

Automating DNS Delegation Management via DDNS

`draft-ietf-dnsop-delegation-mgmt-via-ddns-01`

Johan Stenstam Erik Bergström Leon Fernandez

The Swedish Internet Foundation

March 2026

Where We Are

The draft was adopted as a WG document after IETF 124.

- -00 was a straight re-submission after adoption.

Where We Are

The draft was adopted as a WG document after IETF 124.

- -00 was a straight re-submission after adoption.
- -01 (current) has substantive improvements:
 - ▶ Replaced hand-waving “Mutual Authentication” section with a concrete specification for the UPDATE Receiver’s SIG(0) key.
 - ▶ Restructured child–parent communication sections (motivation before solution).
 - ▶ Replaced obsolete EDNS(0) KeyState policy inquiry with the simpler SVCB bootstrap SvcParamKey.
 - ▶ Expanded terminology (UPDATE Receiver, Bootstrap).
 - ▶ Large number of editorial improvements (thanks Yorgos Thessalonikefs).

Recap: What Does This Draft Do?

A mechanism for child zones to manage their delegation data in the parent zone via DNS UPDATE, secured with SIG(0):

- 1 Child discovers parent's UPDATE Receiver via DSYNC record (RFC 9859).
- 2 Child sends SIG(0)-signed DNS UPDATE containing the exact change (NS, DS, glue).
- 3 UPDATE Receiver verifies signature, applies policy, updates parent-side delegation information.

Recap: What Does This Draft Do?

A mechanism for child zones to manage their delegation data in the parent zone via DNS UPDATE, secured with SIG(0):

- 1 Child discovers parent's UPDATE Receiver via DSYNC record (RFC 9859).
- 2 Child sends SIG(0)-signed DNS UPDATE containing the exact change (NS, DS, glue).
- 3 UPDATE Receiver verifies signature, applies policy, updates parent-side delegation information.

Key properties:

- Works for both DNSSEC-signed **and unsigned** child zones.
- No scanners needed in the parent.
- Efficient: child pushes changes, parent only validates.

Trust Bootstrapping — The Interesting Part

How does the parent learn to trust the child's SIG(0) public key?

Four mechanisms, depending on the child's DNSSEC status:

- at-apex** Child zone is DNSSEC-signed. Look up KEY at child apex, DNSSEC-validate.
- at-ns** At least one NS is in a signed zone (RFC 9615 model). Look up KEY at `_sig0key.{child}._signal.{ns}.`, DNSSEC-validate.
- unsigned** RFC 8078 model: look up from multiple vantage points, multiple times. Best-effort.
- manual** Out-of-band (email, web form, etc.).

Trust Bootstrapping — The Interesting Part

How does the parent learn to trust the child's SIG(0) public key?

Four mechanisms, depending on the child's DNSSEC status:

at-apex Child zone is DNSSEC-signed. Look up KEY at child apex, DNSSEC-validate.

at-ns At least one NS is in a signed zone (RFC 9615 model). Look up KEY at `_sig0key.{child}._signal.{ns}.`, DNSSEC-validate.

unsigned RFC 8078 model: look up from multiple vantage points, multiple times. Best-effort.

manual Out-of-band (email, web form, etc.).

Parent announces supported methods via SVCB at the DSYNC target:

```
updater.parent. IN SVCB 0 . ( bootstrap="at-apex,at-ns>manual" )
```

Mutual Authentication (New in -01)

Problem: the child needs to verify that responses from the parent are authentic.

Solution: UPDATE Receiver also maintains a SIG(0) key pair and publishes its KEY at the DSYNC target name:

```
_dsync.parent.      IN DSYNC ANY UPDATE 5354 updater.parent.  
_dsync.parent.      IN RRSIG DSYNC ...  
updater.parent.     IN KEY ...  
updater.parent.     IN RRSIG KEY ...
```

Mutual Authentication (New in -01)

Problem: the child needs to verify that responses from the parent are authentic.

Solution: UPDATE Receiver also maintains a SIG(0) key pair and publishes its KEY at the DSYNC target name:

```
_dsync.parent.    IN DSYNC ANY UPDATE 5354 updater.parent.  
_dsync.parent.    IN RRSIG DSYNC ...  
updater.parent.   IN KEY ...  
updater.parent.   IN RRSIG KEY ...
```

This enables mutual authentication:

- Child → Parent: SIG(0) signature on UPDATE
- Parent → Child: SIG(0) signature on response

Especially important for KeyState inquiries, where a forged “key unknown” response could trigger unnecessary re-bootstrapping .

KeyState Communication

(draft-berra-dnsop-keystate)

Extended DNS Errors (RFC 8914) can augment error responses but:

- Can only *return* information, not *send* inquiries.
- Cannot carry structured data like a Key ID.

KeyState Communication

(draft-berra-dnsop-keystate)

Extended DNS Errors (RFC 8914) can augment error responses but:

- Can only *return* information, not *send* inquiries.
- Cannot carry structured data like a Key ID.

KeyState EDNS(0) option allows child to inquire about current key state

- Parent returns SIG(0)-signed response: “Trusted” / “Unknown” / “Bootstrap ongoing”

```
# tdns-cli parentsync inquire update --zone whisky.dnslab
```

```
KeyState Inquiry for child whisky.dnslab.
```

```
KeyID:      38193
```

```
Parent says: Trusted (code 4)
```

```
Extra:      Key state: Trusted
```

CLI here only
as an example

KeyState Communication

(draft-berra-dnsop-keystate)

Extended DNS Errors (RFC 8914) can augment error responses but:

- Can only *return* information, not *send* inquiries.
- Cannot carry structured data like a Key ID.

KeyState EDNS(0) option allows child to inquire about current key state

- Parent returns SIG(0)-signed response: “Trusted” / “Unknown” / “Bootstrap ongoing”

```
# tdns-cli parentsync inquire update --zone whisky.dnslab
```

```
KeyState Inquiry for child whisky.dnslab.
```

```
KeyID:      38193
```

```
Parent says: Trusted (code 4)
```

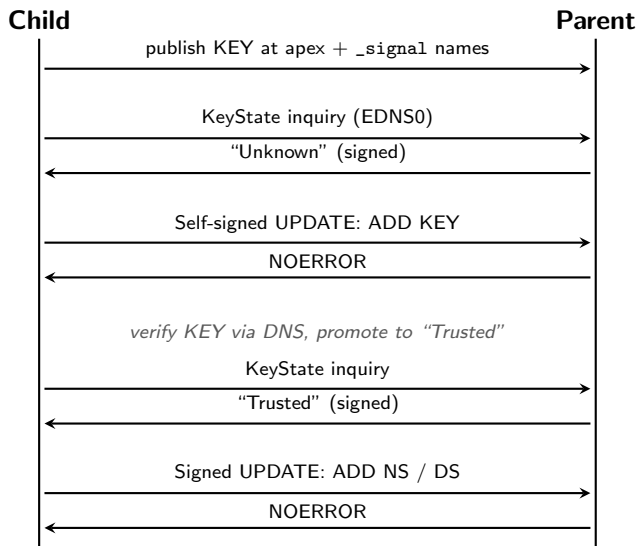
```
Extra:      Key state: Trusted
```

CLI here only
as an example

This drives the bootstrap flow:

- 1 Child publishes KEY, asks parent: KeyState inquiry
- 2 Parent says “Unknown” → child sends bootstrap UPDATE
- 3 Parent verifies KEY via DNS, promotes to “Trusted”
- 4 Child asks again → “Trusted” → ready for delegation UPDATES

The Bootstrap Sequence



Prototype Status

Open source: github.com/johanix/tdns

End-to-end bootstrap flow now works:

- Child agent publishes SIG(0) KEY to zone apex and `_signal` names.
- Child queries parent KeyState → “Unknown”
- Child sends self-signed bootstrap UPDATE.
- Parent stores KEY, verifies via DNS (`at-apex` and `at-ns`).
- Parent promotes key to “Trusted”.
- Child polls KeyState → “Trusted”

Child can now send signed delegation UPDATES.

Prototype: Recent Improvements

- Built-in validating resolver for key verification (at-apex and at-ns), eliminating external resolver dependencies.
- Configurable verification mechanisms per parent zone.
- Parent announces supported bootstrap methods via SVCB at the DSYNC target.
- Parent-side: UPDATE Receiver SIG(0) key generation, publication, and response signing.
- Child-side: automatic KeyState polling after bootstrap with exponential backoff.
- Multiple delegation output backends: direct zone update, zone file text format, and database.
- Improved CLI tooling.

Prototype: What's Next

Delegation UPDATE processing on the parent side:

- Parent receives signed UPDATE with NS/DS/glue changes.
- Verify SIG(0) signature against trusted key.

Output backends:

- Add extract/publish via API to existing backend options.

Other:

- SIG(0) key rollover via DNS UPDATE.
- Re-bootstrapping after key loss (per Section 9.2 of the draft).

Editorial Outlook

We believe the draft is close to technically complete:

- DSYNC-based service discovery (RFC 9859)
- SIG(0) bootstrap for all DNSSEC scenarios
- Mutual authentication
- UPDATE processing and policy
- Re-bootstrapping after key loss
- Scalability analysis

Editorial Outlook

We believe the draft is close to technically complete:

- DSYNC-based service discovery (RFC 9859)
- SIG(0) bootstrap for all DNSSEC scenarios
- Mutual authentication
- UPDATE processing and policy
- Re-bootstrapping after key loss
- Scalability analysis

But we acknowledge that the document needs editorial work:

- Improve readability and flow.
- Make it easier to implement from.
- Better separate normative requirements from informational discussion.
- Add more examples, especially for the bootstrap flow.

We welcome reviewers and co-editors willing to help with this.

Questions?

`draft-ietf-dnsop-delegation-mgmt-via-ddns-01`

Companion drafts:

- RFC 9859 (DSYNC record)
- `draft-berra-dnsop-announce-scanner` (SVCB at DSYNC target)
- `draft-berra-dnsop-keystate` (KeyState EDNS(0) option)

Prototype:

- github.com/johanix/tdns

Contact:

- johan.stenstam@internetstiftelsen.se