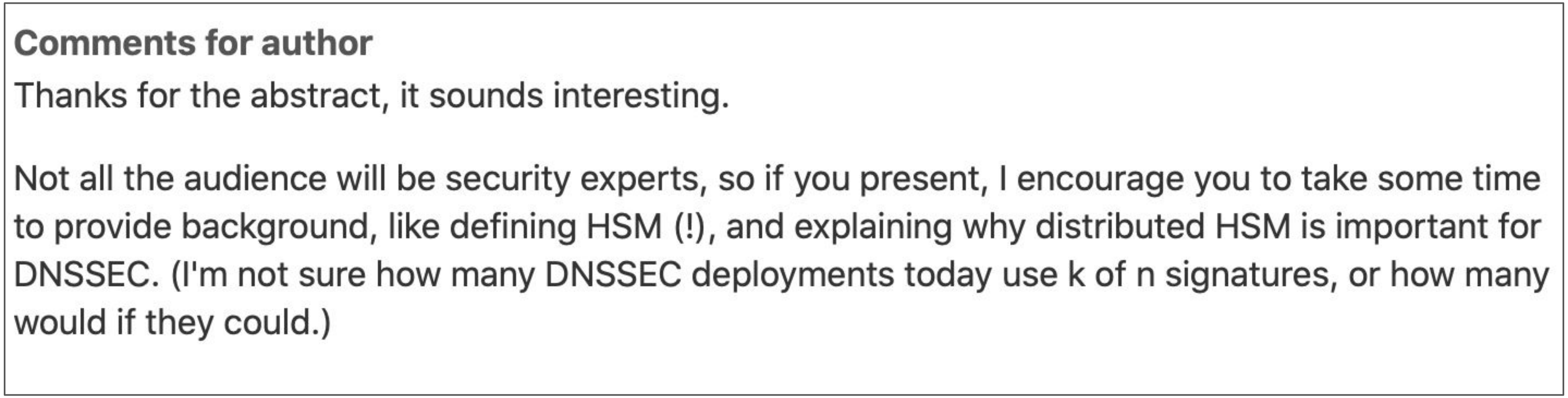
Presenting our Distributed HSM and DNS-Tools

Javier Bustos-Jiménez & Hugo Salgado
jbustos@niclabs.cl, hsalgado@nic.cl

DNS and Internet Naming Research Directions (DINR) 2020



Comments for author

Thanks for the abstract, it sounds interesting.

Not all the audience will be security experts, so if you present, I encourage you to take some time to provide background, like defining HSM (!), and explaining why distributed HSM is important for DNSSEC. (I'm not sure how many DNSSEC deployments today use k of n signatures, or how many would if they could.)

What is an HSM?

A **hardware security module** (**HSM**) is a physical computing device that safeguards and manages **digital keys**, performs encryption and decryption functions for **digital signatures**, strong authentication and other cryptographic functions.

These modules traditionally come in the form of a plug-in card or an external device that attaches directly to a computer or network server.

A hardware security module contains one or more secure cryptoprocessor chips.

Source:**https://en.wikipedia.org/wiki/Hardware_security_modul e**

What is an HSM?

A **hardware security module** (**HSM**) is a physical computing device that safeguards and manages **digital keys**, performs encryption and decryption functions for **digital signatures**, strong authentication and other cryptographic functions.

These modules traditionally come in the form of a plug-in card or an external device that attaches directly to a computer or network server.

A hardware security module contains one or more secure cryptoprocessor chips.

Source:**https://en.wikipedia.org/wiki/Hardware_security_module**

What is an HSM?

A **hardware security module** (**HSM**) is a physical computing device that safeguards and manages **digital keys**, performs encryption and decryption functions for **digital signatures**, strong authentication and other cryptographic functions.

These modules traditionally come in the form of a plug-in card or an external device that attaches directly to a computer or network server.

A hardware security module contains one or more secure cryptoprocessor chips.

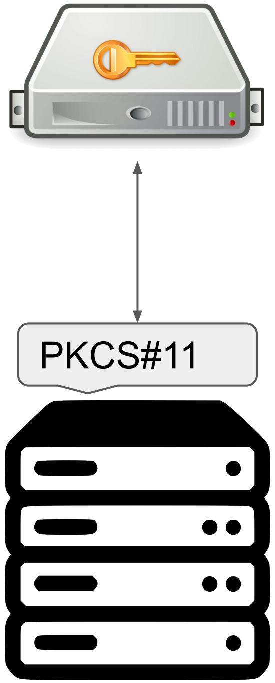
Source:**https://en.wikipedia.org/wiki/Hardware_security_modul e**

HSM Communication: PKCS#11

The **PKCS #11** standard defines a platform-independent API to cryptographic tokens, such as hardware security modules (HSM) and smart cards, "PKCS #11" is often used to refer to the API as well as the standard that defines it.

The API defines most commonly used cryptographic object types (**RSA keys**, X.509 Certificates, DES/Triple DES keys, etc.) and all the functions needed to use, create/generate, modify and delete those objects.

Source: **https://en.wikipedia.org/wiki/PKCS_11**

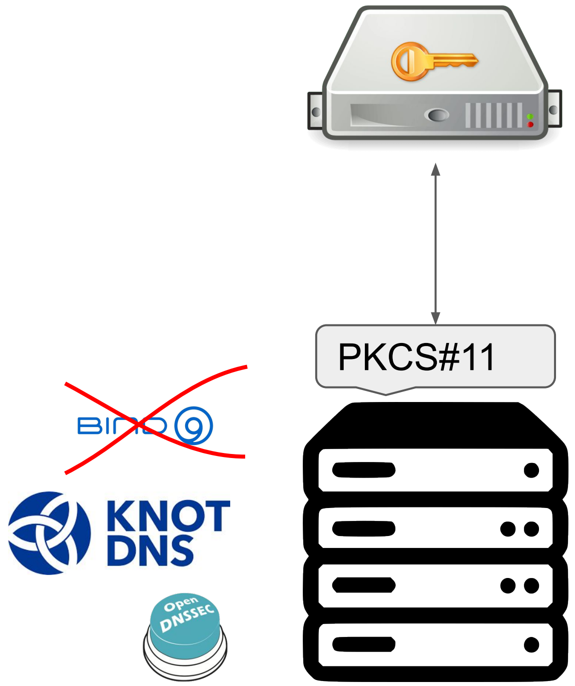


HSM Communication: PKCS#11

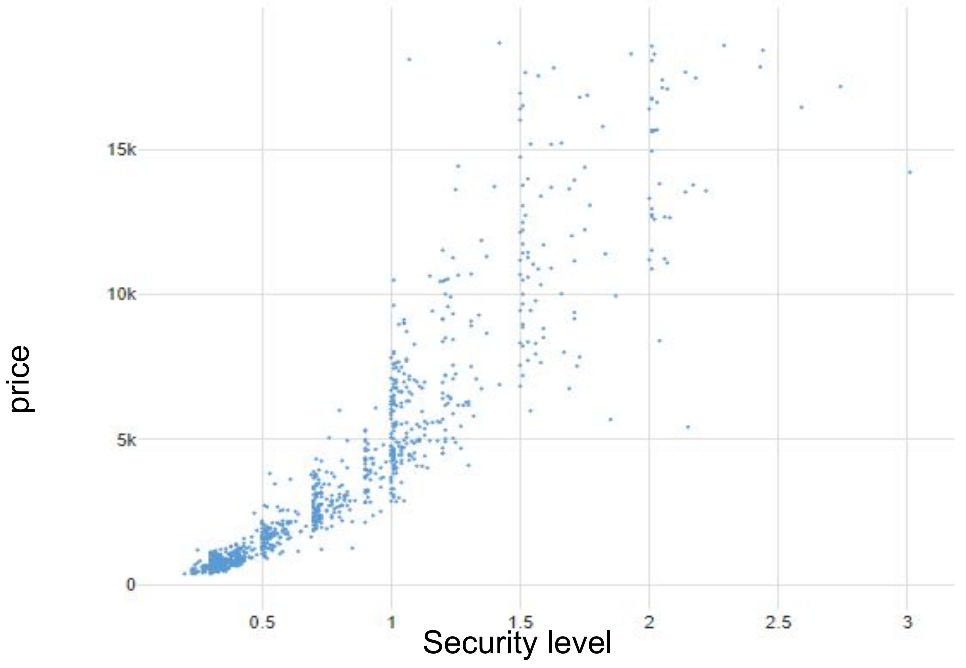
The **PKCS #11** standard defines a platform-independent API to cryptographic tokens, such as hardware security modules (HSM) and smart cards, "PKCS #11" is often used to refer to the API as well as the standard that defines it.

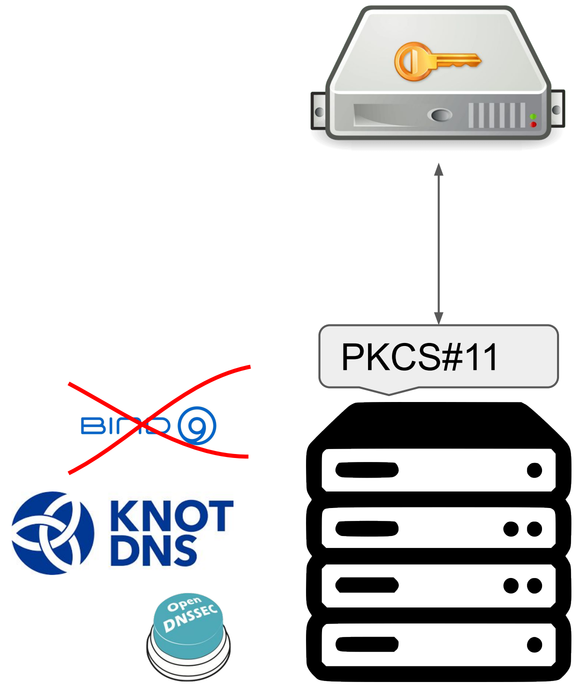
The API defines most commonly used cryptographic object types (**RSA keys**, X.509 Certificates, DES/Triple DES keys, etc.) and all the functions needed to use, create/generate, modify and delete those objects.

Source: **https://en.wikipedia.org/wiki/PKCS_11**



But, how expensive is an HSM?



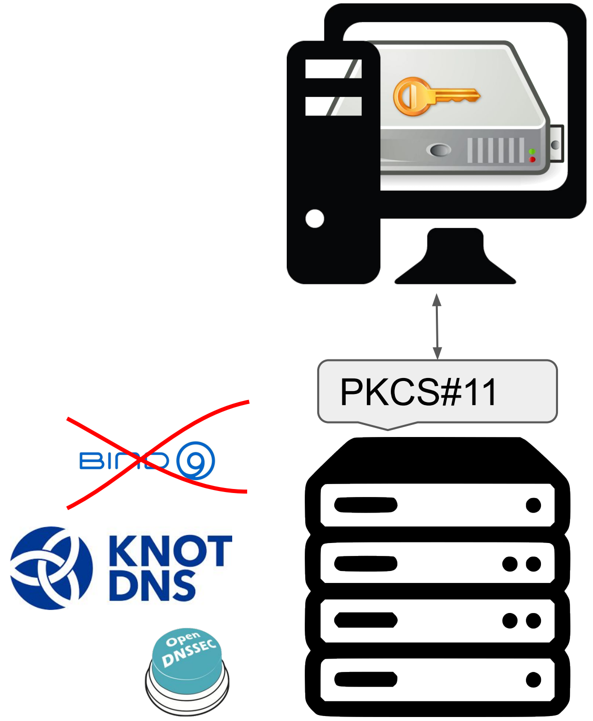


Solution: SoftHSM

A “Hardware” Security Module, but in Software.

Implements PKCS#11.

Stores data in encrypted databases.



Solution: SoftHSM

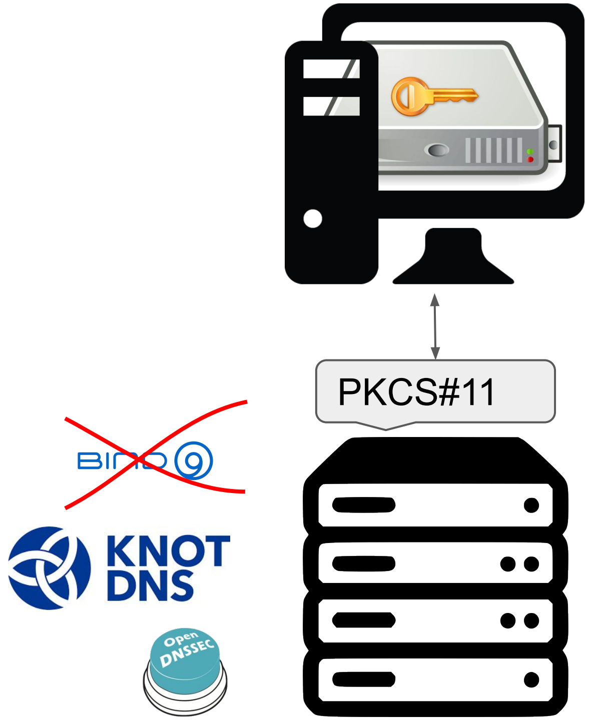
A “Hardware” Security Module, but in Software.

Implements PKCS#11.

Stores data in encrypted databases.

But:

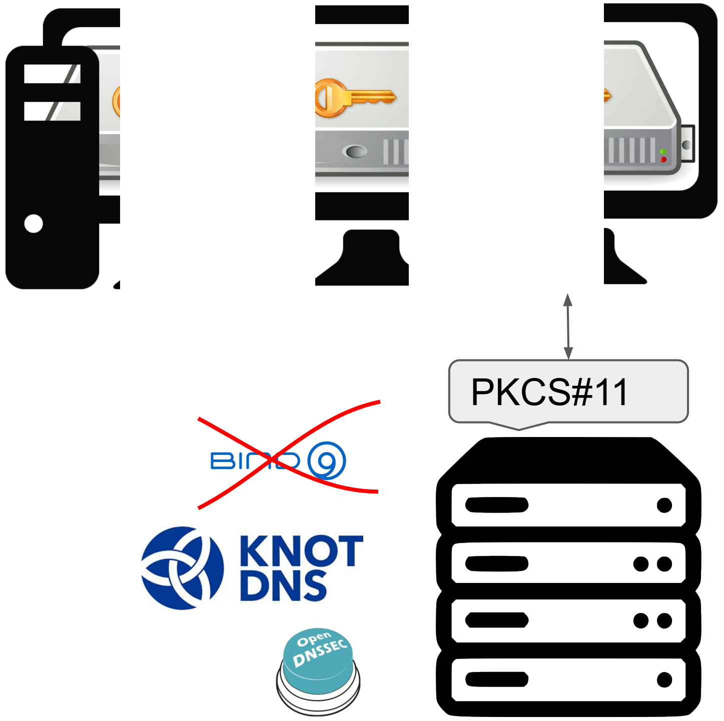
- Single point of failure (as an HSM)
- No security guarantees (hackable)
- Stores SoftHSM encryption keys at same machine.



What if?

We could cut the key in pieces?

… and distributed the pieces?



But:

- Single point of failure (as an HSM)
- No security guarantees (hackable)
- Stores SoftHSM encryption keys at same machine.

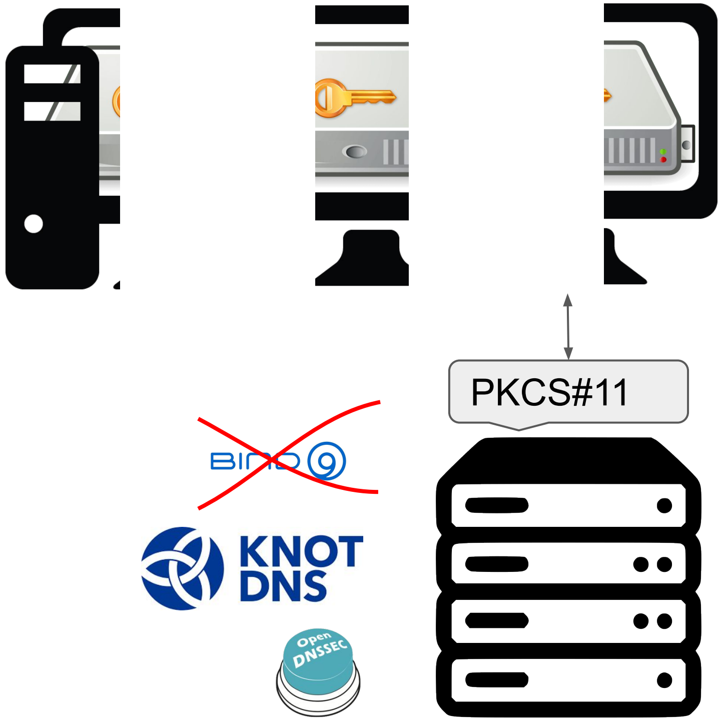
What if?

We could cut the key in pieces?

… and distributed the pieces?

~~**But**~~:

- ~~Single point of failure (as an HSM)~~
- ~~No security guarantees (hackable)~~
- ~~Stores SoftHSM encryption keys at same machine~~.



What if?

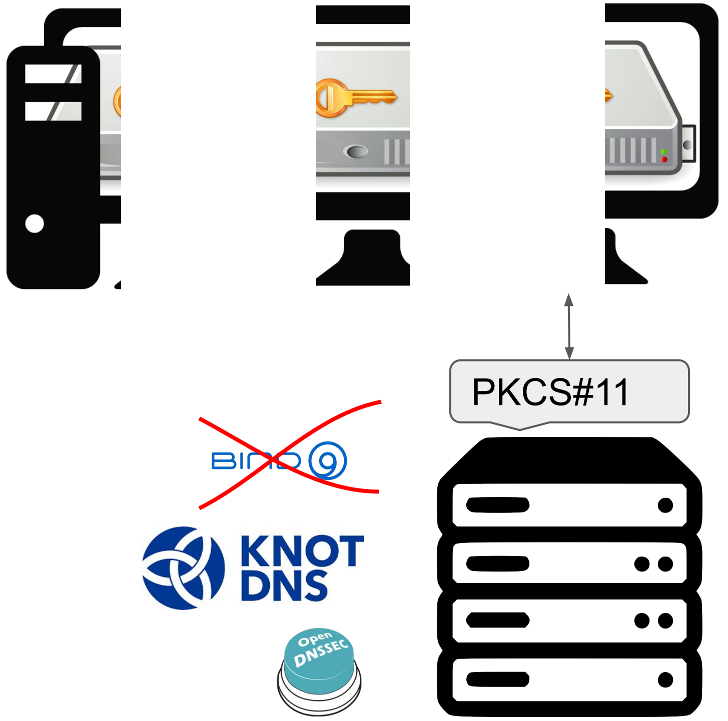
We could cut the key in pieces?

… and distributed the pieces?

… and make a valid signature only if **k > n/2** pieces are valid? (**achieving fault tolerance**)

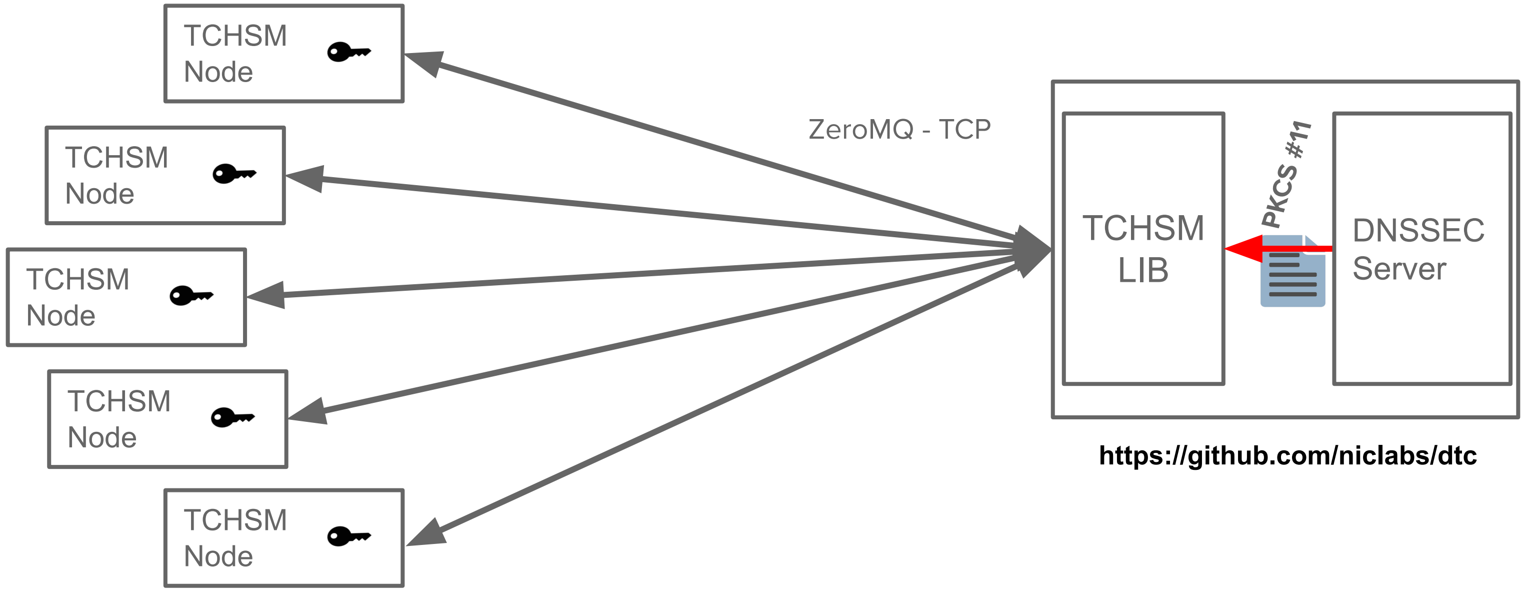
E.g.: 3 signature pieces, needing only 2 to produce a valid signature

… and using commodity hardware?



Our solution: Threshold-Cryptography Distributed HSM

https://niclabs.cl/tchsm/

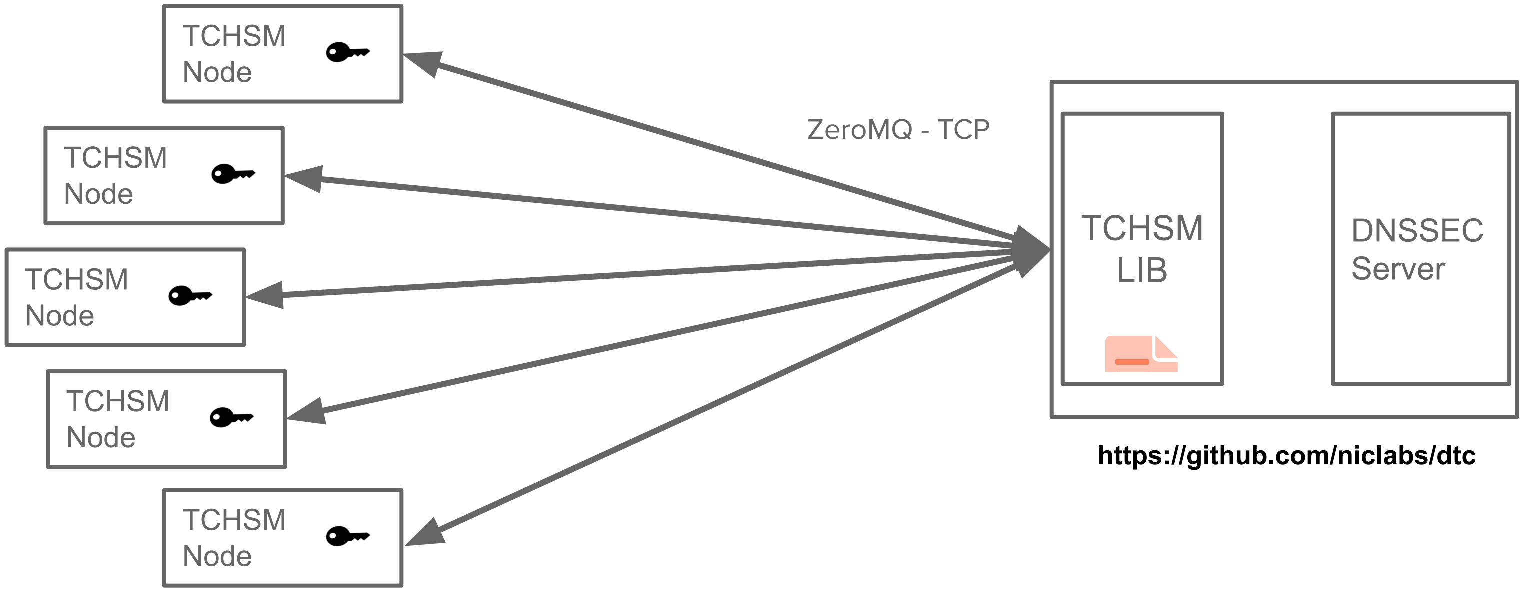


https://github.com/niclabs/dtc

https://github.com/niclabs/dtcnode

Our solution: Threshold-Cryptography Distributed HSM

https://niclabs.cl/tchsm/

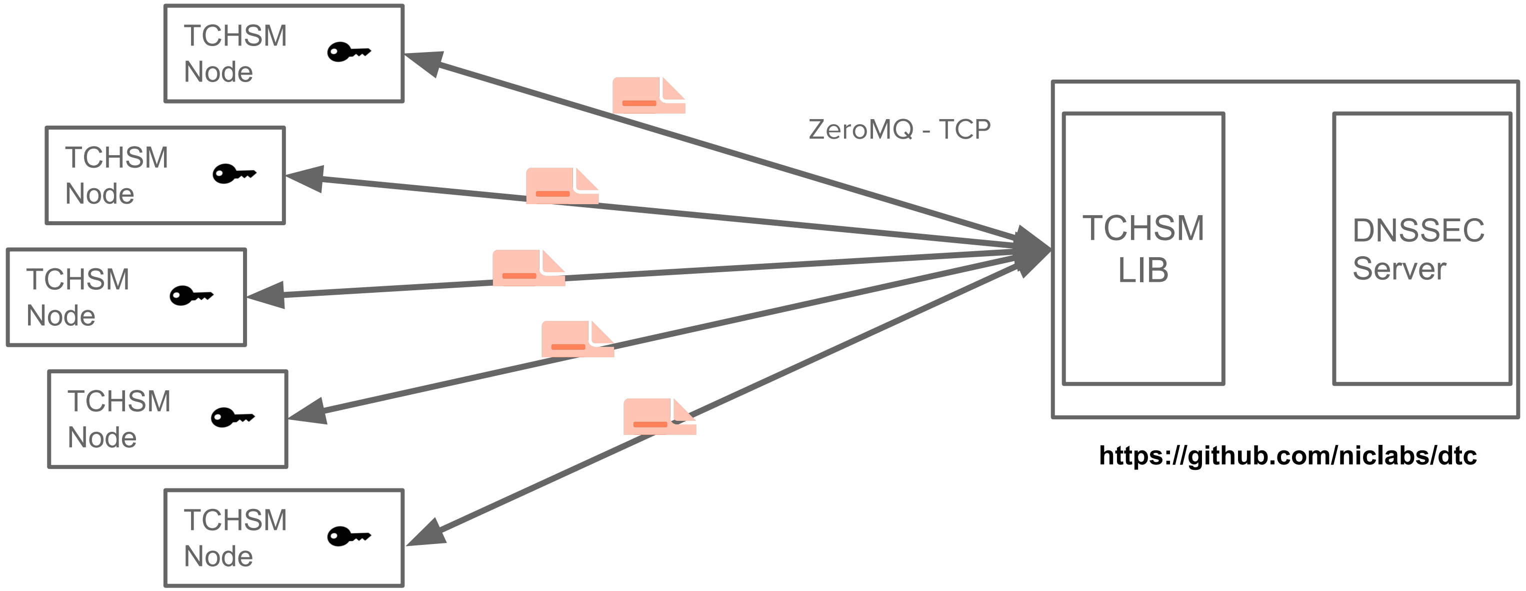


https://github.com/niclabs/dtc

https://github.com/niclabs/dtcnode

Our solution: Threshold-Cryptography Distributed HSM

https://niclabs.cl/tchsm/

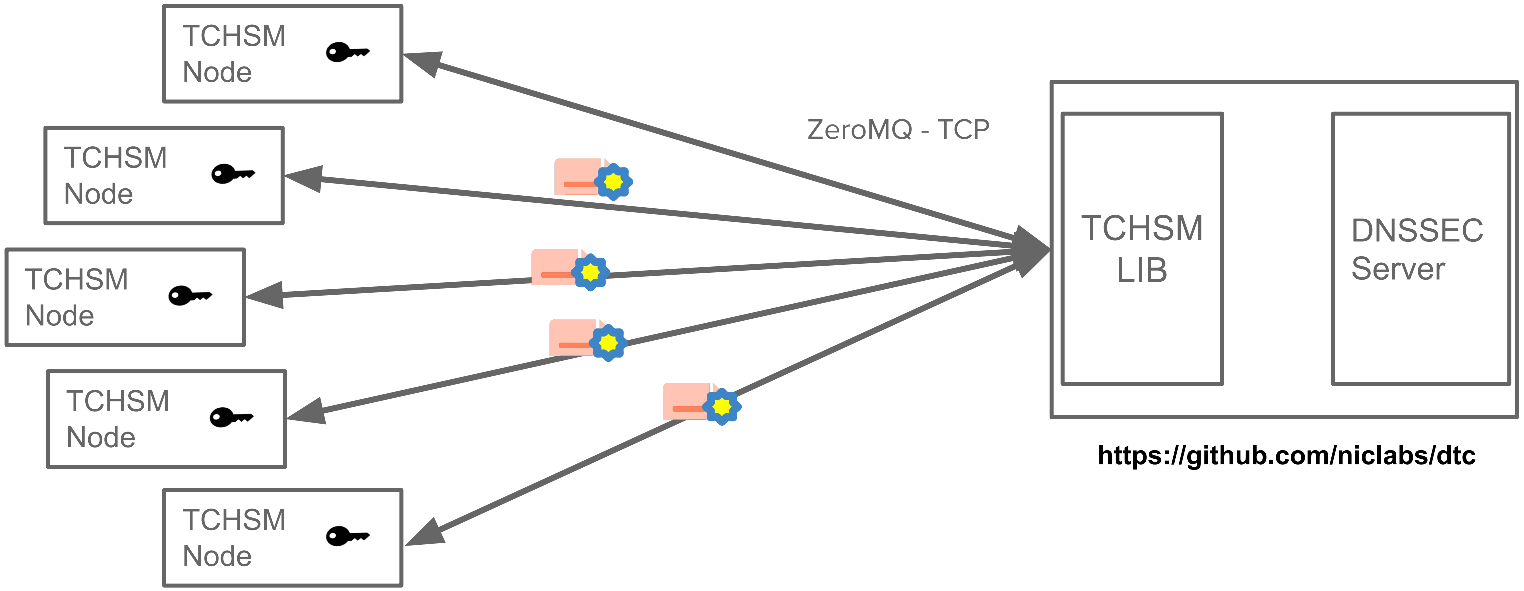


https://github.com/niclabs/dtc

https://github.com/niclabs/dtcnode

Our solution: Threshold-Cryptography Distributed HSM

https://niclabs.cl/tchsm/

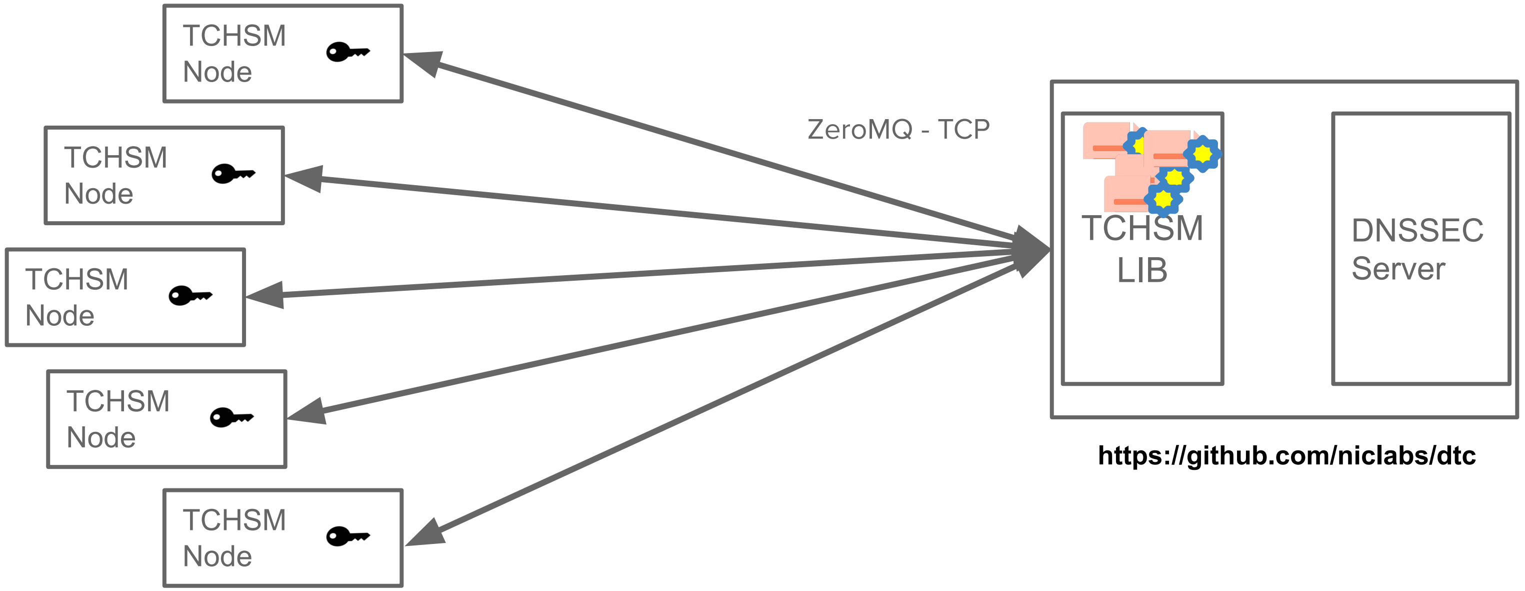


https://github.com/niclabs/dtc

https://github.com/niclabs/dtcnode

Our solution: Threshold-Cryptography Distributed HSM

https://niclabs.cl/tchsm/

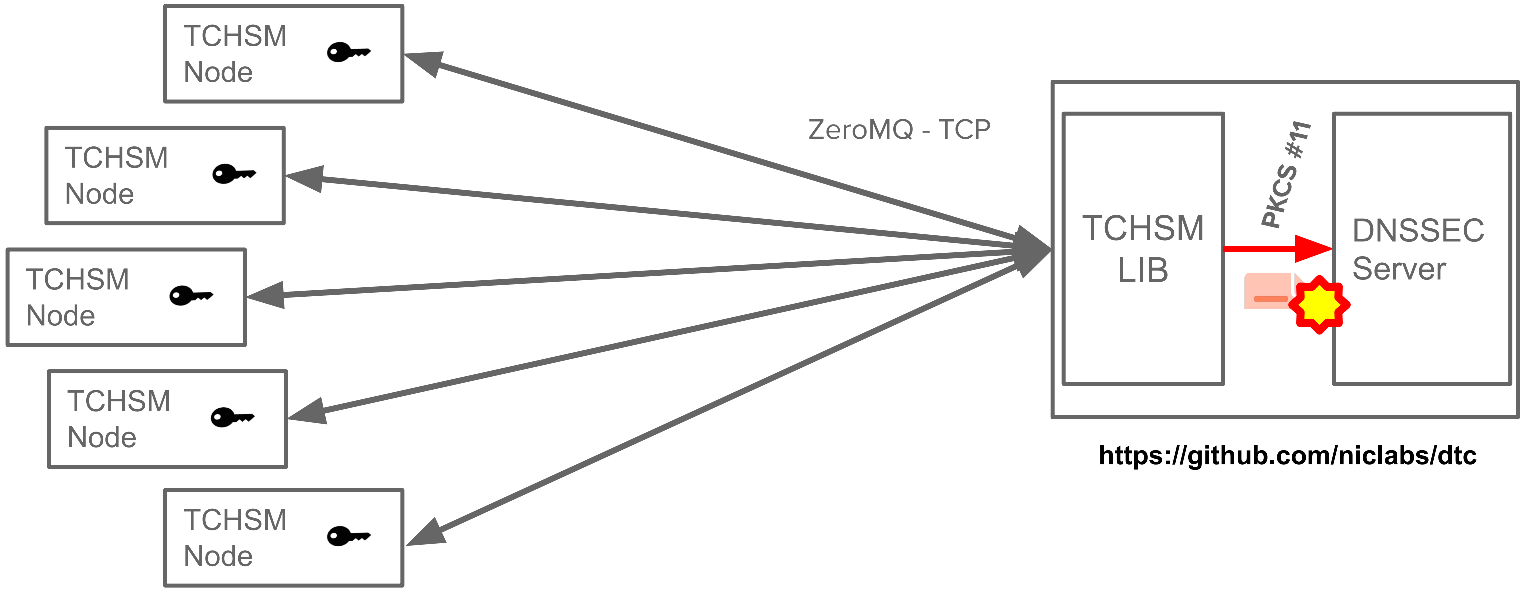


https://github.com/niclabs/dtc

https://github.com/niclabs/dtcnode

Our solution: Threshold-Cryptography Distributed HSM

https://niclabs.cl/tchsm/



https://github.com/niclabs/dtc

https://github.com/niclabs/dtcnode

Our solution: Threshold-Cryptography Distributed HSM

https://niclabs.cl/tchsm/

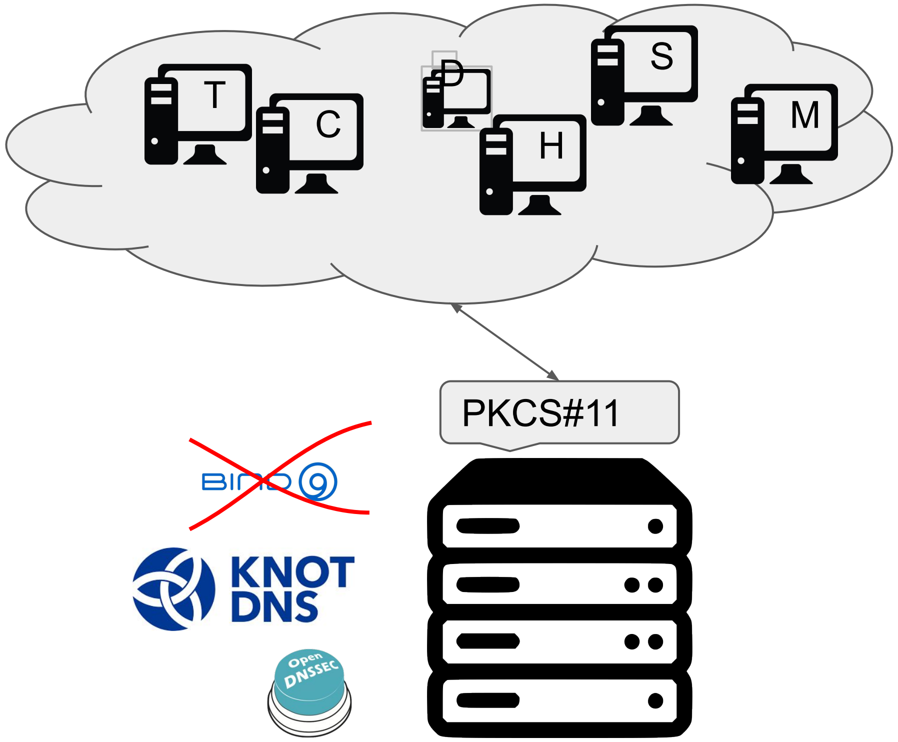
Developed first in C, then C++, now GoLang

Implements PKCS#11.

Implements RSA and ECDSA signatures.

Commodity hardware can be used: around 75 signatures per minute on an Intel® Core™ i5-2400 CPU @ 3.10GHz × 4.

Slower on this -->



Our solution: Threshold-Cryptography Distributed HSM

Q: Why this could be important for the DNS Research Testbed?

A: Root zone signature:

- Centralized key creation and “slicing”
- Distribute the slices using a secure protocol

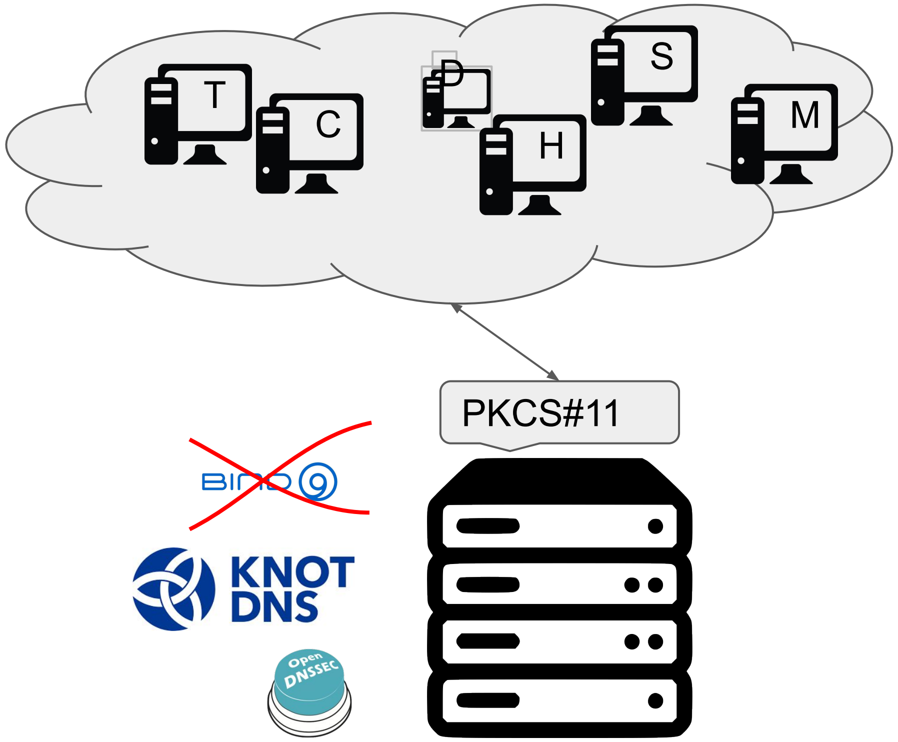
Or

- Distributed key-slices (parts) protocol

Then:

- Distributed root zone signature

Open Question: Quorum? n/2 + 1? 3n/4? 4n/5?



Also: dns-tools!!!

https://github.com/niclabs/dns-tools

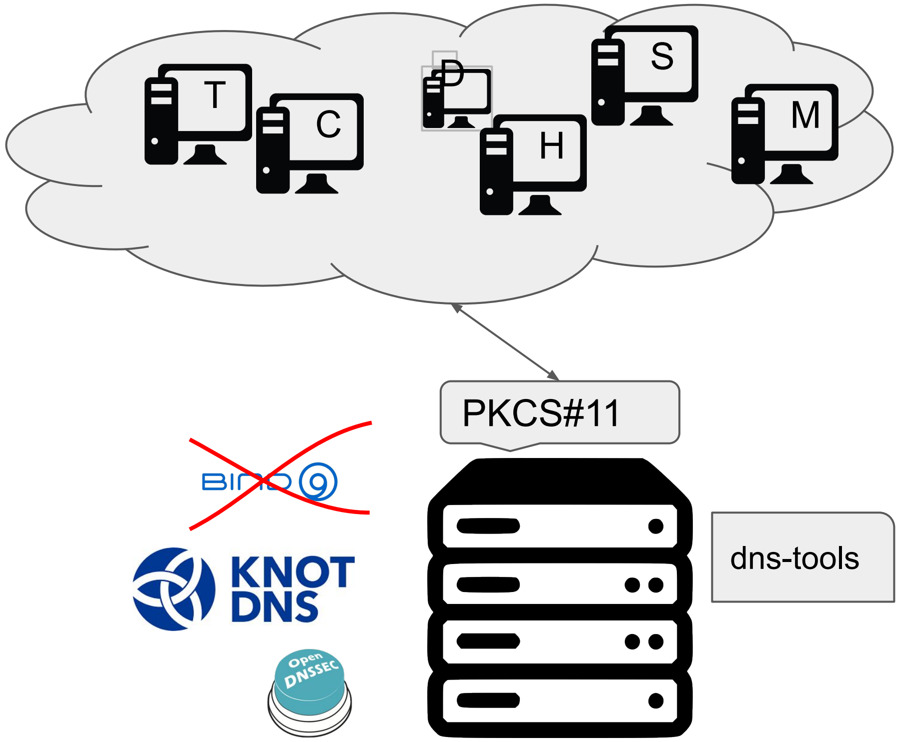
Developed in GoLang

First used to test TCHSM, then extended to a general purpose dns-tools (it was so easy to extend)

Implements PKCS#11 and key files.

Signs using NSEC or NSEC3.

Used to test experimental DNS behavior and procedures, e.g: ZONEMD zone digest
https://tools.ietf.org/html/draft-wessels-dns-zone-digest-00



https://niclabs.cl/tchsm/ :

- **https://github.com/niclabs/dtc**
- **https://github.com/niclabs/dtcnode**

https://github.com/niclabs/dns-tools

Thanks
Questions?

Javier Bustos-Jiménez & Hugo Salgado
jbustos@niclabs.cl, hsalgado@nic.cl
DNS and Internet Naming Research Directions (DINR) 2020