



# DNSSEC Key Ceremonies

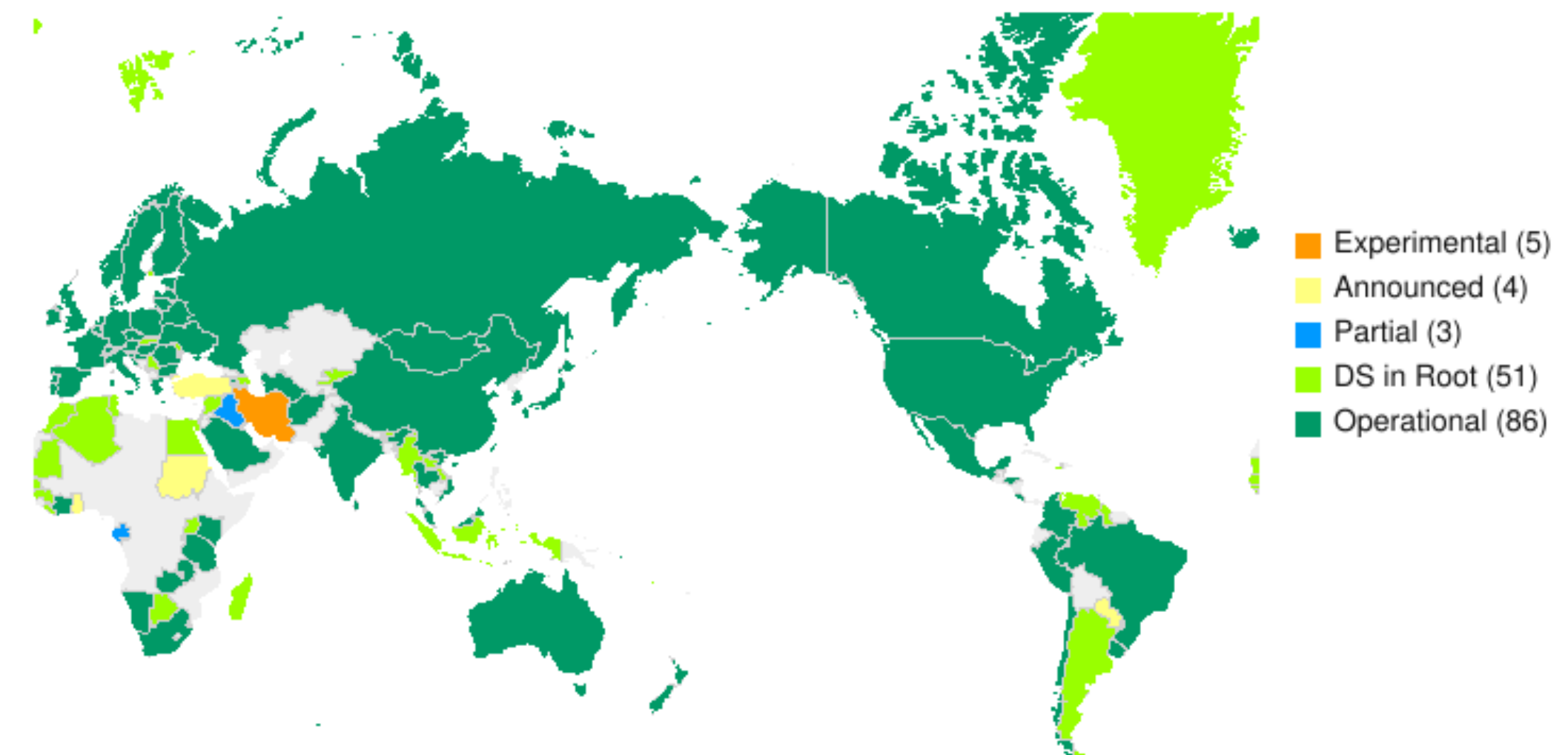
standardising and automating key security

Benno Overeinder

# Project rationale

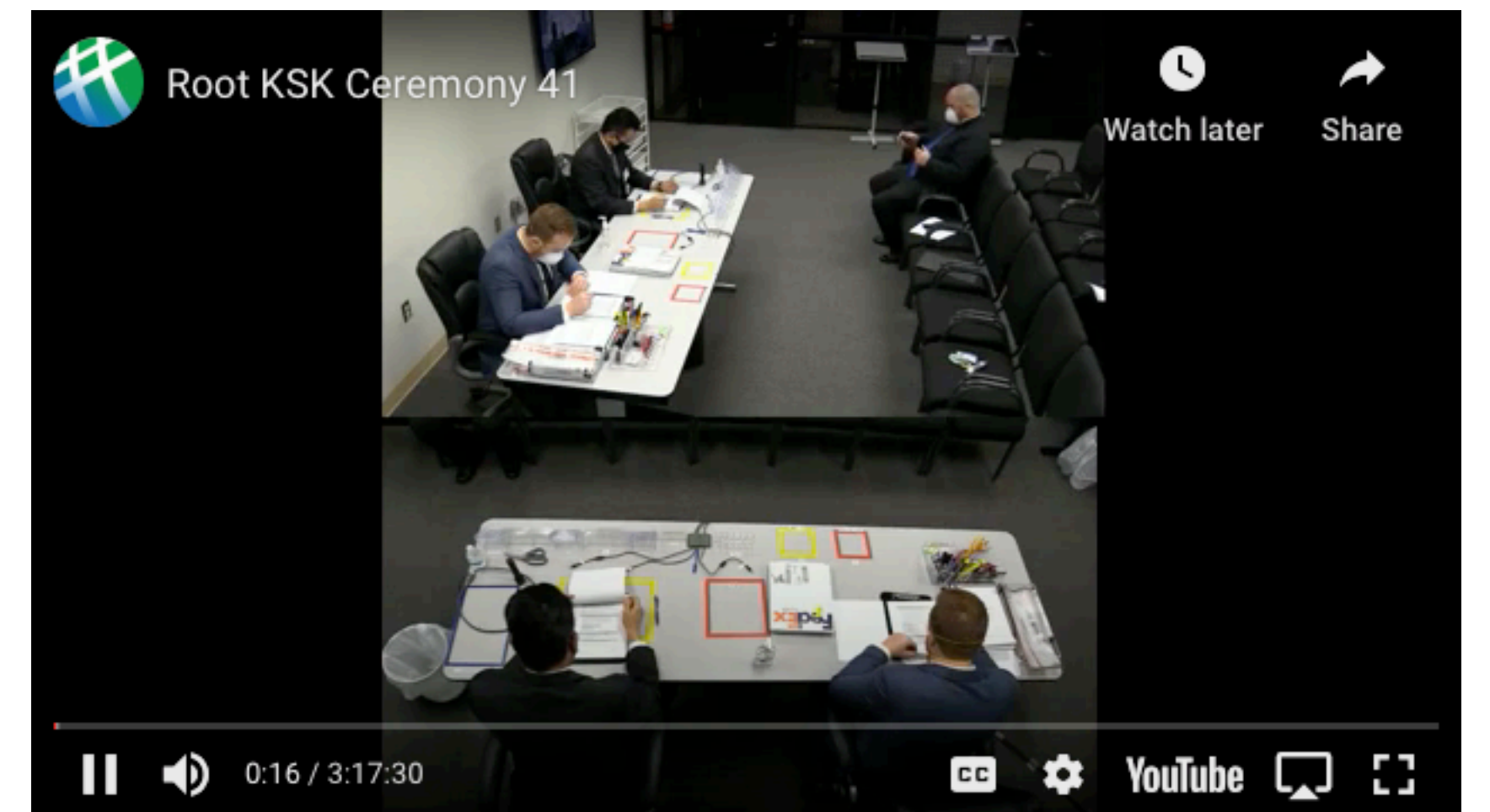
- **DNSSEC** has seen **widespread adoption** over the past decade
- Almost **all top-level domains** are now signed
- High-value domains (such as TLDs) **need strong key protection**
- **Often use HSMs** to protect key material

ccTLD DNSSEC Status on 2020-09-14



# Ceremonies

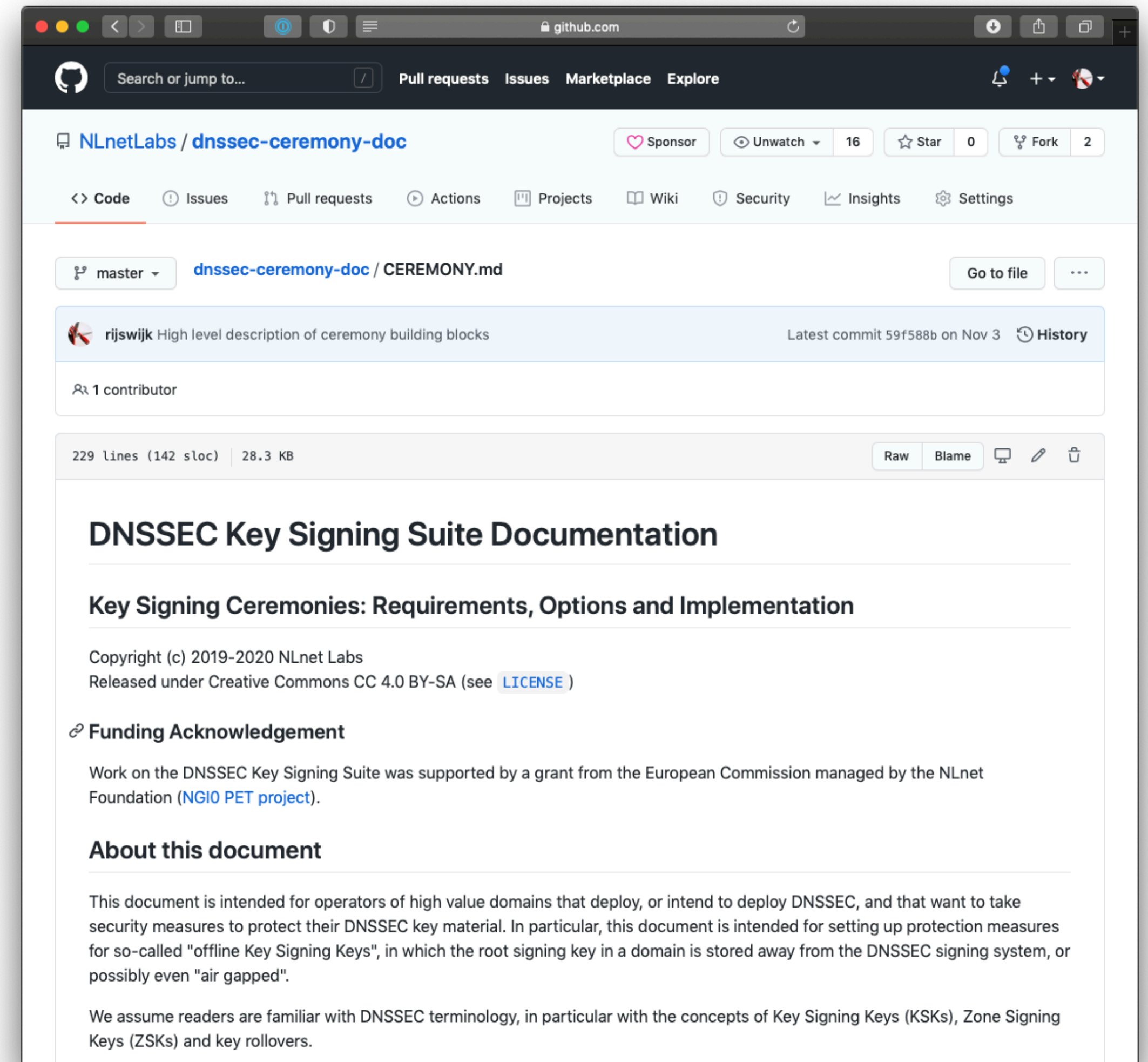
- **HSMs** for high-value domains are often **air-gapped for additional security**
- **Signing with** keys in **air-gapped HSMs** **requires** a process or **ceremony**
- **E.g.** done **for the DNS Root** (organised by IANA)
- **Ceremonies** are **sometimes witnessed** by community representatives or stakeholders



source: <https://www.iana.org/dnssec/ceremonies>

# Ceremony requirements & design

- Until now, these **key ceremonies** are **often bespoke** in terms of **process and tooling**
- Making **standardised guidelines** can **help the community**; we documented this
- **Ensure secure ceremonies** and help **automate** the process



# Automation with recipes

- **Better standardisation** enables **better automation**
- We introduce the concept of a **"recipe"**; **coherent set of instructions for key ceremony automation in the secure environment** with the air-gapped HSM
- **Built the tools to execute recipes and prototype integration with OpenDNSSEC signer**



An example of a recipe with the most common action types is given below:

```
{
  "recipeSpecVersion": "v1.0",
  "recipe": {
    "preamble": { ... },
    "actions": [
      { "actionType": "haveKey", "actionParams": { ... }, "cooked": { ... } },
      { "actionType": "haveKey", "actionParams": { ... }, "cooked": { ... } },
      { "actionType": "deleteKey", "actionParams": { ... }, "cooked": { ... } },
      { "actionType": "generateKey", "actionParams": { ... }, "cooked": { ... } },
      { "actionType": "produceSignedKeyset", "actionParams": { ... }, "cooked": { ... } },
      { "actionType": "produceSignedKeyset", "actionParams": { ... }, "cooked": { ... } },
    ]
  }
}
```

The parameters for each of the action types are specified further down.

### Specifying keys

The recipe specification is designed to be flexible in how keys are specified, in particular keys that are part of a keyset that needs to be signed in a `produceSignedKeyset` action. Two models are supported, a model in which all keys, including ZSKs, are generated in the "bunker", and a model in which some keys are generated outside of the "bunker", but need to be included in the signed keyset. This means that there are also two ways to specify keys in action parameter sets: by reference, or direct. The two ways to specify keys are shown in detail below:

#### Key by reference:

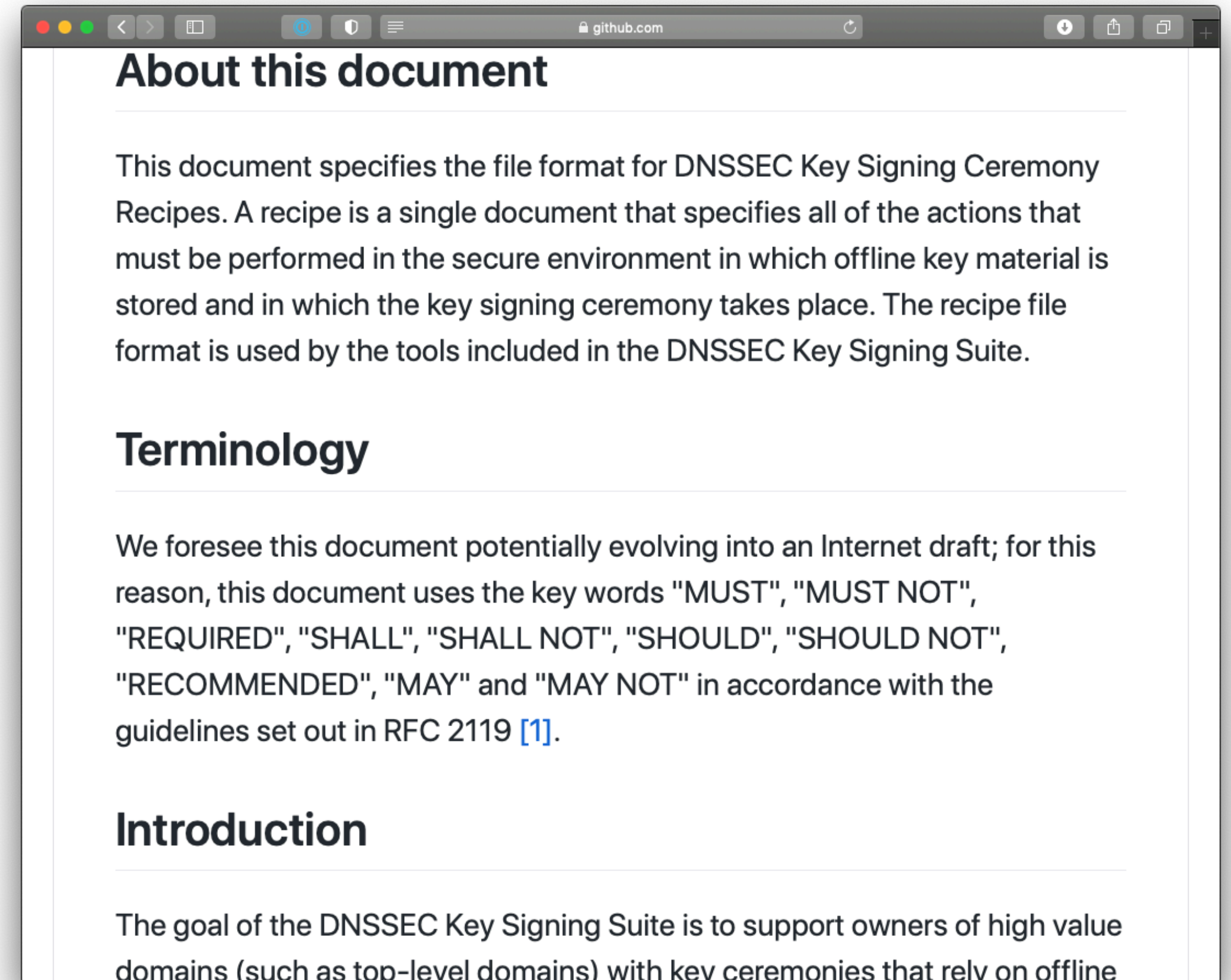
This way of referring to keys may be used in `haveKey`, `deleteKey`, `generateKey` and `produceSignedKeyset` actions.

Keys are specified as follows:

```
{
  "keyType": "byRef",
  "keyAlgo": "INTEGER",
  "keyLength": "INTEGER"
```

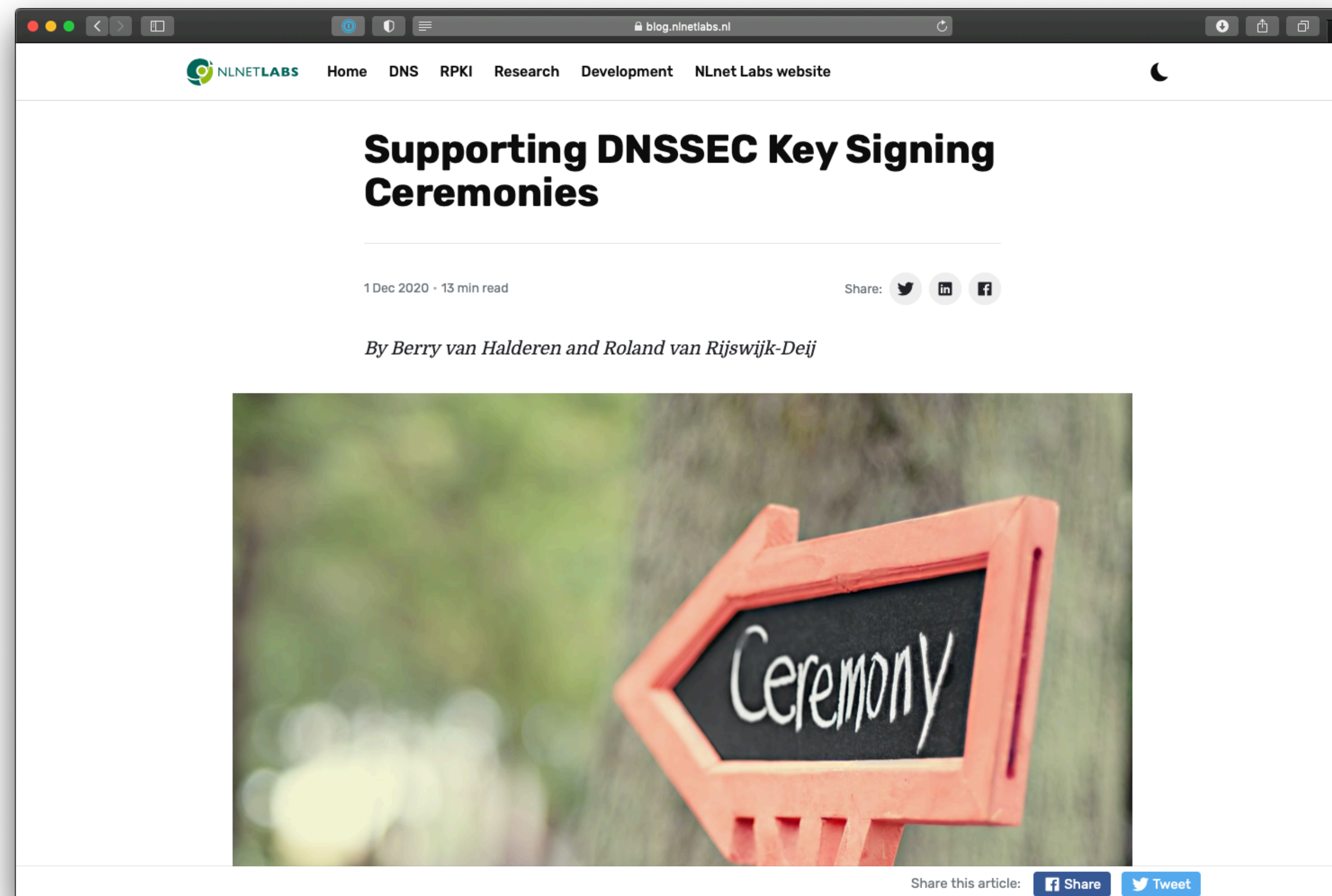
# Future work

- **Get community feedback!**
- If an interest exists, **take recipe API to IETF** for standardisation
- **Adoption by other OSS DNSSEC implementers**



# Further reading

- We wrote a blog about the project:  
<https://blog.nlnetlabs.nl/supporting-dnssec-key-signing-ceremonies/>



# Thank you! Questions?

<https://nlnetlabs.nl/>

 @nlnetlabs

[labs@nlnetlabs.nl](mailto:labs@nlnetlabs.nl)