 (
                                              20311016003557 20211007233557 41535 sub.potaroo.net. mvMe4rgbF
                           86400    NS       wattle.rand.apnic.net.
                           86400    RRSIG    NS 15 3 86400 (
                                              20311016003557 20211007233557 41535 sub.potaroo.net./DuQGdBEIX
                           10800    NSEC     _acme-challenge.sub.potaroo.net. NS SOA RRSIG NSEC DNSKEY
                           10800    RRSIG    NSEC 15 3 10800 (20311016003557 20211007233557 41535 sub.potar
                           86400    DNSKEY   256 3 15 (EQNn2Hp1sPWxC0hHBa5fLwD+bj4TCbgATKBWh+nK1IA=) ;
                                              ZSK; alg = ED25519 ; key id = 41535
                           86400    DNSKEY   257 3 15 (WyprI1oS7+MRLlxIIyZpq5gSzVlSCqRYBATCiJzEPRs=) ;
                                              KSK; alg = ED25519 ; key id = 2776
                           86400    RRSIG    DNSKEY 15 3 86400 (
                                              20311016003557 20211007233557 2776 sub.potaroo.net. S4UQ8Ows5F
                           86400    RRSIG    DNSKEY 15 3 86400 (
                                              20311016003557 20211007233557 41535 sub.potaroo.net.E2MkUyk39r
```

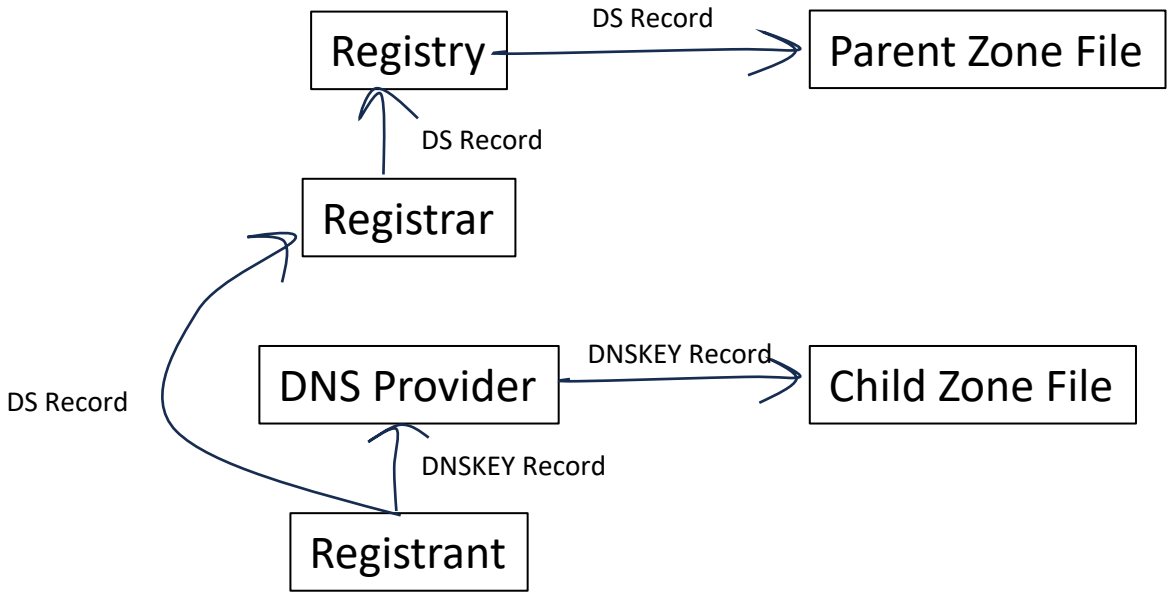
NS record for this zone - signed

KSK record for this zone - signed

Provisioning the DS

- Generate the KSK for the zone
- Generate the hash of the KSK as a DS record
- Pass the hash to the Parent zone admin
  - You could use email or some API to pass the DS record from the **registrant** to the **registrar**
  - Then you could use EPP to pass this DS record from the **registrar** to the **registry**

Infrastructure of today's DNS



- Each of these handovers is potentially vulnerable to attack
- Prefererably all of these handovers use encrypted and authenticated channels

Automating the process

- Can we exploit the signed child zone to automate this process?
- If the data in the child zone can be validated via DNSSEC then the parent can be assured that the published data is current and authentic
- So the child publishes in the child zone and the parent picks up the record, validates it and incorporates it into the parent zone
- To do this, we use a CDS record
  - It’s a signed record in the child zone that contains the hash of the zone KSK

RFC 7344

Internet Engineering Task Force (IETF)
Request for Comments: 7344
Category: Informational
ISSN: 2070-1721

W. Kumari
Google
O. Gudmundsson
OGUD Consulting
G. Barwood
September 2014

Automating DNSSEC Delegation Trust Maintenance

Abstract

This document describes a method to allow DNS Operators to more easily update DNSSEC Key Signing Keys using the DNS as a communication channel. The technique described is aimed at delegations in which it is currently hard to move information from the Child to Parent.

Status of This Memo

RFC 8078

Internet Engineering Task Force (IETF)
Request for Comments: 8078
Updates: 7344
Category: Standards Track
ISSN: 2070-1721

O. Gudmundsson
CloudFlare
P. Wouters
Red Hat

March 2017

Managing DS Records from the Parent via CDS/CDNSKEY

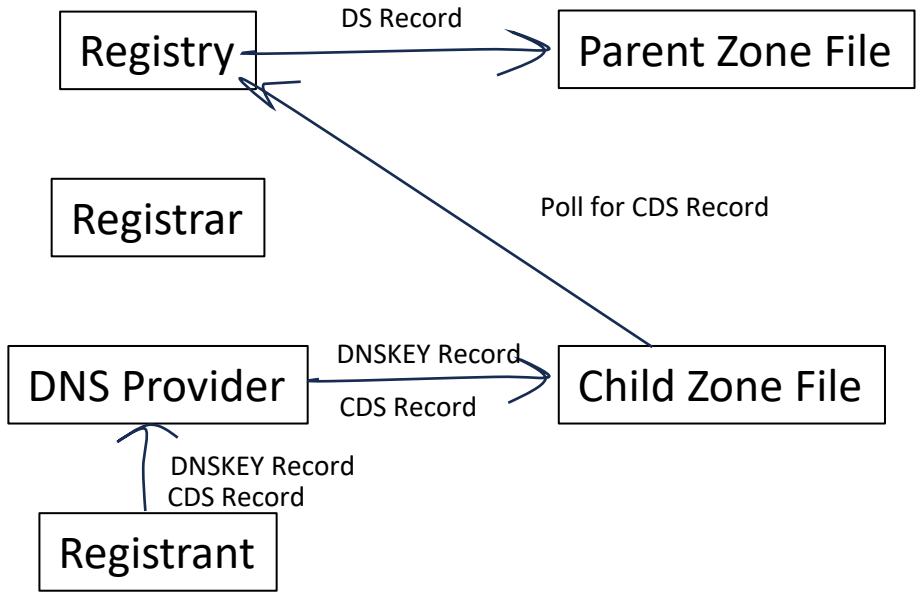
Abstract

RFC 7344 specifies how DNS trust can be maintained across key rollovers in-band between parent and child. This document elevates RFC 7344 from Informational to Standards Track. It also adds a method for initial trust setup and removal of a secure entry point.

Changing a domain's DNSSEC status can be a complicated matter involving multiple unrelated parties. Some of these parties, such as the DNS operator, might not even be known by all the organizations involved. The inability to disable DNSSEC via in-band signaling is seen as a problem or liability that prevents some DNSSEC adoption at a large scale. This document adds a method for in-band signaling of these DNSSEC status changes.

This document describes reasonable policies to ease deployment of the initial acceptance of new secure entry points (DS records).

CDS-driven DNS provisioning



- Parent registry scans the child zone for a CDS record
- Parent DS records (adds/removes) are synchronized against the child-published CDS records

Why do this?

- Removes handling steps to make the DNSSEC process cheaper and easier
- Places control of the domain's DNSSEC status with the registrant
- Can automate bootstrap / key roll / deletion in a single mechanism
- The process could be further automated using dynamic DNS update from child to parent

But

There are some issues:

- CDNSKEY vs CDS records
  - Should the parent generate the hash using a known hash algorithm from the child's DNSKEY records?
  - Or just accept whatever the child used as a hash algorithm to generate the DS record and just publish it?
  - What if the child published both CDS and CDNSKEY and they differ? (one is not the hash of the other)
- Inconsistent DS records
  - Should the parent check **every** child authoritative server to ensure that they all publish the same DS record set?
- Polling
  - How often should the parent poll the child zone?
- Bootstrap from insecure to secured
  - How do you accept the initial DS record?

Who does CDS today?

DNS providers supporting CDS

Provider	CDS	CDNSKEY	Delete
Cloudflare	✓	✓	✓
DNSimple	✓	✓	
GoDaddy	✓	✓	
Google Domains			

Who does CDS today?

Registries supporting CDS

Registry	CDS	CDNSKEY	Delete	Bootstrap from insecure	Notes
.cz	✗	✓	✓	7 days TCP-only	FRED is used
.cr	✗	✓	✓	7 days TCP-only	No info found; FRED is used
.ch	✓	✗	✓	72 hours TCP-only	
.li	✓	✗	✓	72 hours TCP-only	
.sk	✓	✗	✓	72 hours	No clear information about using TCP for bootstrap
RIPE NCC	✓	✗	✓	No support	

Experiences

https://blog.apnic.net/2021/11/02/dnssec-provisioning-automation-with-cds-cdnskey-in-the-real-world/

https://ripe82.ripe.net/wp-content/uploads/presentations/62-Deployment_of_CDS.pdf

Thanks!