dry-run DNSSEC

draft-yorgos-dnsop-dry-run-dnssec

Yorgos Thessalonikefs, Willem Toorop, Roy Arends

IETF 114
Origin story

- Extended DNS Errors (RFC8914)
  - nice
- DNS Error Reporting (draft-ietf-dnsop-dns-error-reporting)
  - very nice
Origin story

Random lunch discussion

- “DNS Error Reports can’t help me if I want to adopt DNSSEC”
- Hmmm…
- Hmmm…
- “DMARC!”
Enter dry-run DNSSEC

From:

Gunshow, by KC Green
Enter dry-run DNSSEC

To:

_u/GameNCode_ on reddit
Enter dry-run DNSSEC - How it works

- Zone is signed and published
- A dry-run DS record is published in the parent
- Resolver is signaled (dry-run DS) that zone is dry-run signed
- If validation fails generate DNS Error Report; fallback to non-dry-run DS record
  - Let’s pretend that never happened
- If validation succeeds return AD bit (opportunistic security)
Use cases - DNSSEC adoption

- Main goal of the proposal
- In the wild testing
- Provide confidence to operators that the newly signed zone is not breaking DNS
- Turn-key action to deploy: replace the dry-run DS with the real DS; no need to touch the zone
Use cases - DNSSEC experimentation

- You can experimentally sign your zone in the wild!
- See what validating resolvers have to say about it
Use cases - Test key rollovers

- Real DS also as dry-run DS
- Sign and introduce the new key with a dry-run DS
- … do key rollover stuff …
- If everything worked, great! Repeat with real DSes this time.
Break it! (AKA end-to-end testing)

- Clients can opt-in (with new EDNS0) to receive dry-run DNSSEC errors (if any)
- Easier debugging from the client side
- Test how an application will behave in case of errors
Break it! (AKA end-to-end testing)

u/leolambertini on reddit
Signaling - IETF 113 Feedback

- Use flags in DNSKEY instead of DS-hack
  - DNSKEY RRset needs to change when done testing; no turn-key action
- General purpose DS-hack for all the DS-hacks
  - Maybe, but we perceive dry-run DNSSEC as integral part of DNSSEC if adopted, not a DS-hack
- Normalize the different DS-hacks with delegation RR types on the parent (like DS)
  - Yes please! But this is another draft…
  - btw, this could work like DDS (new type) identical DS data
Signaling - Two timelines

Single timeline

| 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 3 |
| 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 |

```
+----------------------------------+
<table>
<thead>
<tr>
<th>Key Tag</th>
<th>Algorithm</th>
<th>DRY-RUN</th>
</tr>
</thead>
</table>

+----------------------------------+
| Digest Type |                        |
+-------------+------------------------|
| Digest      |                        |
|             |                        |
```

- Variable length digest
  - Q: How bad is this?
Signaling - Two timelines

Multiple timeline

● Equivalent dry-run DS algorithm for each real DS algorithm
● Essentially burning a bit of the DS digest field (4 currently assigned)
● Q: Can we afford this?
Signaling - Backwards compatibility

Yes

In all timelines, resolvers that do not support dry-run DNSSEC and have no knowledge of the introduced DS Digest Type Algorithms ignore it as per RFC6840, section 5.2
Provisioning

● Parent accepts DS? Great
● Parent accepts only DNSKEY?
  ○ Get the dry-run intent
  ○ Either with accompanying DS
  ○ Or other means (e.g., UI)
● CDS works
● CDNSKEY needs accompanying CDS
Security caveat

- No data integrity for the DNSSEC adoption use case!
- In case of DNSSEC errors (spoofing attacks) the resolver will fallback to insecure
  - Feature not a bug
- Warning that dry-run DNSSEC is a temporarily intermediate step of a zone going secure
Implementation

DNS Error Reporting in Unbound (Hackathon 114; early stage) and this could be the next step
dry-run DNSSEC

Feedback / Questions / (Answers) ?