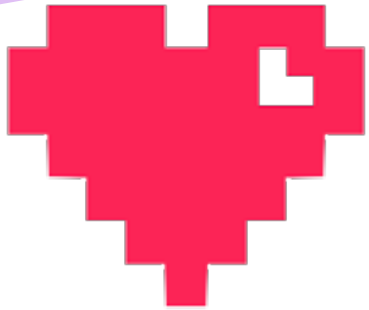
Multi-Provider DNS

Why it is important, where it is now and how to make it simpler and safer

Johan Stenstam Erik Bergström Leon Fernandez

Swedish Internet Foundation

DNS-OARC, Edinburgh — May 2026

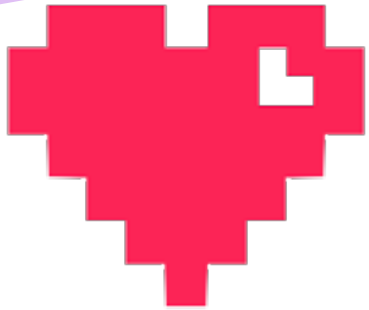


The Real Problem

INTERNET STIFTELSEN

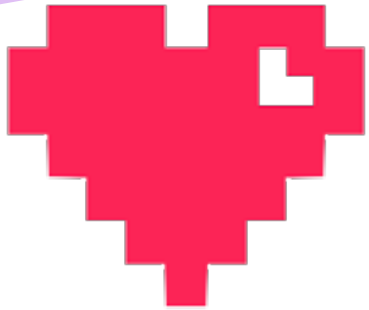
Multi-Provider Is Already Best Current Practice

Many zones — including many of the most important ones — already use multiple DNS providers. This is best current practice for resilience and availability.



INTERNET STIFTELSEN

Multi-Provider Is Already Best Current Practice



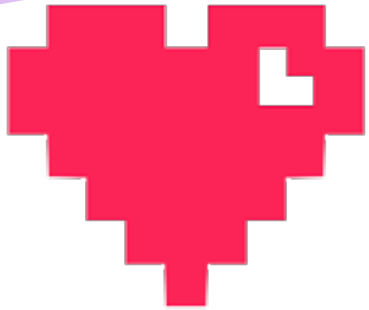
Many zones — including many of the most important ones — already use multiple DNS providers. This is best current practice for resilience and availability.

But the synchronization between providers is largely manual, fragile, or handled by proprietary integrations.

This works for NS records (barely), and breaks down completely when DNSSEC is involved.

INTERNET STIFTELSEN

Multi-Provider Is Already Best Current Practice



Many zones — including many of the most important ones — already use multiple DNS providers. This is best current practice for resilience and availability.

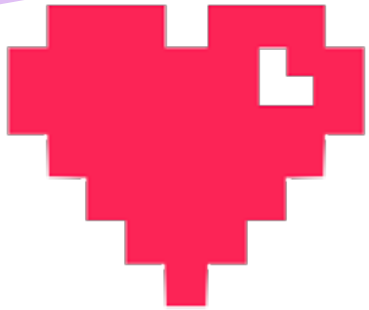
But the synchronization between providers is largely manual, fragile, or handled by proprietary integrations.

This works for NS records (barely), and breaks down completely when DNSSEC is involved.

Everyone knows that "multi-signer DNSSEC" is hard.
The perceived need is limited:
"How many zones really need multiple *signing* providers?"

INTERNET STIFTELSEN

Multi-Provider Is Already Best Current Practice



Many zones — including many of the most important ones — already use multiple DNS providers. This is best current practice for resilience and availability.

But the synchronization between providers is largely manual, fragile, or handled by proprietary integrations.

This works for NS records (barely), and breaks down completely when DNSSEC is involved.

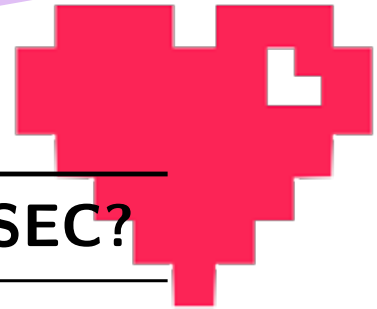
Everyone knows that "multi-signer DNSSEC" is hard.
The perceived need is limited:
"How many zones really need multiple *signing* providers?"

But this framing misses the point.

The hard part is not the signing — it is *synchronizing sensitive zone data across organizational boundaries*.

INTERNET STIFTELSEN

It's the Synchronization, Not the Signing



Consider what providers need to coordinate:

Record	**Why**	**DNSSEC?**
NS	Each provider's nameservers	No
In-bailiwick glue	Each provider's NS addresses	Yes
DNSKEY	Each provider's signing keys	Yes
CDS/CSYNC	DS/NS updates to parent	Yes

INTERNET STIFTELSEN

It's the Synchronization, Not the Signing

Consider what providers need to coordinate:

Record	**Why**	**DNSSEC?**
NS	Each provider's nameservers	No
In-bailiwick glue	Each provider's NS addresses	Yes
DNSKEY	Each provider's signing keys	Yes
CDS/CSYNC	DS/NS updates to parent	Yes

Synchronizing the NS RRset across providers is *structurally identical* to synchronizing DNSKEYs.

Same trust problem. Same timing problem.
Same need for confirmed delivery.

INTERNET STIFTELSEN

It's the Synchronization, Not the Signing

Consider what providers need to coordinate:

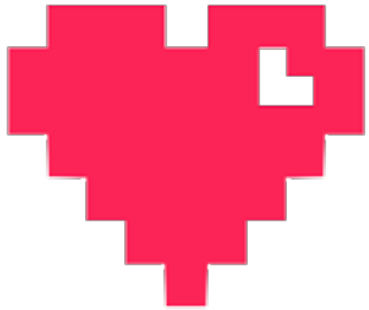
Record	Why	DNSSEC?
NS	Each provider's nameservers	No
In-bailiwick glue	Each provider's NS addresses	Yes
DNSKEY	Each provider's signing keys	Yes
CDS/CSYNC	DS/NS updates to parent	Yes

Synchronizing the NS RRset across providers is *structurally identical* to synchronizing DNSKEYs.

Same trust problem. Same timing problem.
Same need for confirmed delivery.

If we solve the synchronization problem for multi-provider, we get multi-signer almost for free.

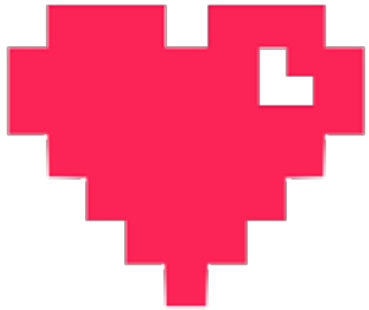
INTERNET STIFTELSEN



Why Has This Been Difficult?

INTERNET STIFTELSEN

Three Hard Sub-Problems

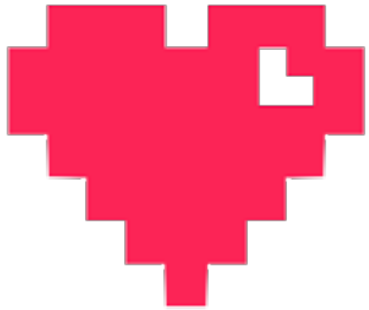


1. Trust and policy

How does a provider *know* who the other providers (if any) are and how to communicate securely with them?
Who is authorized to contribute which records?

INTERNET STIFTELSEN

Three Hard Sub-Problems



1. Trust and policy

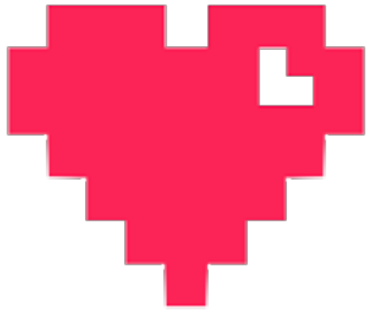
How does a provider *know* who the other providers (if any) are and how to communicate securely with them? Who is authorized to contribute which records?

2. Confirmed delivery

"I sent it and hope for the best" is not acceptable. DNSKEY rollovers require *proof* that all providers have published the new key before signing can begin.

INTERNET STIFTELSEN

Three Hard Sub-Problems



1. Trust and policy

How does a provider *know* who the other providers (if any) are and how to communicate securely with them? Who is authorized to contribute which records?

2. Confirmed delivery

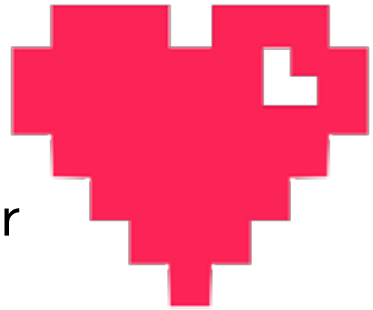
"I sent it and hope for the best" is not acceptable. DNSKEY rollovers require *proof* that all providers have published the new key before signing can begin.

3. No standard infrastructure

There is no standard way to discover peers, establish secure channels, or exchange structured messages between DNS providers.

INTERNET STIFTELSEN

Current Operational Reality

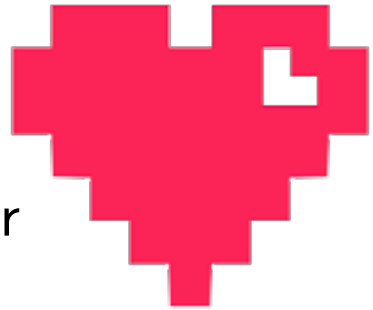


Manual

Operators coordinate via email, tickets, or ad hoc scripts. Error-prone and slow.

INTERNET STIFTELSEN

Current Operational Reality

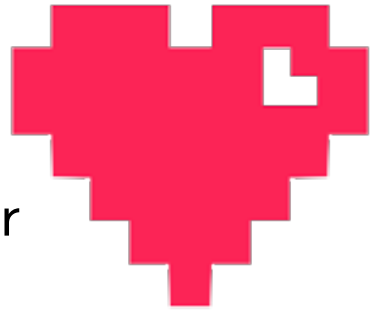


Manual Operators coordinate via email, tickets, or ad hoc scripts. Error-prone and slow.

Proprietary Some provider pairs have bilateral integrations. Does not generalize.

INTERNET STIFTELSEN

Current Operational Reality



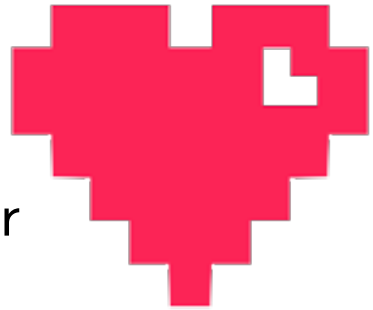
Manual Operators coordinate via email, tickets, or ad hoc scripts. Error-prone and slow.

Proprietary Some provider pairs have bilateral integrations. Does not generalize.

Avoided Most zone owners accept single-provider risk rather than deal with the complexity.

INTERNET STIFTELSEN

Current Operational Reality



Manual Operators coordinate via email, tickets, or ad hoc scripts. Error-prone and slow.

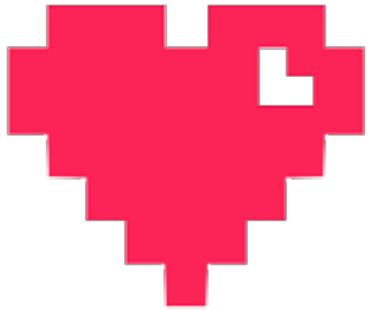
Proprietary Some provider pairs have bilateral integrations. Does not generalize.

Avoided Most zone owners accept single-provider risk rather than deal with the complexity.

There is no standard, interoperable way to automate multi-provider DNS synchronization.

This is the gap we are trying to fill.

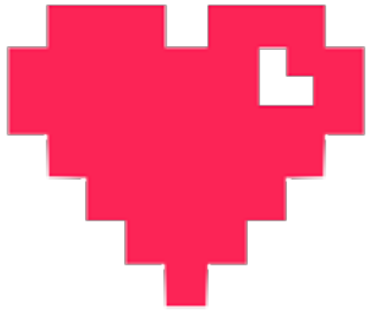
INTERNET STIFTELSEN



Architecture

INTERNET STIFTELSEN

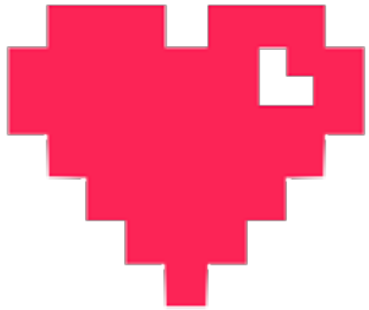
The Workflow



1. Zone owner publishes HSYNC RRset at the zone apex
   — *"I want multi-provider sync for this zone"*

INTERNET STIFTELSEN

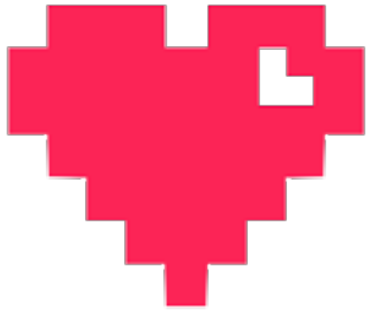
The Workflow



1. Zone owner publishes HSYNC RRset at the zone apex
   — *"I want multi-provider sync for this zone"*
2. Agents pick up the HSYNC RRset, analyze it,
   and discover their peers

INTERNET STIFTELSEN

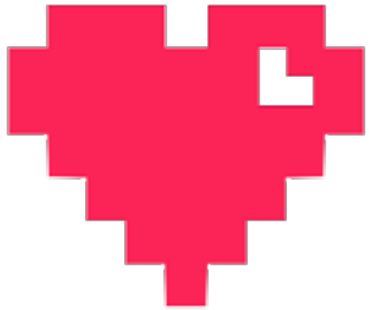
The Workflow



1. Zone owner publishes HSYNC RRset at the zone apex
   — *"I want multi-provider sync for this zone"*
2. Agents pick up the HSYNC RRset, analyze it, and discover their peers
3. Secure communications are established
   — *bootstrapped entirely from DNS*

INTERNET STIFTELSEN

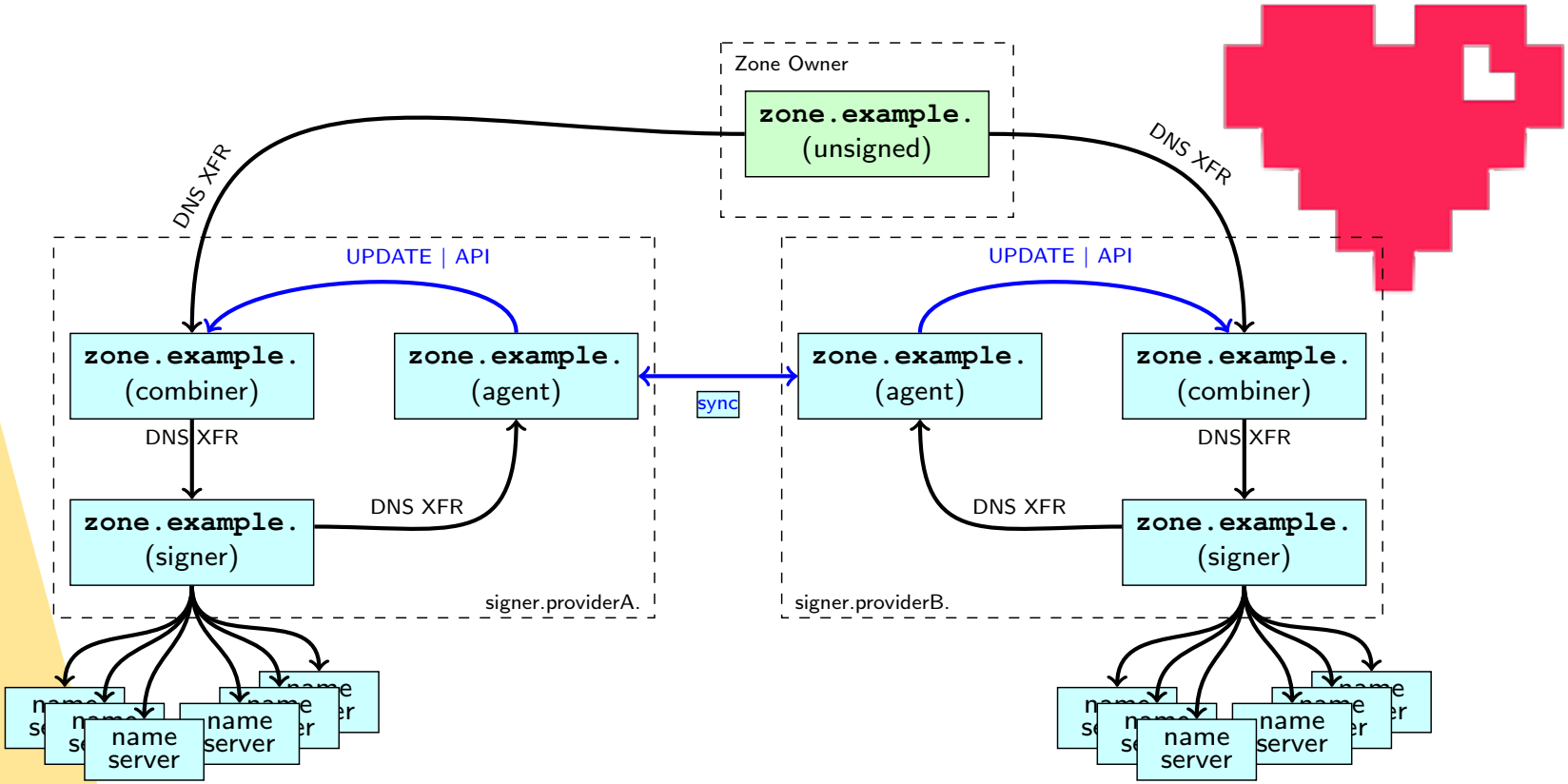
The Workflow



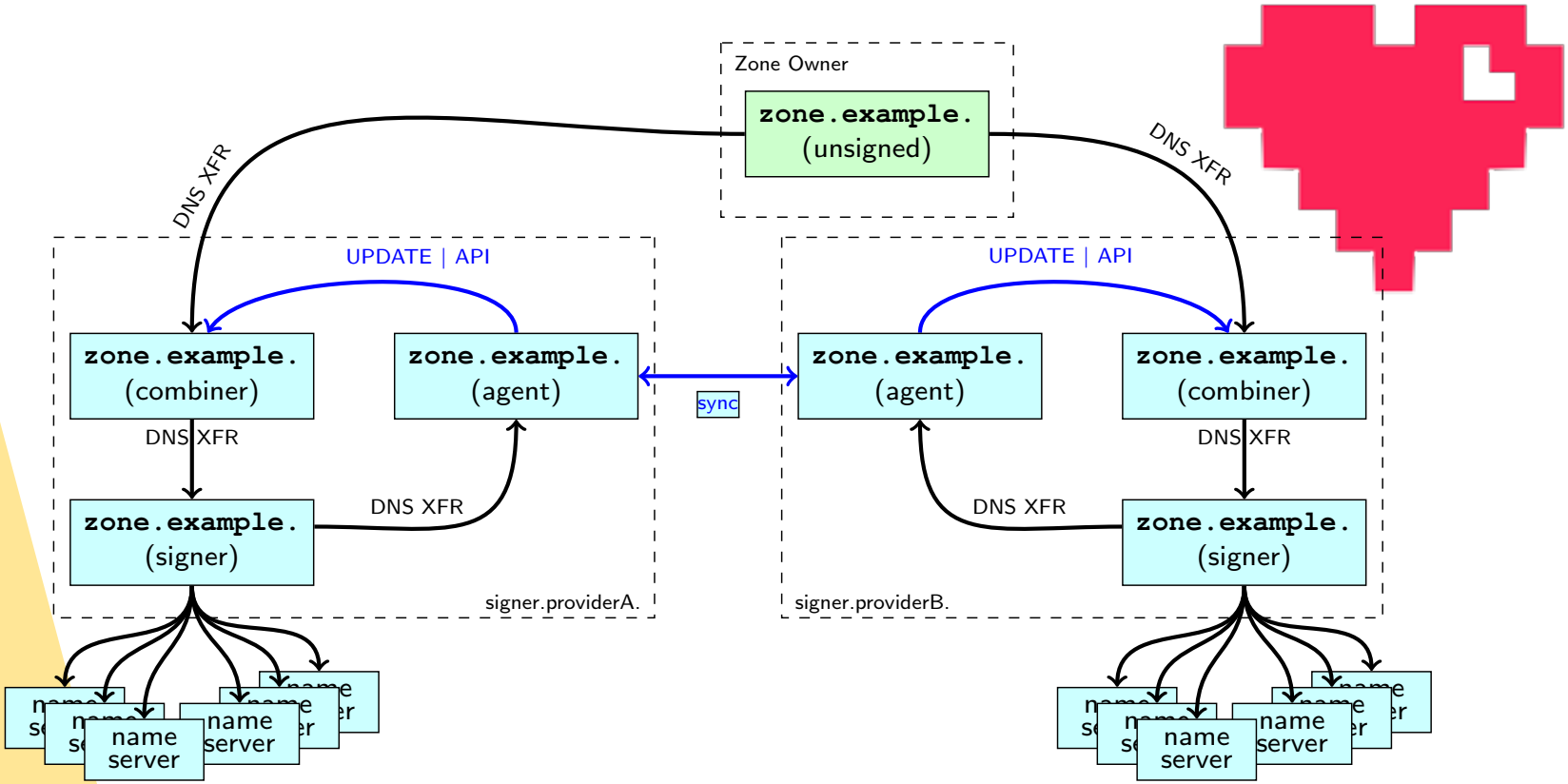
1. Zone owner publishes HSYNC RRset at the zone apex
   — *"I want multi-provider sync for this zone"*
2. Agents pick up the HSYNC RRset, analyze it, and discover their peers
3. Secure communications are established
   — *bootstrapped entirely from DNS*
4. Synchronized management of the zone begins

INTERNET STIFTELSEN

Roles


Zone Owner
zone.example.
(unsigned)
DNS XFR
DNS XFR
UPDATE | API
UPDATE | API
zone.example.
(combiner)
zone.example.
(agent)
sync
zone.example.
(agent)
zone.example.
(combiner)
DNS XFR
DNS XFR
DNS XFR
zone.example.
(signer)
zone.example.
(signer)
signer.providerA.
signer.providerB.
name server
name server
name server
name server

Roles

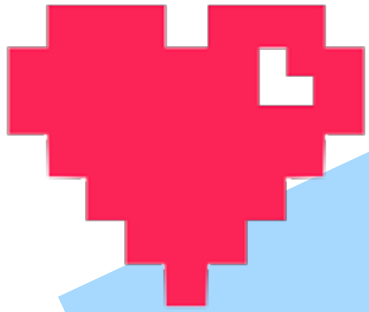


Four roles, clear separation of concerns:

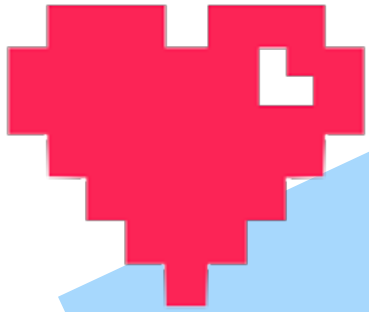
- **Zone Owner**: Authoritative source of the unsigned zone
- **Agents**: Per-provider sync (peer-to-peer, discovered via HSYNC)
- **Combiners**: Merge and validate contributions from all agents
- **Signers**: DNSSEC-sign the combined zone

INTERNET STIFTELSEN

The HSYNC and HSYNCPARAM RRsets



INTERNET STIFTELSEN

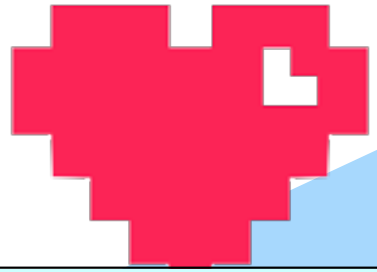


The HSYNC and HSYNCPARAM RRsets

Two records, one source of truth

INTERNET STIFTELSEN

HSYNC + HSYNCPARAM at the Zone Apex

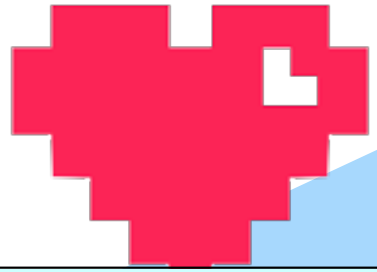


The zone owner publishes one HSYNC record per agent at the zone apex. Each record identifies one provider:

```
customer.zone. IN HSYNC ON alpha agent.alpha.example. .
customer.zone. IN HSYNC ON bravo agent.bravo.example. .
customer.zone. IN HSYNC ON echo  agent.echo.example.  alpha
```

INTERNET STIFTELSEN

HSYNC + HSYNCPARAM at the Zone Apex



The zone owner publishes one HSYNC record per agent at the zone apex. Each record identifies one provider:

```
customer.zone. IN HSYNC ON alpha agent.alpha.example. .
customer.zone. IN HSYNC ON bravo agent.bravo.example. .
customer.zone. IN HSYNC ON echo  agent.echo.example.  alpha
```

Four HSYNC RDATA fields:

- **State**: **`ON`** or **`OFF`** (onboarded vs. being removed)
- **Label**: short tag for this provider (referenced by HSYNCPARAM)
- **Identity**: the provider's FQDN (starting point for agent discovery)
- **Upstream**: label of the provider this one pulls from (**`.`** = manual setup)

The HSYNC RRset identifies the involved parties.

INTERNET STIFTELSEN

HSYNC + HSYNCPARAM at the Zone Apex

HSYNCPARAM completes the picture: a single record at the apex carrying the zone-wide multi-provider policy:

```
customer.zone. IN HSYNC ON alpha agent.alpha.example. .
customer.zone. IN HSYNC ON bravo agent.bravo.example. .
customer.zone. IN HSYNC ON echo  agent.echo.example.  alpha
customer.zone. IN HSYNCPARAM servers="alpha,bravo,echo" signers="alpha,bravo" nsmgmt="agent"
```

HSYNCPARAM keys (extensible, SVCB-style):

- **servers**: providers (labels) authorized to **serve** the zone
- **signers**: providers (labels) authorized to **sign** the zone
- **nsmgmt**: **`owner`** or **`agent`** — who manages NS records
- **parentsync**: **`owner`** or **`agent`** — who handles parent sync

HSYNC + HSYNCPARAM maps each provider to the zone policy via labels and roles.

INTERNET STIFTELSEN

From HSYNC to Secure Communication

1. Agent analyzes the HSYNC RRset for each zone it serves
   If it finds itself listed ⇒ this zone requests sync
   HSYNCPARAM provides the zone policy (signing, NS mgmt, ...)
   ↓
2. Extract remote agents' Identity fields from the HSYNC records
   ↓
3. Look up each remote agent's transport endpoint
   (URI → SVCB records for details)
   ↓
4. Look up the remote agent's public key (JWK or TLSA)
   ↓
5. Handshake: establish authenticated channel
   ↓
6. Normal sync messaging begins

INTERNET STIFTELSEN

From HSYNC to Secure Communication

1. Agent analyzes the **HSYNC RRset** for each zone it serves
   If it finds itself listed ⇒ this zone requests sync
   HSYNCPARAM provides the zone policy (signing, NS mgmt, ...)
   ↓
2. Extract remote agents' Identity fields from the HSYNC records
   ↓
3. Look up each remote agent's transport endpoint
   (URI → SVCB records for details)
   ↓
4. Look up the remote agent's public key (JWK or TLSA)
   ↓
5. Handshake: establish authenticated channel
   ↓
6. Normal sync messaging begins

Everything bootstraps from DNS itself — no out-of-band setup

INTERNET STIFTELSEN

From HSYNC to Secure Communication

1. Agent analyzes the **HSYNC RRset** for each zone it serves
   If it finds itself listed ⇒ this zone requests sync
   HSYNCPARAM provides the zone policy (signing, NS mgmt, ...)
   ↓
2. Extract remote agents' Identity fields from the HSYNC records
   ↓
3. Look up each remote agent's transport endpoint
   (URI → SVCB records for details)
   ↓
4. Look up the remote agent's public key (JWK or TLSA)
   ↓
5. Handshake: establish authenticated channel
   ↓
6. Normal sync messaging begins

Everything bootstraps from DNS itself — no out-of-band setup

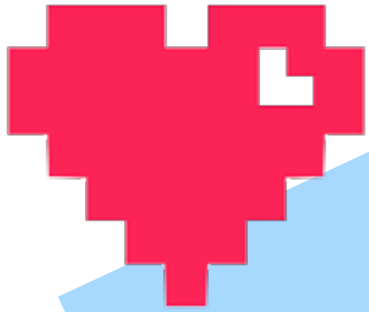
Adding or removing a provider = updating the HSYNC RRset.
Agents detect the change and act automatically.

INTERNET STIFTELSEN

Validation

Validation: The Combiner

Architectural decision: *one single point* where all remote contributions are validated before being applied.



INTERNET STIFTELSEN

Validation: The Combiner

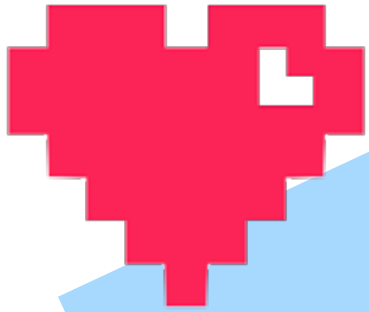
Architectural decision: *one single point* where all remote contributions are validated before being applied.

The combiner ensures that contributions are safe and correct — nothing has gone wrong, nothing is malicious:

- Standard validation rules (defined by the protocol)
- Optional manual approval (likely needed by some providers)

INTERNET STIFTELSEN

Validation: The Combiner



Architectural decision: *one single point* where all remote contributions are validated before being applied.

The combiner ensures that contributions are safe and correct — nothing has gone wrong, nothing is malicious:

- Standard validation rules (defined by the protocol)
- Optional manual approval (likely needed by some providers)

Each provider decides for themselves what validation requirements their customers require.

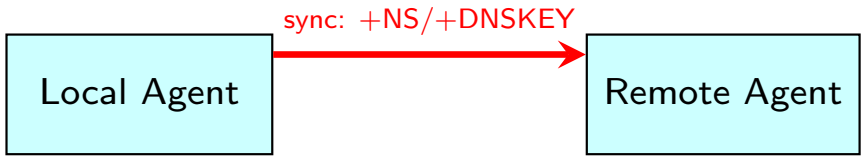
Every contribution gets a per-RR confirmation back to the originating agent: accepted or rejected, with reasons.

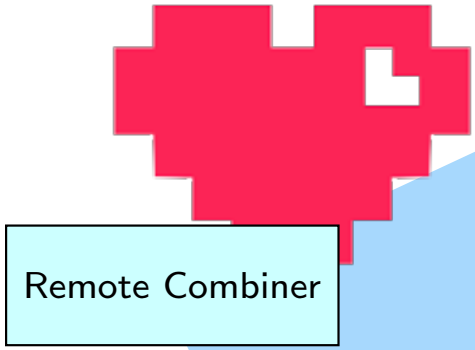
INTERNET STIFTELSEN

Confirmed Delivery

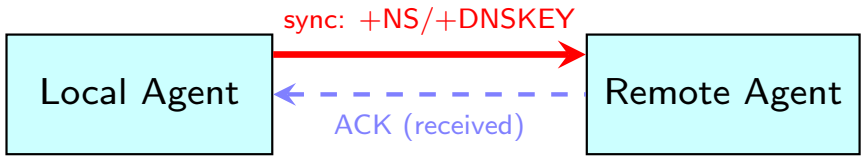
INTERNET STIFTELSEN

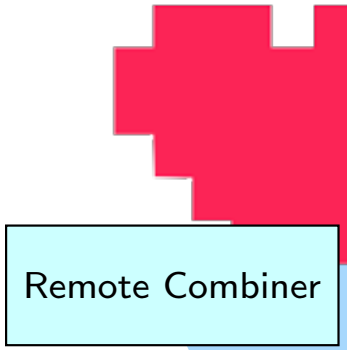
Confirmed Delivery





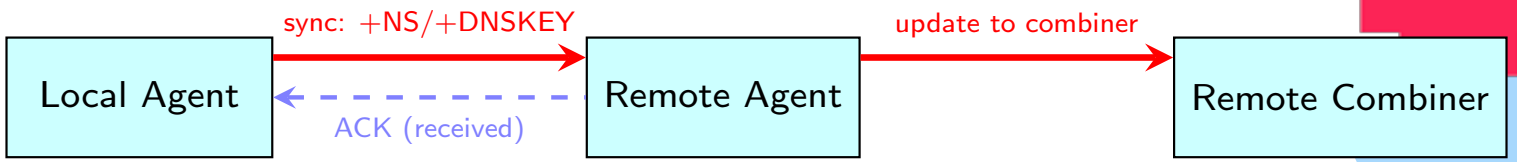
Confirmed Delivery



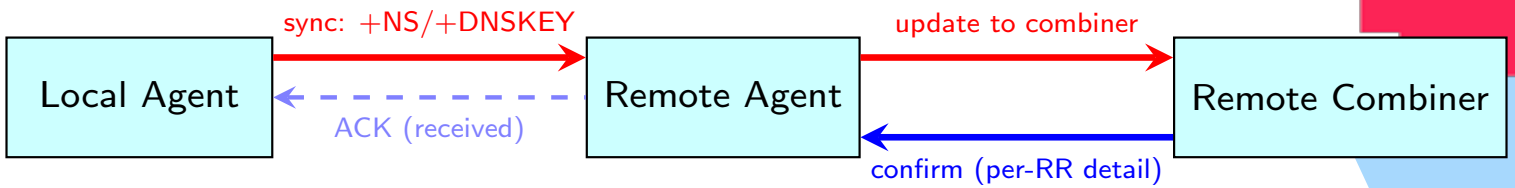


INTERNET STIFTELSEN

Confirmed Delivery



Confirmed Delivery



Confirmed Delivery

Confirmed Delivery

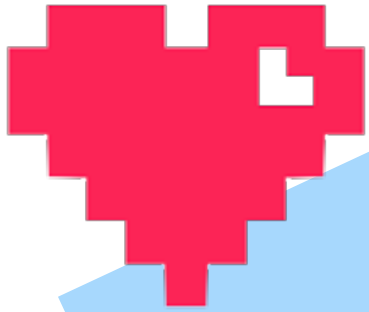
Two-step confirmation:

- **Step 1**: Immediate ACK (prevents retransmission)
- **Step 2**: Final per-RR confirmation from combiner (accepted/rejected with reasons)

The originating agent *knows* its contribution was accepted and published. This is what makes coordinated DNSKEY rollovers safe.

INTERNET STIFTELSEN

Self-Healing

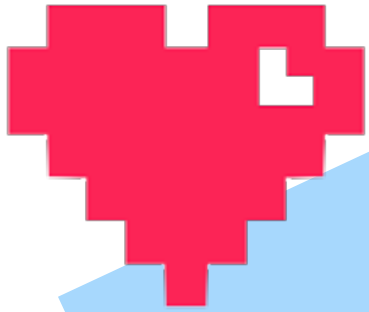


All sync operations are *idempotent*:

- **SYNC**: send my contributions to a peer.
  If the peer already has the same data: nothing changes.
  Safe to repeat after restart, reconnection, or timeout.

INTERNET STIFTELSEN

Self-Healing



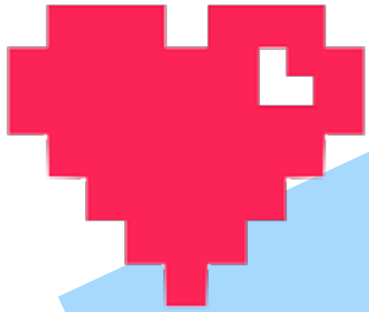
All sync operations are *idempotent*:

- **SYNC**: send my contributions to a peer.
  If the peer already has the same data: nothing changes.
  Safe to repeat after restart, reconnection, or timeout.

- **RFI** (Request For Information): "send me your current contributions." Used at startup, after failures, or when joining the group.

INTERNET STIFTELSEN

Self-Healing



All sync operations are *idempotent*:

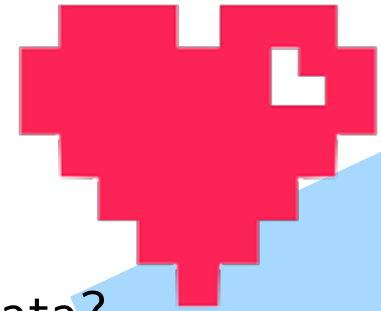
- **SYNC**: send my contributions to a peer.
  If the peer already has the same data: nothing changes.
  Safe to repeat after restart, reconnection, or timeout.

- **RFI** (Request For Information): "send me your current contributions." Used at startup, after failures, or when joining the group.

- Any agent can recover full state from its peers.
  No special recovery logic. This is also how a *new provider bootstraps*: send RFI, get up to date.

INTERNET STIFTELSEN

The Hard Part: Synchronization



A challenge has been the development of a model for distributed synchronization of DNS data.

- Which providers should be able to contribute what data?
- Where should the gates be? At the sender or at the receivers?
- How should the synchronization be designed to allow arbitrary restarts and also recovery from crashes?
- How should responsibility for parent synchronization be agreed upon (assuming it is wanted)?
- How does a multi-provider setup affect the DNSSEC key rollover state-machine?
- Etc.

There are still details to iron out. But there is a working prototype, and all the issues do seem to have good answers.

INTERNET STIFTELSEN

Prototype Status

Prototype Status

What works, what doesn't, what is next

INTERNET STIFTELSEN

What Works Today

The prototype ("TDNS") implements the full architecture:

Capability	**Status**
Agent discovery via HSYNC	Working
Secure messaging (encrypted + signed)	Working
NS synchronization across providers	Working
DNSKEY synchronization (lifecycle)	Working
Combiner policy + namespace protection	Working
Two-step confirmed delivery	Working
Reliable delivery (retry until confirmed)	Working
Idempotent sync + RFI	Working
Agent restart & bootstrap from peers	Working
Combiner persistence (survives restart)	Working
CLI management tools	Working

INTERNET STIFTELSEN

What Doesn't Work Yet

Gap	Description
CDS/CSYNC notifications to parent	Prototype working, but not yet fully integrated.
Automated key rollover	DNSKEY sync works across multi-provider; full KSK rollover is still only single-provider
Transport	DNS-based transport complete; REST API transport incomplete
Provider addition/removal	Orderly works; disorderly needs more testing
Automatic zone transfer setup	Not yet implemented

INTERNET STIFTELSEN

What Doesn't Work Yet

Gap	Description
CDS/CSYNC notifications to parent	Prototype working, but not yet fully integrated.
Automated key rollover	DNSKEY sync works across multi-provider; full KSK rollover is still only single-provider
Transport	DNS-based transport complete; REST API transport incomplete
Provider addition/removal	Orderly works; disorderly needs more testing
Automatic zone transfer setup	Not yet implemented

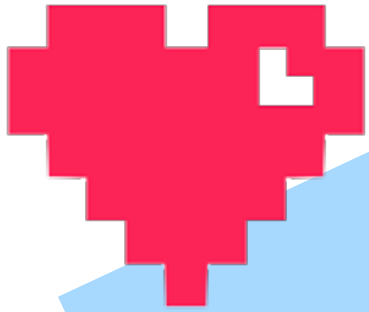
These are engineering tasks, not research problems.
The hard architectural questions have been answered.

INTERNET STIFTELSEN

What Is Next

Near-term:

- Automated DNSKEY rollover (end-to-end)
- Wrap-up parent synchronization
- Automatic setup of zone transfers
- Stress testing and fault injection



INTERNET STIFTELSEN

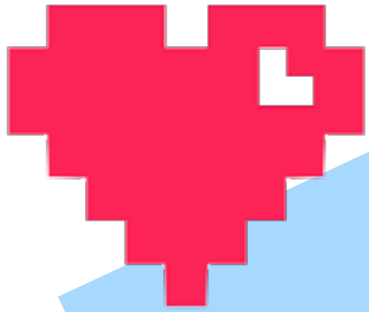
What Is Next

Near-term:

- Automated DNSKEY rollover (end-to-end)
- Wrap-up parent synchronization
- Automatic setup of zone transfers
- Stress testing and fault injection

Medium-term:

- Interoperability testing with real providers
- API transport alongside DNS transport
- Operational tooling (monitoring, auditing)



INTERNET STIFTELSEN

What Is Next

Near-term:

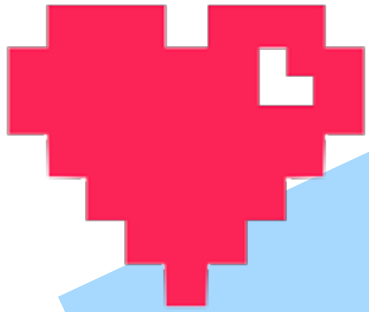
- Automated DNSKEY rollover (end-to-end)
- Wrap-up parent synchronization
- Automatic setup of zone transfers
- Stress testing and fault injection

Medium-term:

- Interoperability testing with real providers
- API transport alongside DNS transport
- Operational tooling (monitoring, auditing)

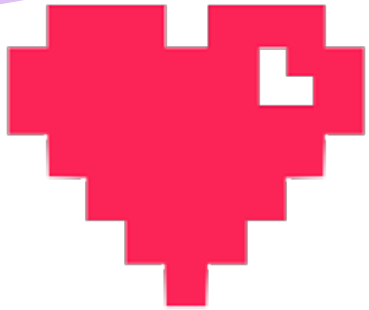
Longer-term:

- Standardization of the sync protocol (IETF)
- Making multi-provider DNS *easier* to deploy than single-provider DNS is today



INTERNET STIFTELSEN

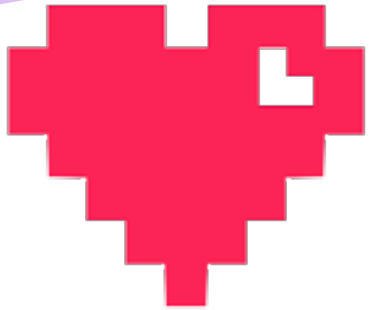
Summary



- **Multi-provider is harder than people think** — even without DNSSEC, synchronizing NS records across providers is a real problem

INTERNET STIFTELSEN

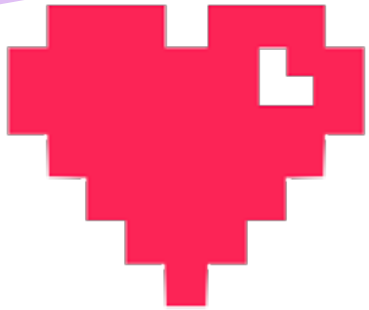
Summary



- **Multi-provider is harder than people think** — even without DNSSEC, synchronizing NS records across providers is a real problem
- **The hard part is synchronization**, not signing — solve sync and you get multi-signer almost for free

INTERNET STIFTELSEN

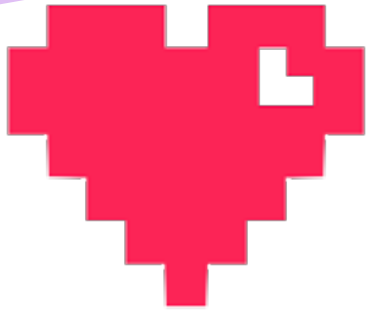
Summary



- **Multi-provider is harder than people think** — even without DNSSEC, synchronizing NS records across providers is a real problem
- **The hard part is synchronization**, not signing — solve sync and you get multi-signer almost for free
- **HSYNC RRset** as the foundation: opt-in, discovery, and authorization from DNS itself

INTERNET STIFTELSEN

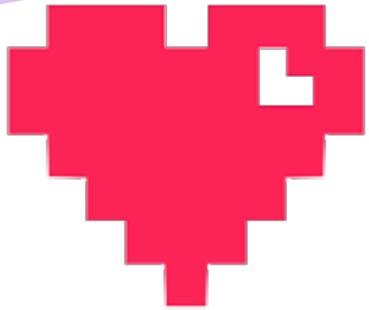
Summary



- **Multi-provider is harder than people think** — even without DNSSEC, synchronizing NS records across providers is a real problem
- **The hard part is synchronization**, not signing — solve sync and you get multi-signer almost for free
- **HSYNC RRset** as the foundation: opt-in, discovery, and authorization from DNS itself
- **Architecture**: agents, combiners, signers with policy enforcement and confirmed delivery

INTERNET STIFTELSEN

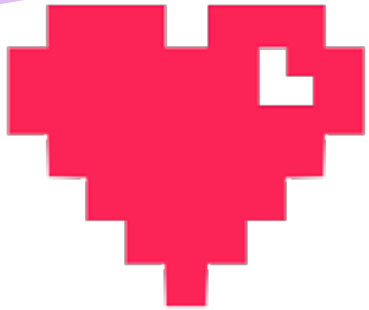
Summary



- **Multi-provider is harder than people think** — even without DNSSEC, synchronizing NS records across providers is a real problem
- **The hard part is synchronization**, not signing — solve sync and you get multi-signer almost for free
- **HSYNC RRset** as the foundation: opt-in, discovery, and authorization from DNS itself
- **Architecture**: agents, combiners, signers with policy enforcement and confirmed delivery
- **Working prototype**: full pipeline from zone owner through sync to signed zone on name servers

INTERNET STIFTELSEN

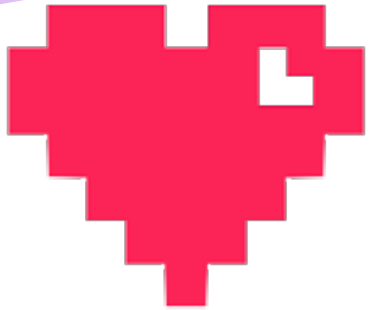
Summary



- **Multi-provider is harder than people think** — even without DNSSEC, synchronizing NS records across providers is a real problem
- **The hard part is synchronization**, not signing — solve sync and you get multi-signer almost for free
- **HSYNC RRset** as the foundation: opt-in, discovery, and authorization from DNS itself
- **Architecture**: agents, combiners, signers with policy enforcement and confirmed delivery
- **Working prototype**: full pipeline from zone owner through sync to signed zone on name servers

Goal: make multi-provider DNS *operationally feasible* for everyone, not just the largest operators.

INTERNET STIFTELSEN



Questions?

Contact `johan.stenstam@internetstiftelsen.se`

INTERNET STIFTELSEN