

Security Challenges for Domain Names

APTLD84

September 19, 2023

Peter Thomassen

<peter@desec.io>

Bio

— — —

- Originally, a particle physicist
- Today: **trying to making DNS better**
- Created deSEC (<https://desec.io/>)
 - DNS hosting platform with security by design (full DNSSEC automation, ~33,000 zones)
 - Of course, people use it because of great UX ;-)
- IETF
 - Working on standards for automated DNS operations (catalog zones) and automated DNSSEC
- Member of ...
 - ICANN SSAC (Security and Stability Advisory Committee)
 - Advisory Board at Internet Namespace Security Observatory

DNS Security Issues

— — —

- What security issues are you concerned about?
 - Registry security?
 - DNS manipulations?
 - Transport security?
 - Post-quantum cryptography?
 - ...
- Please **collect topics of interest** (→ Zoom chat)
- This talk: Focus on **DNS manipulations**
 - First, hijacking of expired nameserver domains
 - Then, issues related to DNSSEC deployment

➤ We need a **trustworthy infrastructure**

Dangling NS Records (pointing at expired names) [Case 1]

— — —

Assumptions:

- *.net* registry uses EPP
- *example.net* has two NS records: *ns.provider.org* and *ns.provider.net*

What happens if provider.org expires?

- Someone else can register it
- Allows new domain owner to hijack (part of the) DNS traffic ☠️

Solution (?)

- Remove expired NS name *ns.provider.org* from *example.net* delegation
- But how does *.net* registrar/registry know when *provider.org* expired?

Dangling NS Records (pointing at expired names) [Case 2]

And what happens if provider.net expires?

- EPP does not allow deleting *provider.net* as long as *example.net* depends on it
→ registrar who sold *provider.net* **continues to pay \$\$\$**

Solution (?)

- Rename “*provider.net* host object” to different TLD → removes dependency
- But: allows hijacking the renamed nameserver domain! 💀

⚠ **These things have happened.** [Details available in research paper.](#) ⚠

512k domains exposed in 2011–2020, and **163k actually hijacked**
Including accidental deletion and renaming of *tiktok.com*’s nameserver domain

Avoid Dangling NS Records!

— — —

Several **suboptimal solutions** found in the wild:

- Defensively register expired NS names (costs \$\$\$)
- Notify domain owners of expired NS names (does not solve the problem itself)
- Rename expired NS names to sinkhole nameservers (requires infrastructure)
 - Might also expire → worse than before
- Rename to reserved TLD (e.g. *.invalid*)

Better:

- Change EPP specification: **allow and even encourage deletion** of host objects
 - Reduces incentive to create dangling configurations
 - [RFC draft](#) is progress
 - Also, SSAC report coming up

**How do you handle expired NS names?
Any other thoughts?**

How to prevent DNS Manipulations?

— — —

By adding **digital signatures** to DNS records, **DNSSEC** prevents ...

- DNS cache poisoning
- intervention by malicious actors (e.g. state-sponsored, criminals)

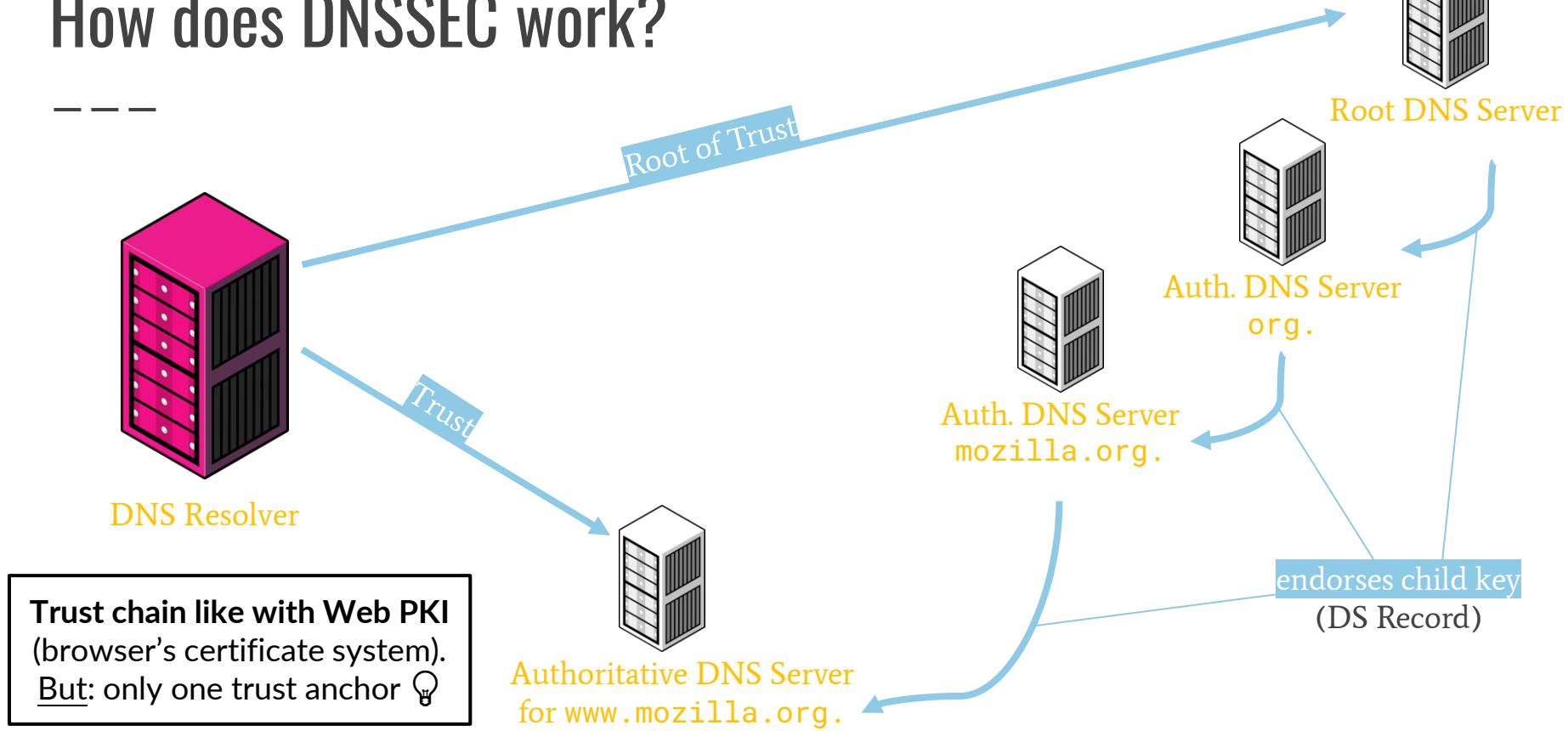
Makes **other attacks much less attractive**:

- Nameserver takeover (e.g. NS hijacking at registry: [Sea Turtle attack](#) in 2019)
- Routing attacks (e.g. BGP hijacking: [Amazon Route 53 takeover](#) in 2018)

Provides **technical foundation for trust applications**:

- [4M email domains](#) protected with [DANE](#) (email transport security)
- [Digital Trust Registries](#) with global chain of trust (e.g. global ID card checks!)

How does DNSSEC work?



DNSSEC validation rate

31 %

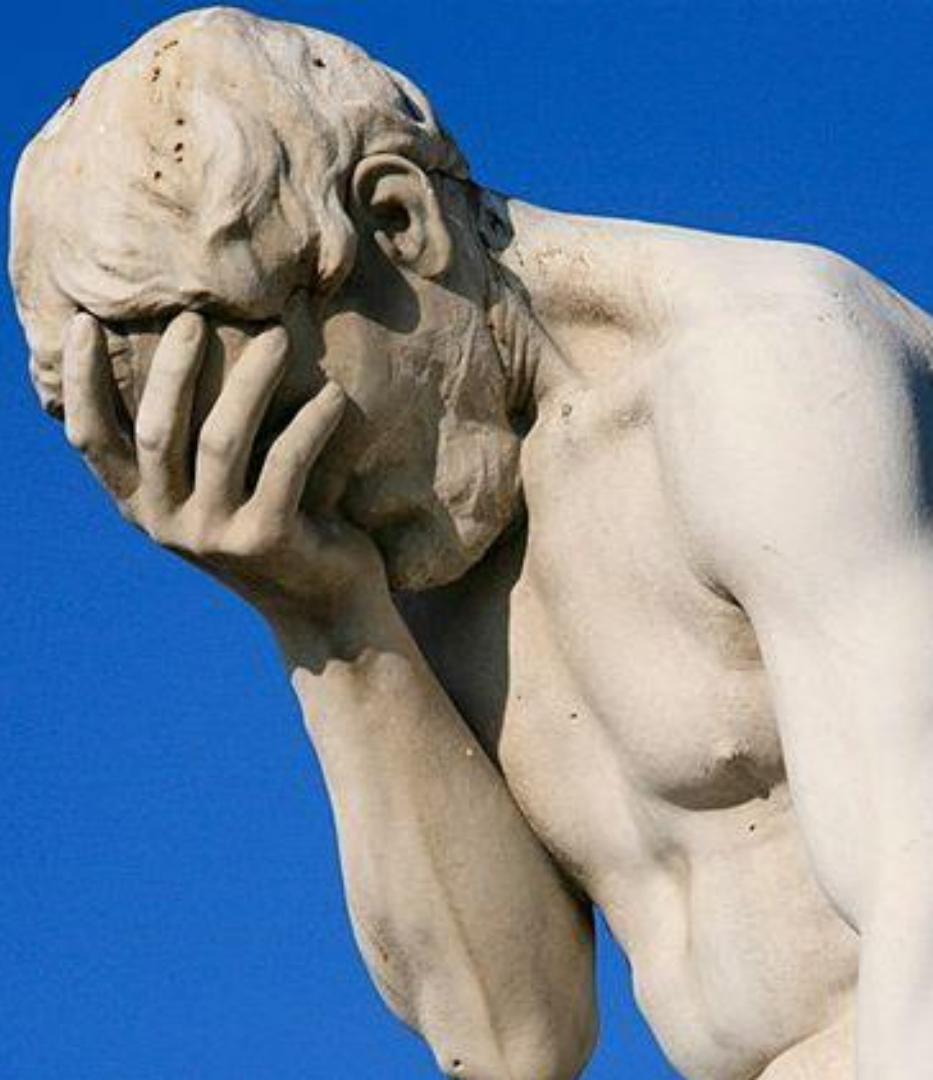
vs.

Domains using DNSSEC

8 %

- Asia 28%
- Oceania 43%
- Africa 31%
- Americas 33%
- Europe 40 %

- That's 23M domains
- Fraction of **signed** zones is higher!
- But < 50% have **DS records** ...



DNSSEC is too hard

(and scary?)

Don't be afraid.

— — —

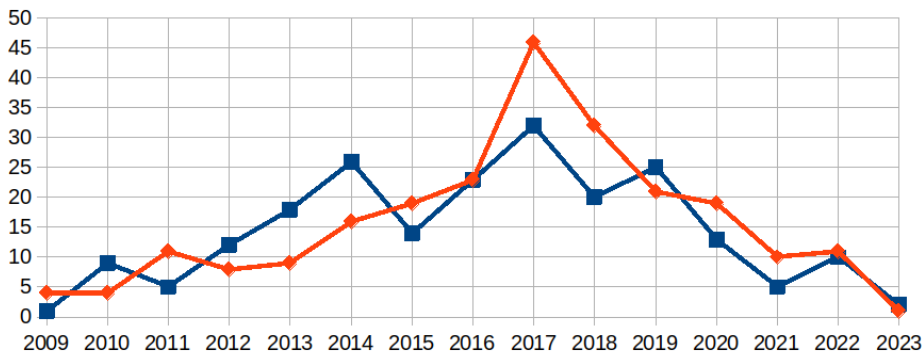
- Initially (> 6 years ago): **Manual signing, manual key maintenance**
- Today: **Tooling available** for all of that
 - Knot DNS supports automatic key rotation (including waiting for cache expiration etc.)
 - ... and even automatic change of algorithm
- Automation greatly **reduces overhead**
 - deSEC runs a registry with 20k registrations (at dedyn.io), **all with DNSSEC, very little overhead**
- Number of **deployments increasing**, number of **incidents decreasing**
 - $\text{risk} = \text{\#incidents} / \text{\#deployments} \rightarrow \text{risk decreasing}$
- Keep in mind: **There's No Perfection** (\leftrightarrow frequency of **invalid certificates!**)
 - Some things still go wrong. But when fixed *in the tooling*, it won't happen again!
 - Example: In 2021, [Amazon Route 53's buggy DNSSEC implementation broke slack.com](#) (fixed!)

DNSSEC Incidents vs. Deployment Size

DNSSEC outages (yearly)

<https://ianix.com/pub/dnssec-outages.html> (2023-06-26)

■ TLDs ◆ Major Sites

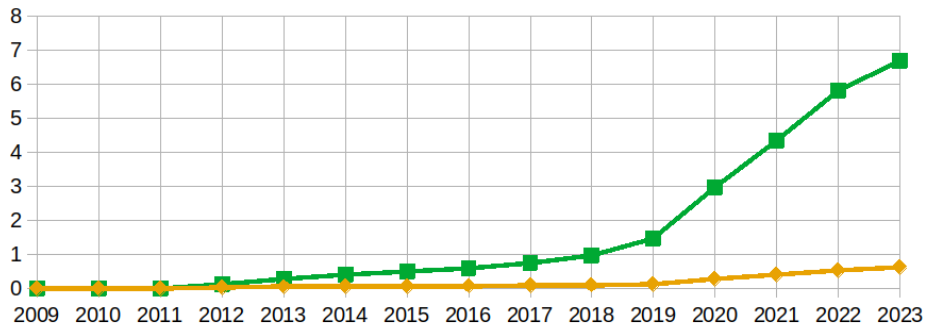


<https://ianix.com/pub/dnssec-outages.html>

Domain Names with DS Records (millions; yearly)

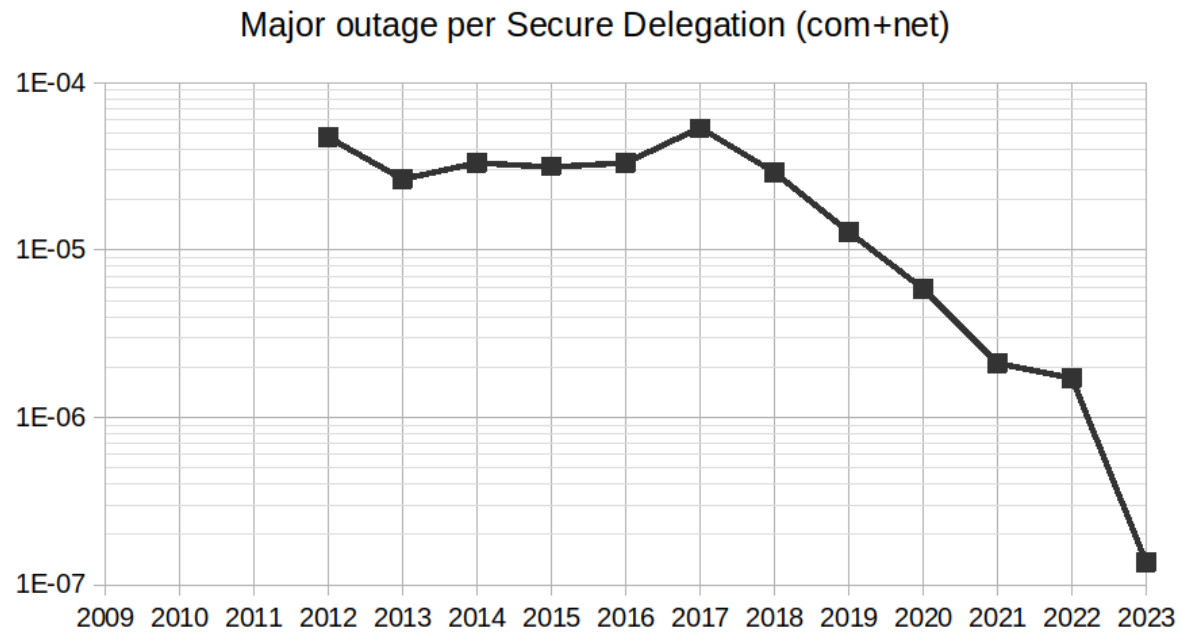
https://www.verisign.com/en_US/company-information/verisign-labs/internet-security-tools/dnssec-scoreboard/index.xhtml (2023-06-26)

■ com DS ◆ net DS



<https://scoreboard.verisignlabs.com/>

Estimated Deployment Risk of DNSSEC Decreased 31x



This is the ratio of the two previous diagrams.
 (2023 numbers until June.) – Note: **logarithmic** scale!

Phew. First part is over. – Ě ĭ Ĥ Ĳ Ĭ Ħ Ĩ Ĳ

So, why is DNSSEC deployment so low?

— — —

- Turning on DNSSEC requires 2 steps:
 1. Get the domain's DNSSEC parameters from the signer (usually: child DNS provider)
 2. Put these parameters into the parent (DS records next to delegation's NS records)
- **Shortcut possible when DNS provided by domain registrar** (can do both steps)
- But: Many domains use third-party DNS provider
 - The **DNS provider cannot login to the registry/registrar** to update the DS records
 - **Domain owners have to do this manually**
- Many **domain owners don't know** about this possibility at all
 - And if they know, they might not have the necessary knowledge
 - It is **error-prone** ([40% of domain owners who make an attempt don't succeed](#))
 - It is **time-consuming** (especially for many domains)
 - People are used to smooth services → **need for human intervention is unexpected**

Why do some ccTLDs have high DNSSEC deployment?

— — —

- Some ccTLDs have high DNSSEC deployment (> 50%)
 - Examples: **.cz** (Czech Republic), **.nl** (Netherlands), **.no** (Norway), **.se** (Sweden), **.nu** (Niue)
- **Why is that?**
- Two reasons:
 1. **.nl** gives **discount on registration fees** when DNSSEC is enabled
 2. Some registries implement **automatic DS record maintenance**
- Advantages of automation:
 - You don't need to give a discount :-)
 - Also allows **automatic maintenance** (e.g. changing keys / algorithms etc.)

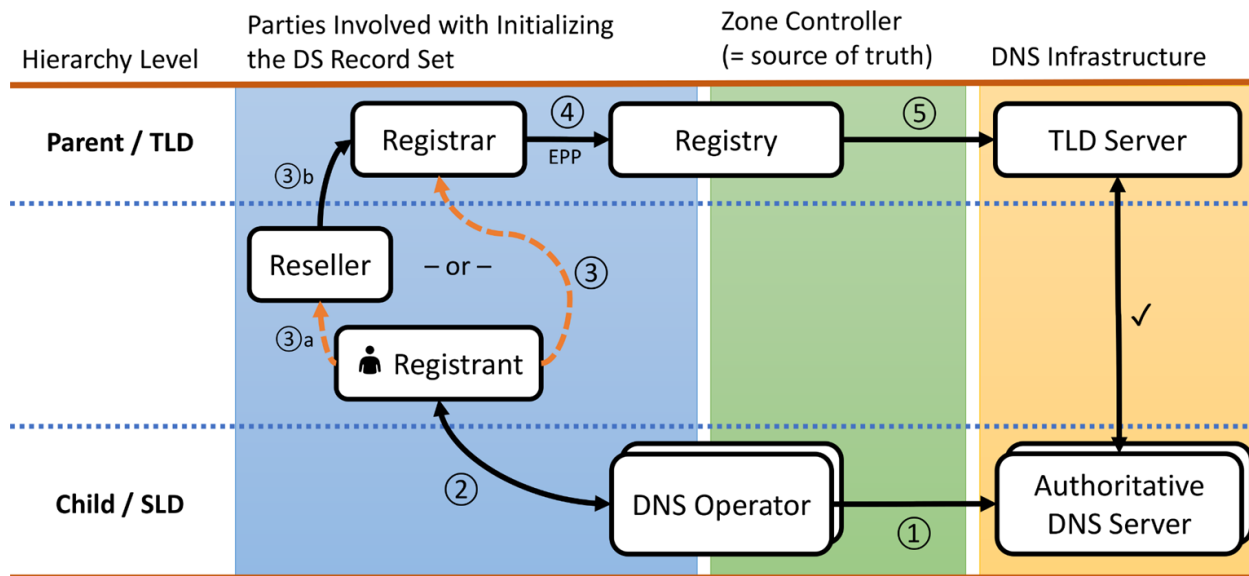
Approaches to DS Record Configuration

— — —

1. manual submission

- Generally supported, but cumbersome

Approaches: Manual Submission



- Slow
- Error-prone
- Out of band
- Not properly authenticated

- Involves the child DNS provider (origin) and TLD registry (recipient)
 - ... often with the registrar as the messenger
 - ... usually coordinated by the domain owner (registrant)

Approaches to DS Record Configuration

— — —

1. manual submission

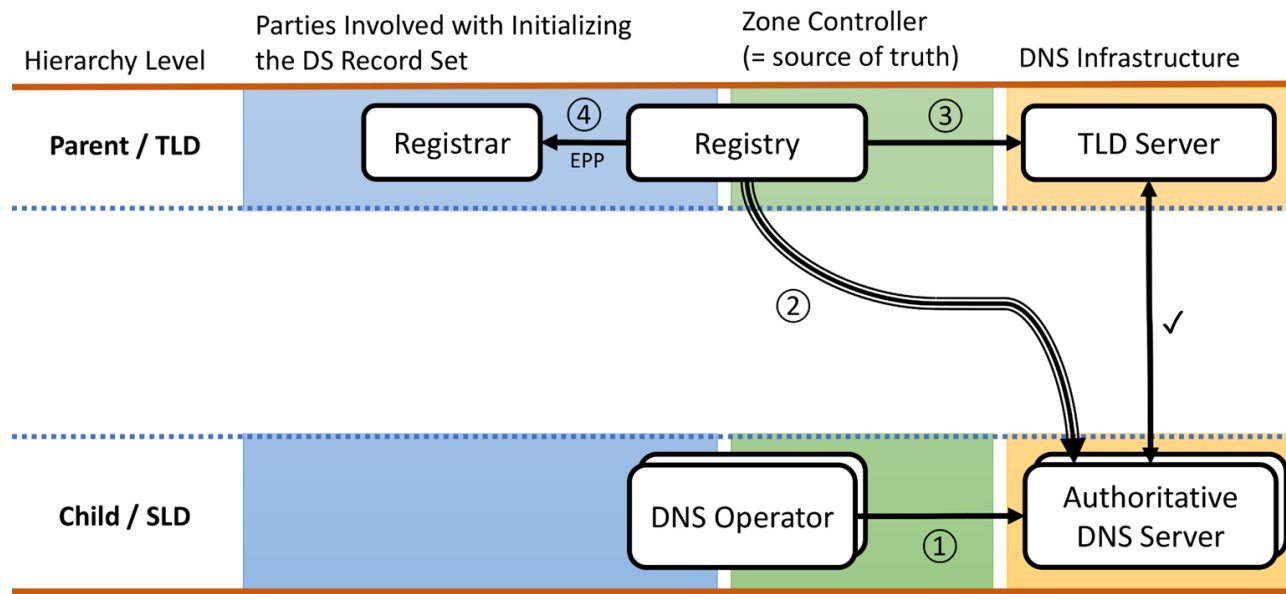
- Generally supported, but cumbersome

2. **trust on first use (TOFU)**: query DNSKEY, compute DS, and hope for the best

- Used by notable registrar in Germany

Approaches: Trust on First Use (various interfaces)

- No manual dealing with cryptographic parameters
- Known timing
- No authentication!



Approaches to DS Record Configuration

— — —

1. **manual submission**
 - Generally supported, but cumbersome
2. **trust on first use (TOFU)**: query DNSKEY, compute DS, and hope for the best
 - Used by notable Registrar in Germany
3. Several attempts on **REST interfaces** or REST-DNS hybrids, driven by CIRA
 - ICANN [53](#), [54](#) (2015), [draft-ietf-regext-dnsoperator-to-rrr-protocol](#) (2018)
 - No known deployments

“Need to redesign around the DNS Operator”

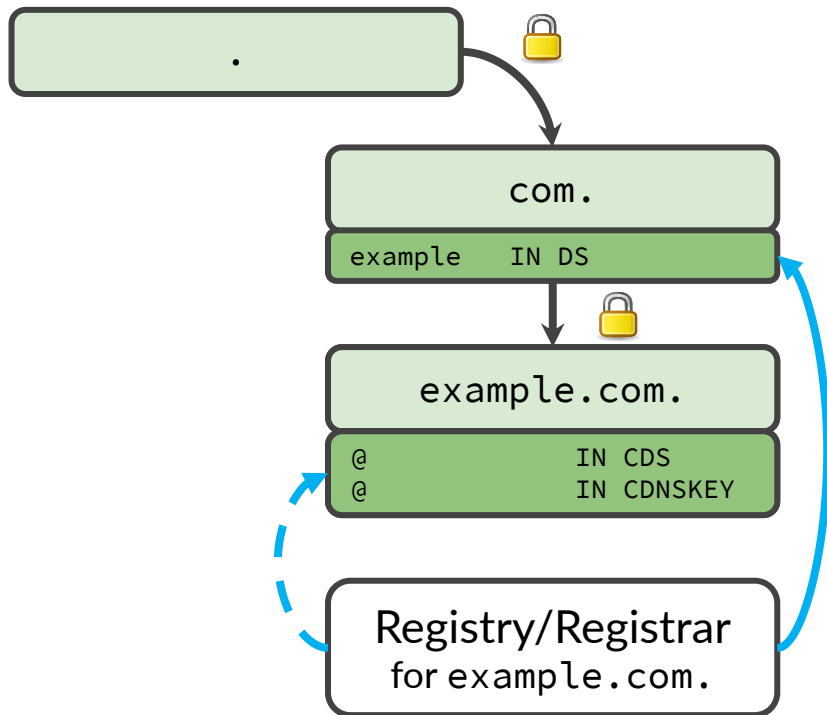
— Jacques Latour, Tech Day at ICANN 53

Approaches to DS Record Configuration

— — —

1. **manual submission**
 - Generally supported, but cumbersome
2. **trust on first use (TOFU)**: query DNSKEY, compute DS, and hope for the best
 - Used by notable Registrar in Germany
3. Several attempts on **REST interfaces** or REST-DNS hybrids, driven by CIRA
 - ICANN [53](#), [54](#) (2015), [draft-ietf-regext-dnsoperator-to-rrr-protocol](#) (2018)
 - No known deployments
4. **CDS/CDNSKEY from insecure child (RFC 8078)**
 - Requires extra checks for initial DS records (**updates use existing key!**)
 - [Used](#) by [.ch/.cr/.cz/.fo/.li/.nu/.se/.sk/.alt.za/.edu.za](#) (parent) and various DNS operators (child)

Approaches: CDS/CDNSKEY from Insecure Child



Approaches to DS Record Configuration

— — —

1. manual submission

- Generally supported, but cumbersome

2. **trust on first use** (TOFU): query DNSKEY, compute DS, and hope for the best

- Used by notable Registrar in Germany

3. Several attempts on **REST interfaces** or REST-DNS hybrids, driven by CIRA

- ICANN [53](#), [54](#) (2015), [draft-ietf-regext-dnsoperator-to-rrr-protocol](#) (2018)
- No known deployments

4. CDS/CDNSKEY from **insecure child** (RFC 8078)

- Requires extra checks for initial DS records (**updates use existing key!**)
- Used by [.ch/.cr/.cz/.fo/.li/.nu/.se/.sk/.alt.za/.edu.za](#) (parent) and various DNS operators (child)

5. **CDS/CDNSKEY with authentication** by child operator ([RFC draft](#))

- Used by [.ch/.li](#) (parent) and **Cloudflare/deSEC/Glaucia HexDNS** (child)
- Implementations exist for PowerDNS and Knot DNS (not yet upstream)

Approaches: CDS/CDNSKEY with Authentication

— — —

1. Define a **signaling mechanism for DNS operators**

- allow **publishing arbitrary information** about the zones under management, **on a per-zone basis**
- do so using namespace **under each nameserver hostname** with **zone-specific subdomains**
- **require DNSSEC for authentication** (requires nameserver domains to be secure)

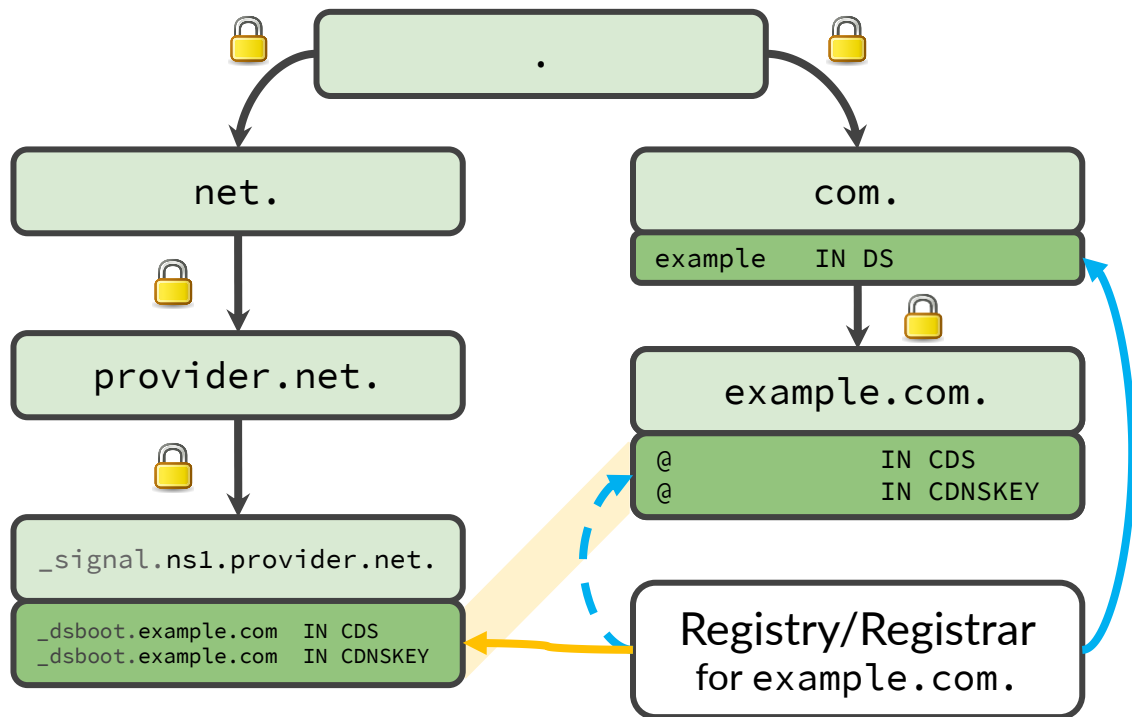
2. Ask DNS Operators to **publish authentication signal** for CDS/CDNSKEY

- start with conventional **CDS/CDNSKEY records** at the apex of the target zone (RFC 8078)
- **co-publish** these records **via signaling mechanism** (signed with NS zone's keys)

3. **Validate** target domain's CDS/CDNSKEY records **against this signal**

- if successful: **“transfer trust to the target domain”**
→ **provision DS records** at parent

Authentication: Co-Publication from Safe Grounds



💡 Use an **established chain of trust** (left) to take a detour

- identically co-published
- authenticated, immediate
- no active on-wire attacker



It's already in Production

— — —

Parent:

- **2 ccTLDs:** .ch/.li (+ .cl testing)
- gTLDs in ICANN process to ensure consistent behavior
- GoDaddy to introduce automatic **automated DS maintenance as a registrar**

Child:

- **3 DNS operators**, for all DNSSEC-enabled domains
 - Cloudflare (manages **23% of Top 1M domains**)
 - Glauca HexDNS
 - deSEC

Main Takeaways: Why Automated DS Maintenance is Good

— — —

Why DNSSEC?

- **Mitigates attacks** (cache poisoning, registry manipulations, BGP hijacks, ...)
- Enables **innovative solutions** under your TLD: DANE, digital identities, ...

Why DS automation?

- DNSSEC deployment suffers from difficult / manual configuration
- **Automated DS maintenance is crucial** for moving this forward
- Used at about 10 TLDs, more under way → **works well in practice**

Bonus:

- **Multi-provider setups** with automation (supported e.g. by Cloudflare, deSEC)
- **Automatic provider transfer** (also needs implement CSYNC, RFC 7477)

Further Considerations

— — —

Registries deploying DNSSEC need to consider some **operational topics**:

- Accept DS records directly, or accept DNSKEY records and compute DS?
 - Better: **accept DS directly**
- For automated DS maintenance, perhaps **scaling** is a problem?
 - It is not: there are deployments with **20M scans/day on single virtual machine**
- Don't make DS record TTLs too long (**1 hour is good**, 1 day too long)
 - Helps the child to fix mistakes quickly, [no noticeable impact on DNS traffic](#) (recently measured)
- Further aspects: **SSAC report & recommendations** coming up (October 2023)
 - **Acceptance checks** (e.g., CDS consistency across nameservers → helps **detect dangling NS!**)
 - Considerations on registration locks, error reporting, etc.
 - **Interested?** – Let me know, and I will send you a copy (→ peter@desec.io)

You are invited!

— — —

- DNSSEC is alive: > 30% validation rate, ~10% signing rate
 - In last 12 months, 8 ccTLDs enabled DNSSEC (.ci, .bj, .kz, .ht, .bd, .tr, .ph, .ec got a DS record)
- Automated DS maintenance
 - **deployed** at DNS operators
 - **deployed** at about 10 ccTLD registries
 - **more to come:** including [all TLDs under CentralNIC management](#) (.bh, .fm, .gd, .gl, .la, .pw., .vg)
- **It would be great to see implementations in the AP region** 🌟
 - **Give it a try:** set up an experiment! Do some testing!
 - You can get assistance from the DNS community
- Let's make DNSSEC easy.

Thank you!

... also to our supporters:



Questions?

Dr. Peter Thomassen

peter@desec.io

peter.thomassen@securesystems.de

Backup

— — —

Authenticated DNSSEC Bootstrapping: Protocol Details

Algorithm

- **Co-publish CDS/CDNSKEY records** under a subdomain of the NS hostnames:
→ CDS/CDNSKEY IN `_dsboot.example.com._signal.ns1.provider.net`
- Use **DNSSEC to validate** these records, under **each NS hostname**

Technical Considerations

- Naming scheme with `_signal` label allows delegating to separate zone
 - removes risk of accidentally modifying the nameserver's A/AAAA records
 - reduces churn on nameserver zone
 - allows splitting off DNS operations (e.g. online-signing with different key; delegate by parent)
- prefix allows different types of signals (e.g. for multi-signer p2p key exchange)