Making Music With DNSSEC

Getting the pesky details of a complex DNSSEC corner right through automation

Johan Stenstam Roger Murray

Internetstiftelsen

February 28, 2022

To many people DNSSEC sounds a bit like...

To many people DNSSEC sounds a bit like...

...nails scratching against a black board

To many people DNSSEC sounds a bit like...

...nails scratching against a black board

- Unfortunately, it isn't hard to understand why.
- There are lots of details, lots of things to get wrong.
- In particular there are things that may go wrong **over time** even if one successfully turns on DNSSEC for a zone.

DNSSEC Progress and Failures over the Years

Compared to the initial versions, we've come a long way

- Key generation and zone signing is mostly automated.
- Software quality and robustness has improved dramatically.
- There are multiple quality providers of both primary and secondary services for DNSSEC signed zones.
- Knowledge of what DNSSEC does and why is now widespread .

DNSSEC Progress and Failures over the Years

Compared to the initial versions, we've come a long way

- Key generation and zone signing is mostly automated.
- Software quality and robustness has improved dramatically.
- There are multiple quality providers of both primary and secondary services for DNSSEC signed zones.
- Knowledge of what DNSSEC does and why is now widespread .

But there are also things that still aren't quite sorted

- Failure scenarios are still painful, both to experience and to debug.
- Some zone owners avoid rolling their keys, in particular their KSKs, as that involves communicating with the parent zone.
- Changing DNS operator for a signed zone is so utterly difficult that the normal case is to "go unsigned" during transition.

DNSSEC Progress and Failures over the Years

Let's focus here

Compared to the initial versions, we've come a long way

- Key generation and zone signing is mostly automated.
- Software quality and robustness has improved dramatically.
- There are multiple quality providers of both primary and secondary services for DNSSEC signed zones.
- Knowledge of what DNSSEC does and why is now widespread .

But there are also things that still aren't quite sorted

- Failure scenarios are still painful, both to experience and to debug.
- Some zone owners avoid rolling their keys, in particular their KSKs, as that involves communicating with the parent zone.
- Changing DNS operator for a signed zone is so utterly difficult that the normal case is to "go unsigned" during transition.

Focusing on the Issue of Communicating With the Parent

In the DNSSEC case the parent communication is primarily updating the **DS** (Delegation Signer) record in the parent zone.

- To be fair, it is not just an issue with communication it is just as much an issue about the perceived risks of making a change to critical DNSSEC stuff that may break.

However, this is not the only case when it is necessary to communicate with the parent.

- There is also the old classic of keeping the **NS** records and **A** and **AAAA** glue records in sync.

We all know the importance of this, but for some reason it seems that there is always a certain percentage of delegations that are not in sync and therefore, while they may still work, are not as robust as they think they are.

Communicating With the Parent, cont'd

There are several reasons for this communication failure but among them a clear contender for primary cause is the lack of communication path between the DNS operator and the registry (i.e. the parent).

- The registrar has that communication channel (usually via EPP) and this has led to a prevalence of the registrar also being the DNS operator, to the detriment of third party DNS operators.

What is really needed is a mechanism for the DNS operator to **signal to the parent** that certain changes to the data in the parent zone are needed. The good news is that there now exists such a mechanism

- It is implemented via special signalling records in the child zone, called **CDS** and **CSYNC**.

Enter the DNS Record Types **CDS** and **CSYNC**

To update the **DS** records in the parent zone it should be sufficient that:

- The child zone publishes a **CDS** RRset ("Child DS") containing the data it wants the parent to publish as a **DS** RRset.

Likewise, to update the **NS** RRset and/or the **A** and **AAAA** glue records in the parent it should be sufficient that:

- The child zone publishes a **CSYNC** record that signals to the parent what data in the child zone to sync into the parent zone (out of relevant **NS**, **A** and **AAAA** RRsets)

In both cases a requirement is that the child zone is DNSSEC signed. This is needed to be able to verify the data that is copied from the child zone (like the **CDS** RRset).

CDS and **CSYNC**, cont'd

An important caveat is the part about "should be sufficient". **CDS** and **CSYNC** publication only works (as in cause updates in the parent) for parent zones that have chosen to "scan" the child zones to look for these records.

- At this time most parent zones do not scan for **CDS** and **CSYNC** but the expectation is that this will change quite rapidly.

The almost magic part of **CDS** and **CSYNC** is that it suddenly makes synchronization of delegations between parent and child automatic and seamless. And this is enabled by DNSSEC.

- Suddenly DNSSEC is not so much a cause for nail scratching as many people still think.
- Rather, that DNSSEC makes the delegation synchronisation problem that DNS has had for decades just. . . go away is great.

It's almost like. . . MUSIC.

CDS and **CSYNC**, cont'd

An important caveat is the part about "should be sufficient". **CDS** and **CSYNC** publication only works (as in cause updates in the parent) for parent zones that have chosen to "scan" the child zones to look for these records.

- At this time most parent zones do not scan for **CDS** and **CSYNC** but the expectation is that this will change quite rapidly.

The almost magic part of **CDS** and **CSYNC** is that it suddenly makes synchronization of delegations between parent and child automatic and seamless. And this is enabled by DNSSEC.

- Suddenly DNSSEC is not so much a cause for nail scratching as many people still think.
- Rather, that DNSSEC makes the delegation synchronisation problem that DNS has had for decades just. . . go away is great.

It's almost like. . . M^{U}S$_{I}$C.

Going back to: DNSSEC Progress and Failures

Compared to the initial versions of DNSSEC, we've come a long way

- Key generation and zone signing is mostly automated.
- Software quality and robustness has improved dramatically.
- There are multiple quality providers of both primary and secondary services for DNSSEC signed zones.
- Knowledge of what DNSSEC does and why is now widespread.

But there are also things that still aren't quite sorted

- Failure scenarios are still painful, both to experience and to debug.
- Some zone owners avoid rolling their keys, in particular their KSKs, as that involves the parent zone.
- Changing DNS operator for a signed zone is so utterly difficult that the normal case is to "go unsigned" during transition.

Going back to: DNSSEC Progress and Failures

Now let's focus here

Compared to the initial versions of DNSSEC, we've come a long way

- Key generation and zone signing is mostly automated.
- Software quality and robustness has improved dramatically.
- There are multiple quality providers of both primary and secondary services for DNSSEC signed zones.
- Knowledge of what DNSSEC does and why is now widespread.

But there are also things that still aren't quite sorted

- Failure scenarios are still painful, both to experience and to debug.
- Some zone owners avoid rolling their keys, in particular their KSKs, as that involves the parent zone.
- Changing DNS operator for a signed zone is so utterly difficult that the normal case is to "go unsigned" during transition.

Focusing On the Issue of Changing the Signer

Why is it so difficult?

Well, it's because:

- It involves keys and in particular exchanging keys between two parties (gaining and losing operator) that have different incentives with the operation.
- Keys are often (for good reason) stored in a way that intentionally makes it difficult to do anything with them other than the "normal operation" (i.e. signing stuff).
- The margins in the DNS services sector are so thin that operators only make money when they don't spend any staff hours on doing something special for certain zones.

Enter the Multi Signer Controller, MUSIC

MUSIC is a piece of software that implements the "multi-signer" Internet-Draft by Shumon Huque and Ulrich Wisser.

- That document describes the processes to migrate a signed zone from having one set of "signers" to a new set where either a signer has been added or removed.
- A "signer" is a service that receives the unsigned zone and produces a signed version. A signer is able to generate and manage the necessary keys itself.
- The most common case is that of migrating from one DNS operator to another
  - In multi-signer terms that equals going from one signer via a temporary state of having two signers and then back to a single signer (which is the new one, and the original signer has been removed).

Exactly What Is It That MUSIC Does?

MUSIC does three things:

- MUSIC will get the **DNSKEY** RRsets for the zone in sync between multiple signers by adding the **DNSKEY**s for the ZSKs from each signer to the other signers (the **DNSKEY**s for the KSKs are not needed).

Exactly What Is It That MUSIC Does?

MUSIC does three things:

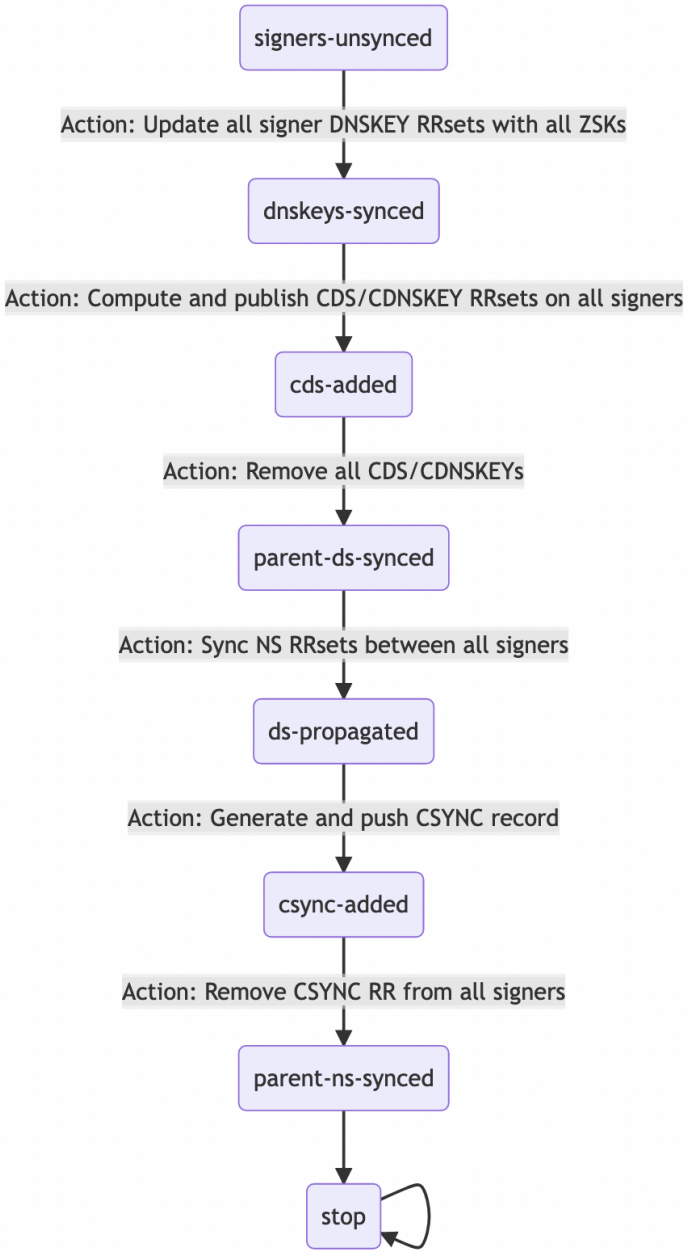
- MUSIC will get the **DNSKEY** RRsets for the zone in sync between multiple signers by adding the **DNSKEY**s for the ZSKs from each signer to the other signers (the **DNSKEY**s for the KSKs are not needed).
- MUSIC will get the **DS** records in the parent zone in sync. It does that by adding instructions to the parent via **CDS** records that are inserted into the zones that the signers maintain.

Exactly What Is It That MUSIC Does?

MUSIC does three things:

- MUSIC will get the **DNSKEY** RRsets for the zone in sync between multiple signers by adding the **DNSKEY**s for the ZSKs from each signer to the other signers (the **DNSKEY**s for the KSKs are not needed).
- MUSIC will get the **DS** records in the parent zone in sync. It does that by adding instructions to the parent via **CDS** records that are inserted into the zones that the signers maintain.
- MUSIC will get the delegation data in the parent zone (**NS** records and also **A** and **AAAA** glue records) in sync. It does that by adding instructions to the parent via **CSYNC** records that are inserted into the zones that the signers maintain.

What Does the MUSIC "Processes" Do?

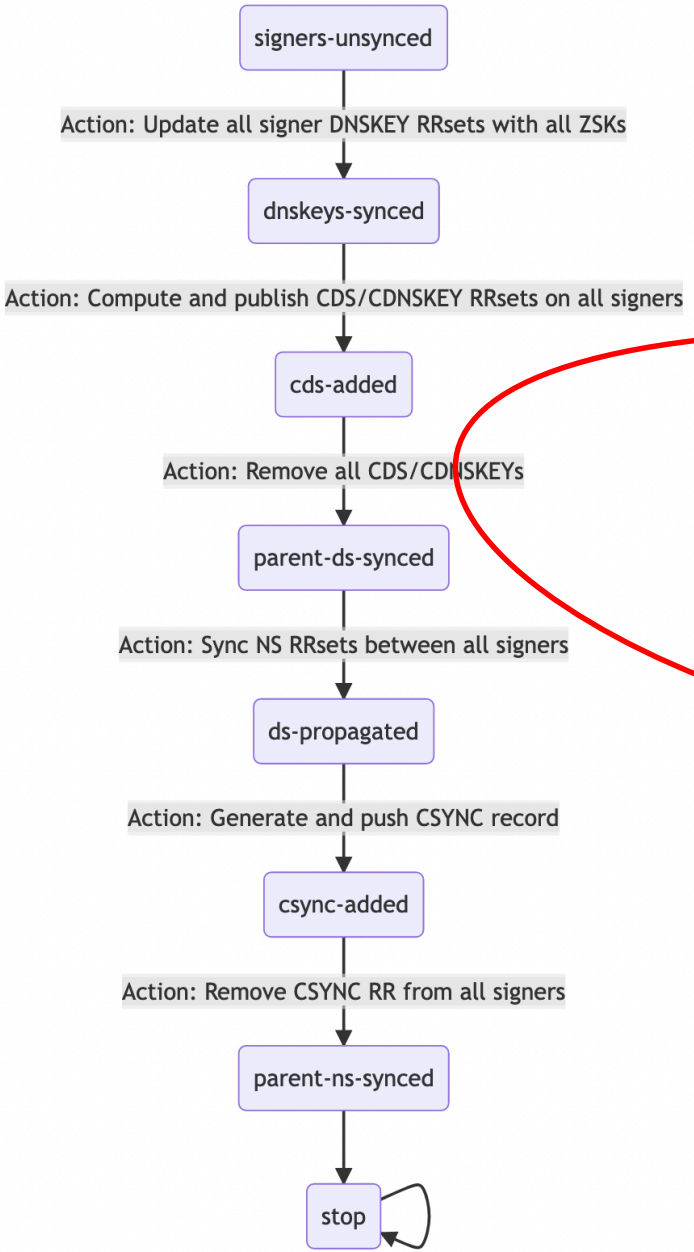


The ADD-SIGNER process

- has a number of "states", all of which are valid
- has a number of "transitions", each with a pre-condition and a post-condition
- starts with getting the signers in sync
- continues with publishing `CDS` records to trigger a `DS` update in the parent
- concludes with syncing remaining delegation data via `CSYNC`

REMOVE-SIGNER is similar, but obviously does things mostly in the opposite order.

What Does the MUSIC "Processes" Do?



This is a crucial point

The ADD-SIGNER process

- has a number of "states", all of which are valid
- has a number of "transitions", each with a pre-condition and a post-condition
- starts with getting the signers in sync
- continues with publishing `CDS` records to trigger a `DS` update in the parent
- concludes with syncing remaining delegation data via `CSYNC`

REMOVE-SIGNER is similar, but obviously does things mostly in the opposite order.

How Does **MUSIC** Work?

MUSIC knows three types of entities:

- **signers** are services that are able to sign and publish zones.
- **zones** are normal DNS zones. MUSIC doesn't care how the zone contents are maintained, nor how zone content is is kept in sync with the signers.
- **signergroups** are data structures that connects signers and zones. All zones in a signergroup will be kept in sync across all signers in the group.

Adding a signer to a signergroup automatically cause the zones in the signergroup to go through the "ADD-SIGNER" process to ensure that that the zones get in sync given the new set of signers will cause changes to **DNSKEY**s, and quite likely also to **NS** records and perhaps glue.

- Removing a signer likewise trigger the "REMOVE-SIGNER" process for the zones in the signergroup.

How Does **MUSIC** Work, cont'd

The core requirement to being able to design a "third party" controller (like **MUSIC**) for the synchronization between signers is the availability of an interface that allows the controller to update DNS records directly in the signers.

- The poster child update interface is standard DNS dynamic updates, which work with several open-source nameserver implementations including BIND9 and PowerDNS.
- Commercial DNS services typically provide update access via some sort of API. Here, we use the deSEC API as the proof-of-concept. But it will likely work well with most other APIs, like Google, Route53, etc, although at this time this has not yet been tested.

What **MUSIC** Is

- **MUSIC** is a proof-of-concept implementation. The goal is to prove that it is possible to fully automate the complex processes of adding or removing signers for a signed zone **without "going unsigned"**.
- **MUSIC** uses a server—client design where the client uses a secure RESTful API to communicate with the server.
- The **MUSIC server** manages all zones through the series of state transitions that are needed according to the multi signer algorithms.
- The **MUSIC client** part can be anything. We provide a command-line tool, but it could just as well be a web frontend, or part of a provisioning system managing a large portfolio of zones.
- **MUSIC** is **safe**. Every state that a zone passes through during the process is a valid state where everything is working fine. Every transition has pre- and post-conditions that must be fulfilled for that step to take place.

What Are the Use Cases for MUSIC?

The goal was to to address the problem of how to change the DNS operator for a DNSSEC signed zone without "going unsigned".
However, it seems that there are several similar scenarios that differ more in organisational structure than in the operations needed:

- Migrate a zone from a publication pipeline dependent on one HSM to a new pipeline dependent on a new HSM.
- For an important zone that has a (usually in-house) DNSSEC pipeline it is a possibility to be able to have two independent pipelines for redundancy reasons. MUSIC would do this automatically.
- Similar to the previous case, a zone may have outsourced zone signing to the DNS operator, but would like to be able to have multiple independent DNS operators, including the signing part. Without MUSIC or similar software this is close to impossible to achieve today.

Current State of MUSIC And Next Steps?

The current system works fine, but is limited to only work with signers that support DDNS as an interface.

That said, there are some improvements that we would like to fix over the coming weeks.

- The interface to the API for the deSEC DNS service is almost done and will be complete soon.
- Today the only user interface is the CLI tool, **`music-cli`**. It would be nice with a web interface to easier present current state.
- For testing purposes we built a simple CDS/CSYNC-scanner to deploy in the parent of test zones. This is not really part of MUSIC but we still want to clean that up a bit.
- Although we don't know of any, there are of course some lingering bugs somehere that we need to expose and dispose of.

Code: `https://github.com/DNSSEC-Provisioning/music.git`

Contact: **`musicians@internetstiftelsen.se`**