Distributed HSM

Yeti Meeting - march 2020

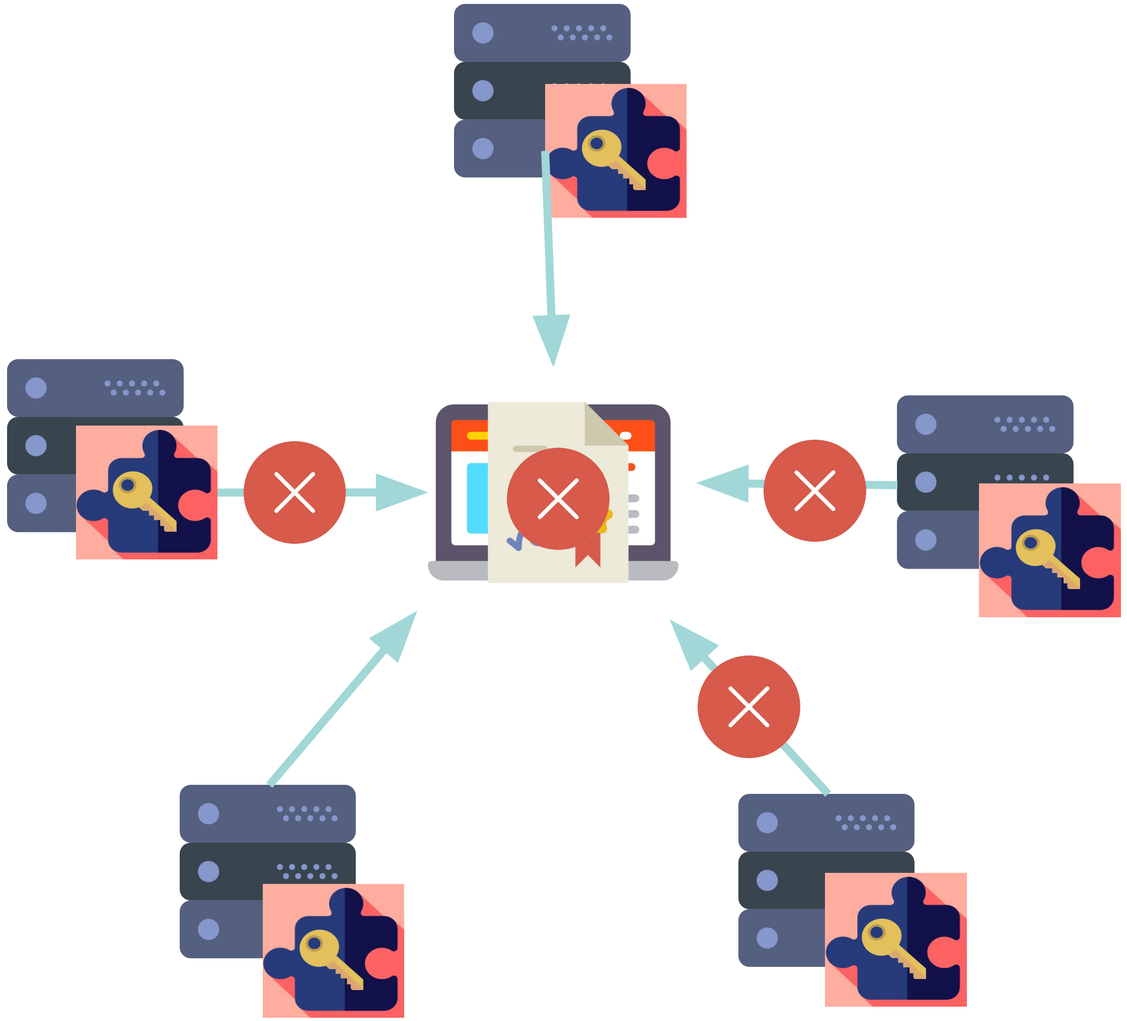
Hugo Salgado, .CL / NIC Chile
Author: Eduardo Riveros, NIC Chile Research Labs

NIC CHILE RESEARCH LABS

Threshold Cryptography

It allows to use a cryptographic key for signing, with a subset of distributed key shares.

The system is functional, even if not all the shares are retrieved. There is a minimum threshold of shares needed, usually **n/2+1**



NIC CHILE RESEARCH LABS

Some Threshold Cryptography Signing algorithms

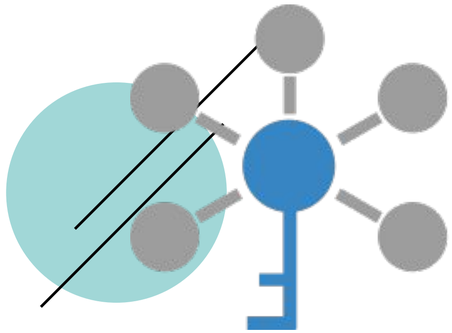
RSA: *Practical Threshold Signatures*
(V. Shoup)

- 1 Round *(Keygen and signing)*
- Joining *k SigShares* allows us to generate an individual signature.
- RSA is a very common algorithm for signatures

ECDSA: *Using Level-1 Homomorphic Encryption To Improve Threshold DSA Signatures For Bitcoin Wallet Security*
(D. Boneh, R. Gennaro, S. Goldfeder)

- 2 rounds (*Keygen*) and 4 rounds (*signing*)
- *SK* is distributed and encrypted with *threshold homomorphic encryption*
- ECDSA Signatures are **smaller** and **more secure** than RSA signatures.

NIC CHILE RESEARCH LABS



dHSM

Distributed Hardware Security Module

Collection of libraries that act like a HSM. They use threshold cryptography to sign documents and distribute the shares between a number of different devices

PKCS#11 API compilant
(OpenDNSSEC, KNOT)

Nodes communicate between them using ZMQ library

C++ and Go implementations

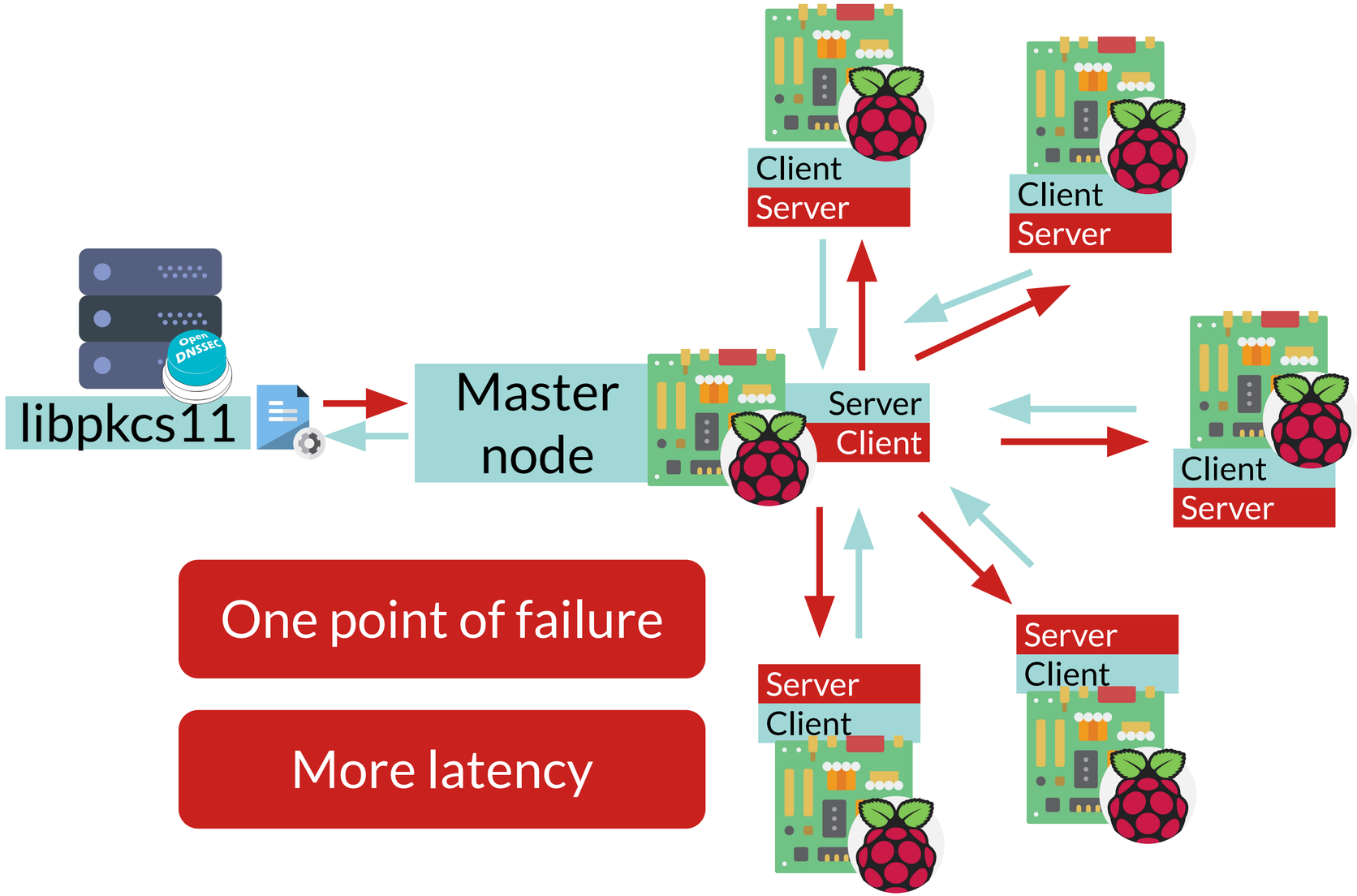
Cheaper deployment,
compared with real HSM.

NIC CHILE RESEARCH LABS

History

2014

C++ implementation with a separated master node



One point of failure

More latency

NIC CHILE RESEARCH LABS

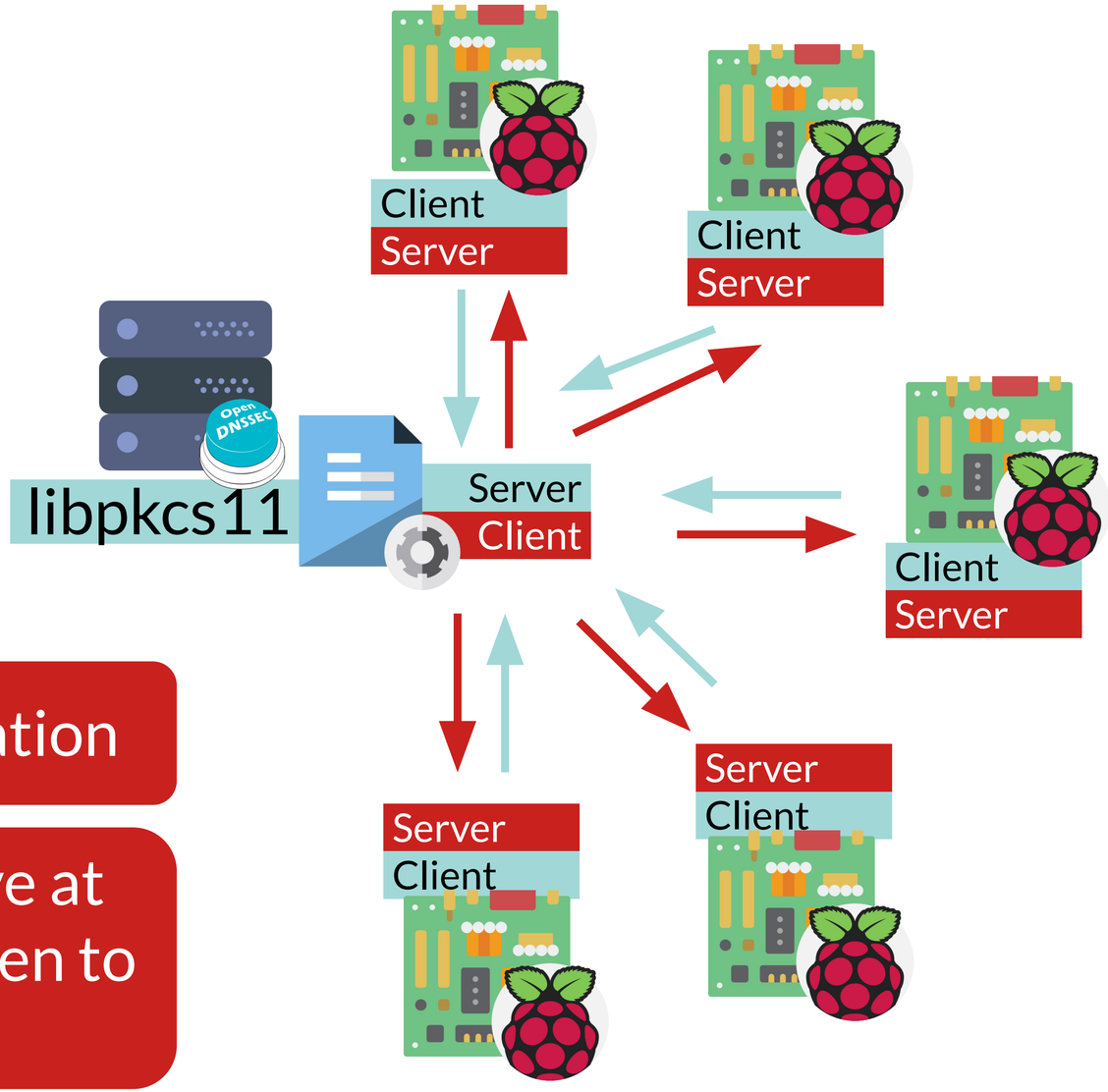
History

2016

C++ implementation without separated master node (client acts like the master node)

Complex installation

Signer must have at least one port open to nodes

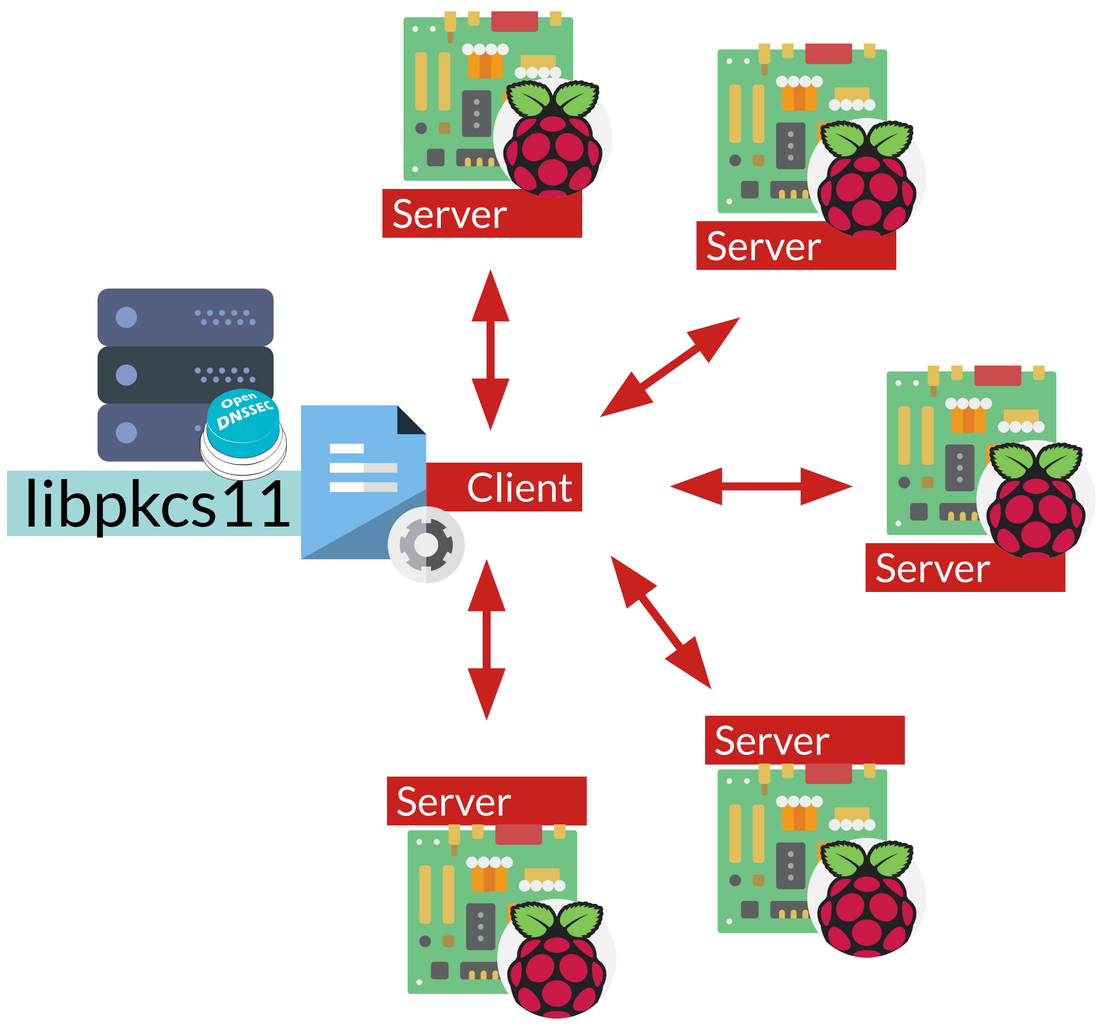


NIC CHILE RESEARCH LABS

History

2019

Reimplemented in Go
Go Modules
installation.
Library server deleted
(no need of open port).
RSA and ECDSA
signatures
implementation

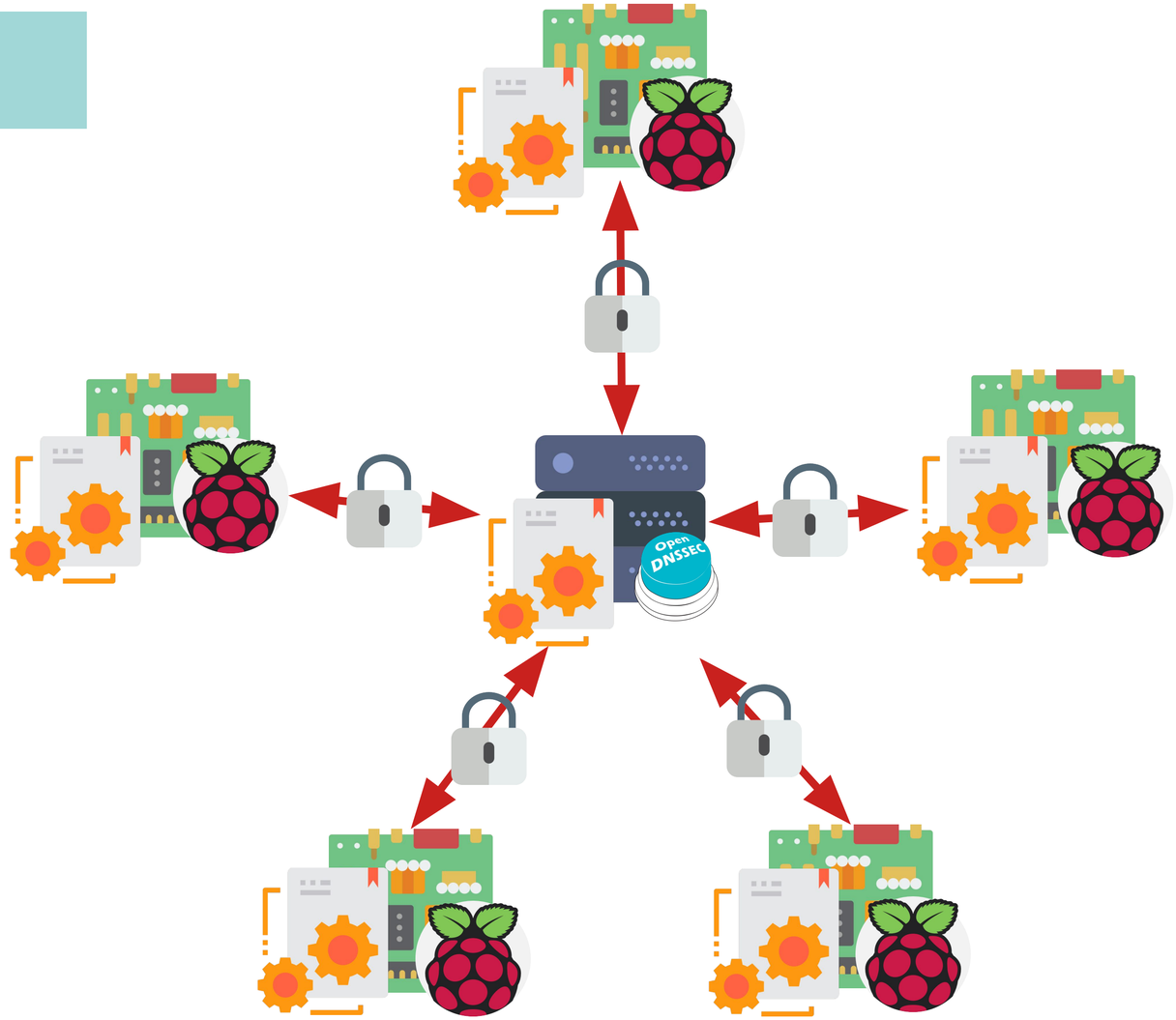


NIC CHILE RESEARCH LABS

How does it work?

Initial Configuration

- Compile DTC library in server which has installed OpenDNSSEC
- Compile DTCNode in each participant node
- Create config files with communication keys
- Configure OpenDNSSEC to use DTC as PKCS#11 library

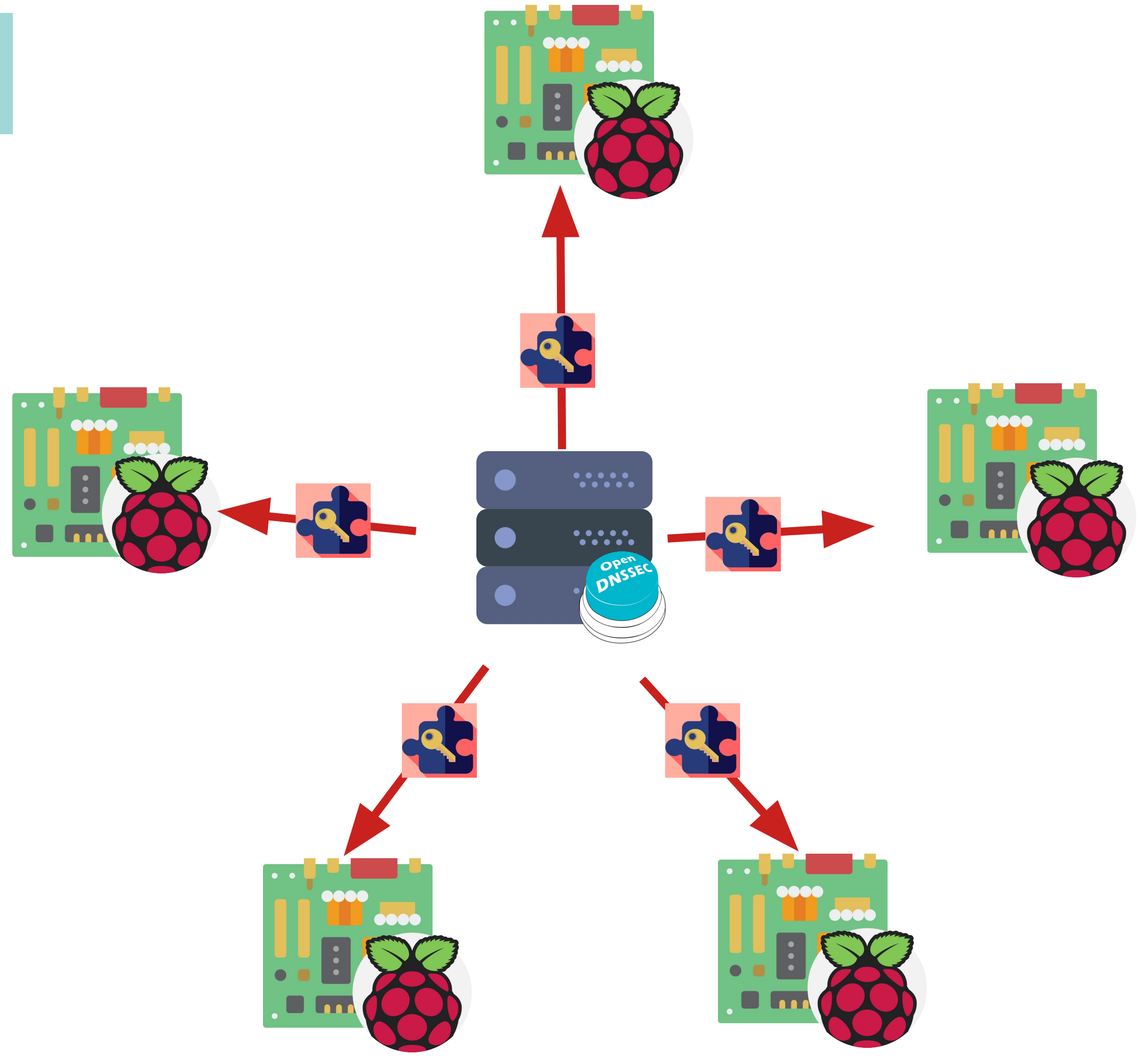


NIC CHILE RESEARCH LABS

How does it work?

Key Creation (RSA Case)

- OPENDNSSEC asks to the library to create a KSK key pair
- Library creates KSK key with **t-of-n threshold** , delivering key shares to all the connected nodes

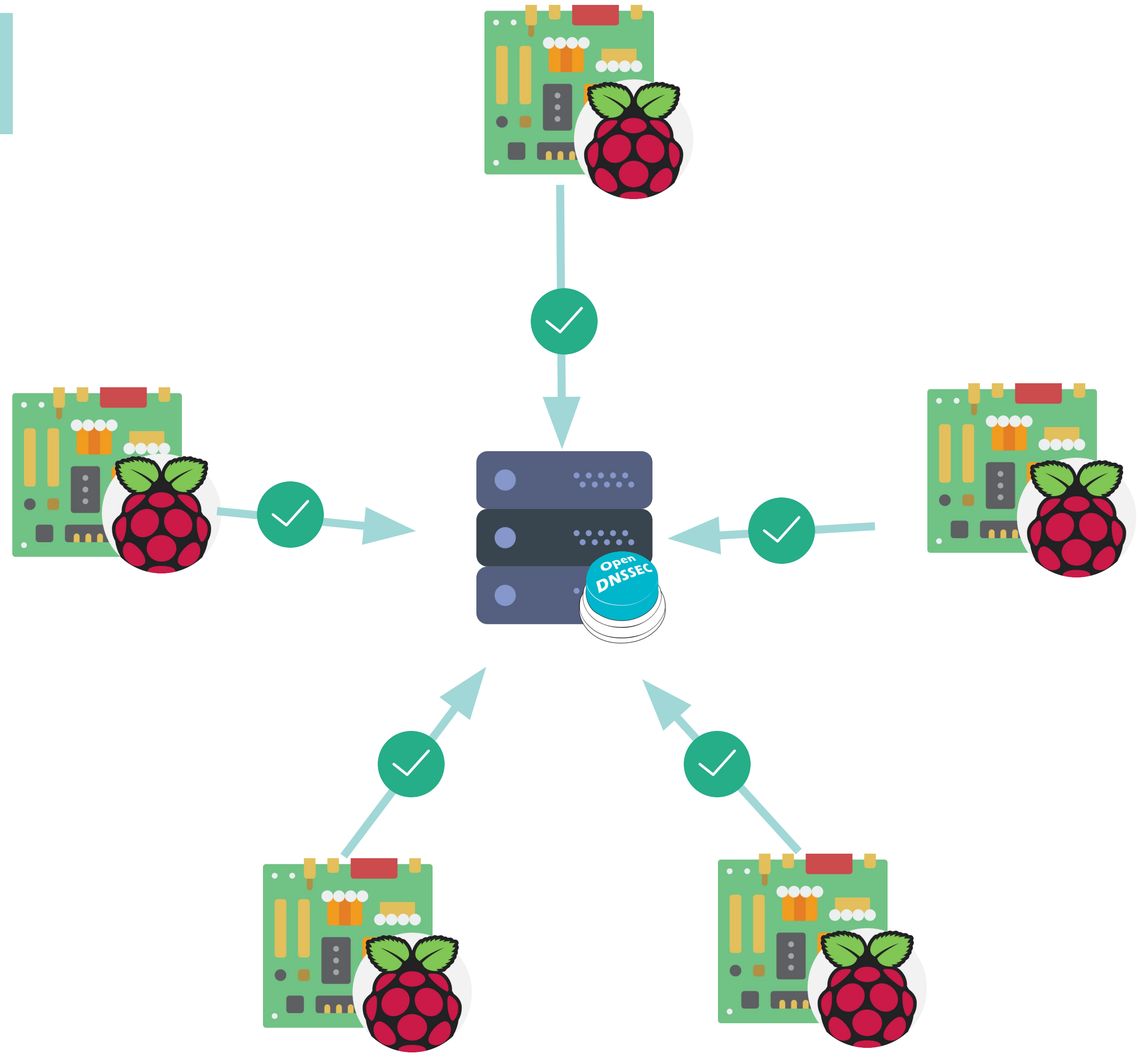


NIC CHILE RESEARCH LABS

How does it work?

Key Creation (RSA Case)

- The connected nodes ACK the successful storage of key shares.

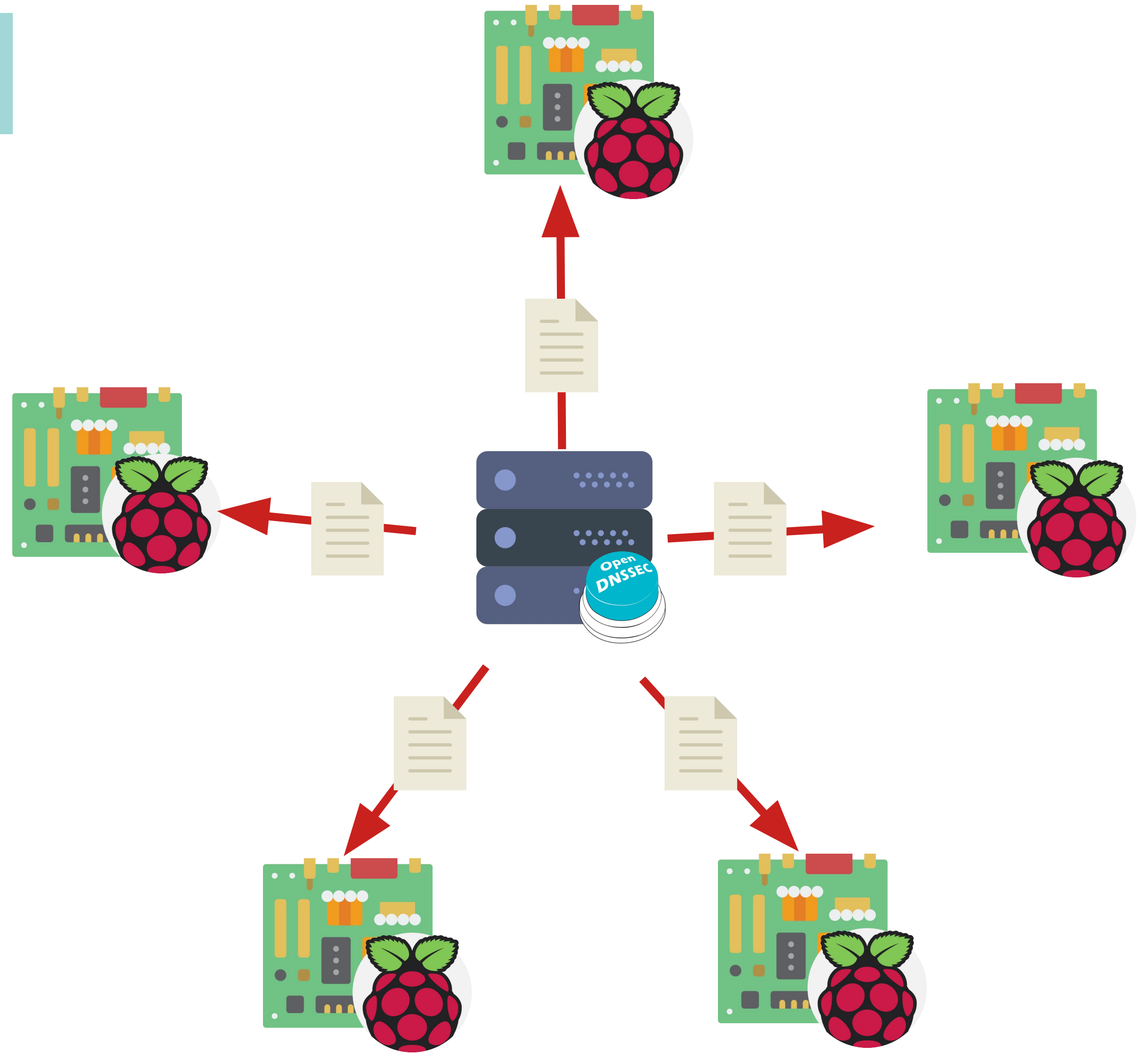


NIC CHILE RESEARCH LABS

How does it work?

Signing Process (RSA Case)

- OPENDNSSEC asks to the library to sign a RR
- Library sends a RR to be signed to all the connected nodes.

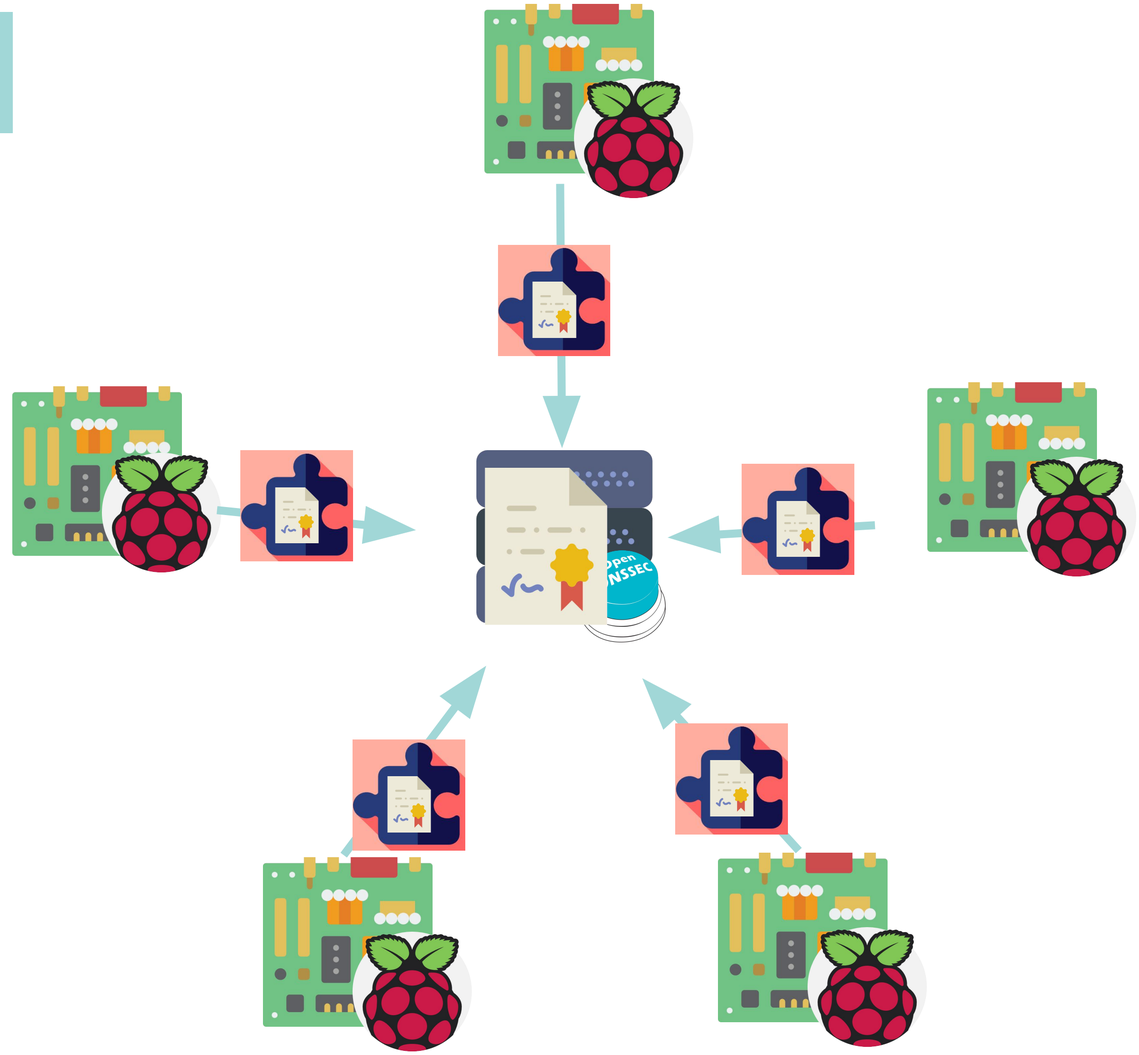


NIC CHILE RESEARCH LABS

How does it work?

Signing Process (RSA Case)

- The nodes sign the zone and return their signature shares.
- The shares are joined by the client, generating a signature.

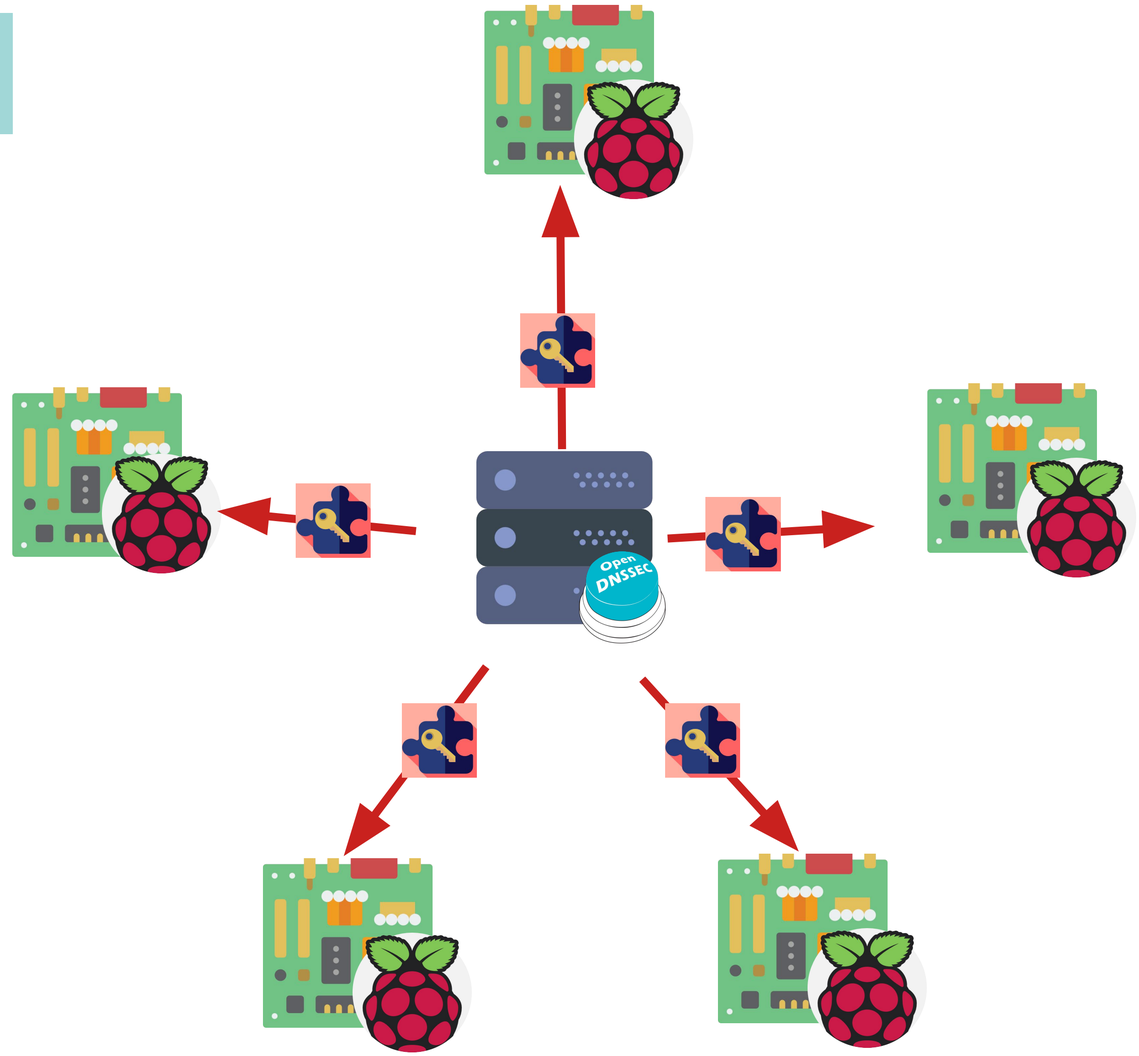


NIC CHILE RESEARCH LABS

How does it work?

Key Creation (ECDSA Case)

- OPENDNSSEC asks to the library to create a KSK key pair.
- Library generates a threshold Paillier's encryption scheme keypair, and distributes a SK share and the PK to each node.

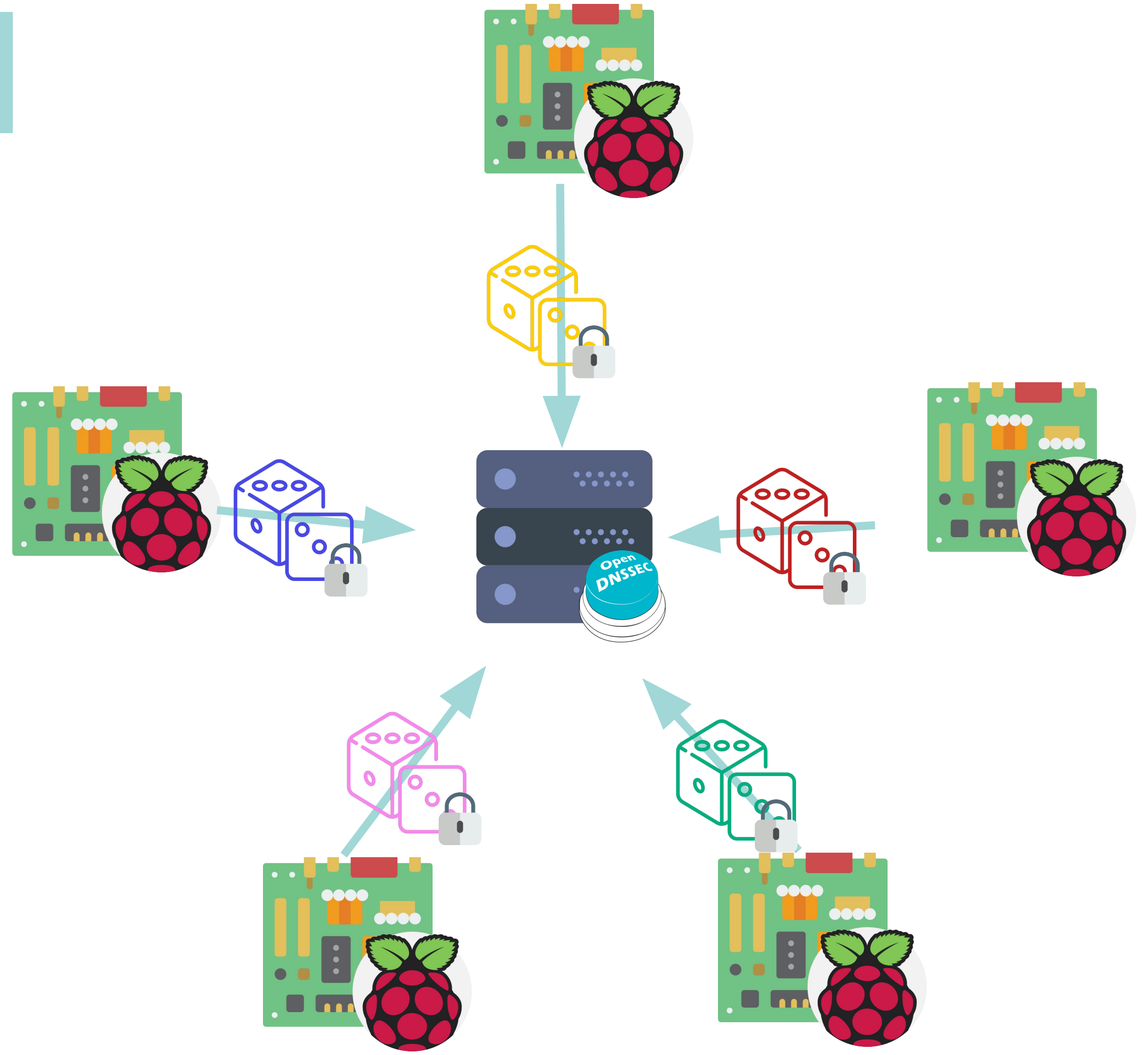


NIC CHILE RESEARCH LABS

How does it work?

Key Creation (ECDSA Case)

- Each node encrypts a random value with Paillier's PK and sends it to the client.

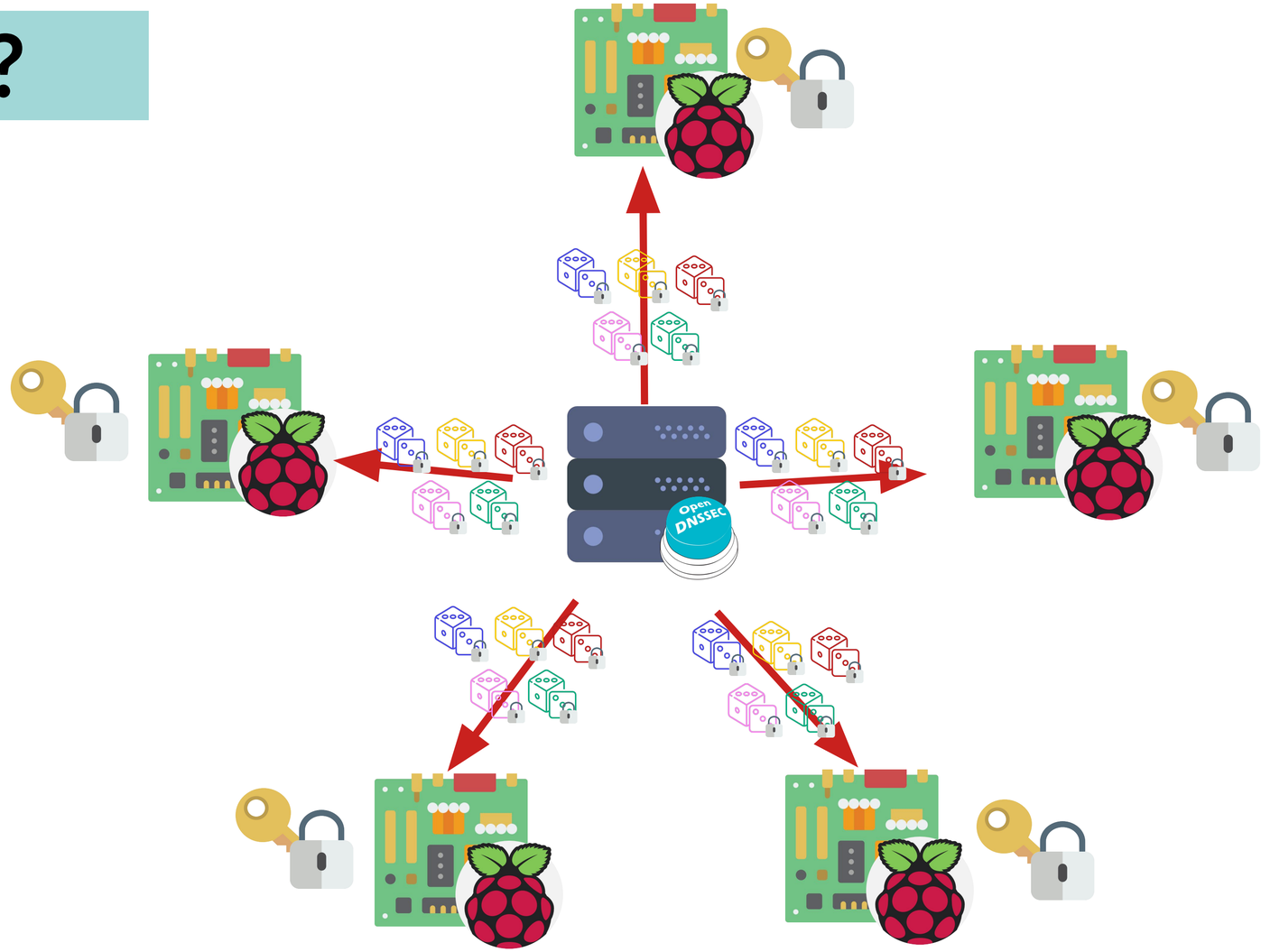


NIC CHILE RESEARCH LABS

How does it work?

Key Creation (ECDSA Case)

- The values are broadcasted by the client to all nodes, and and each node joins them homomorphically to form a Paillier Encrypted SK.

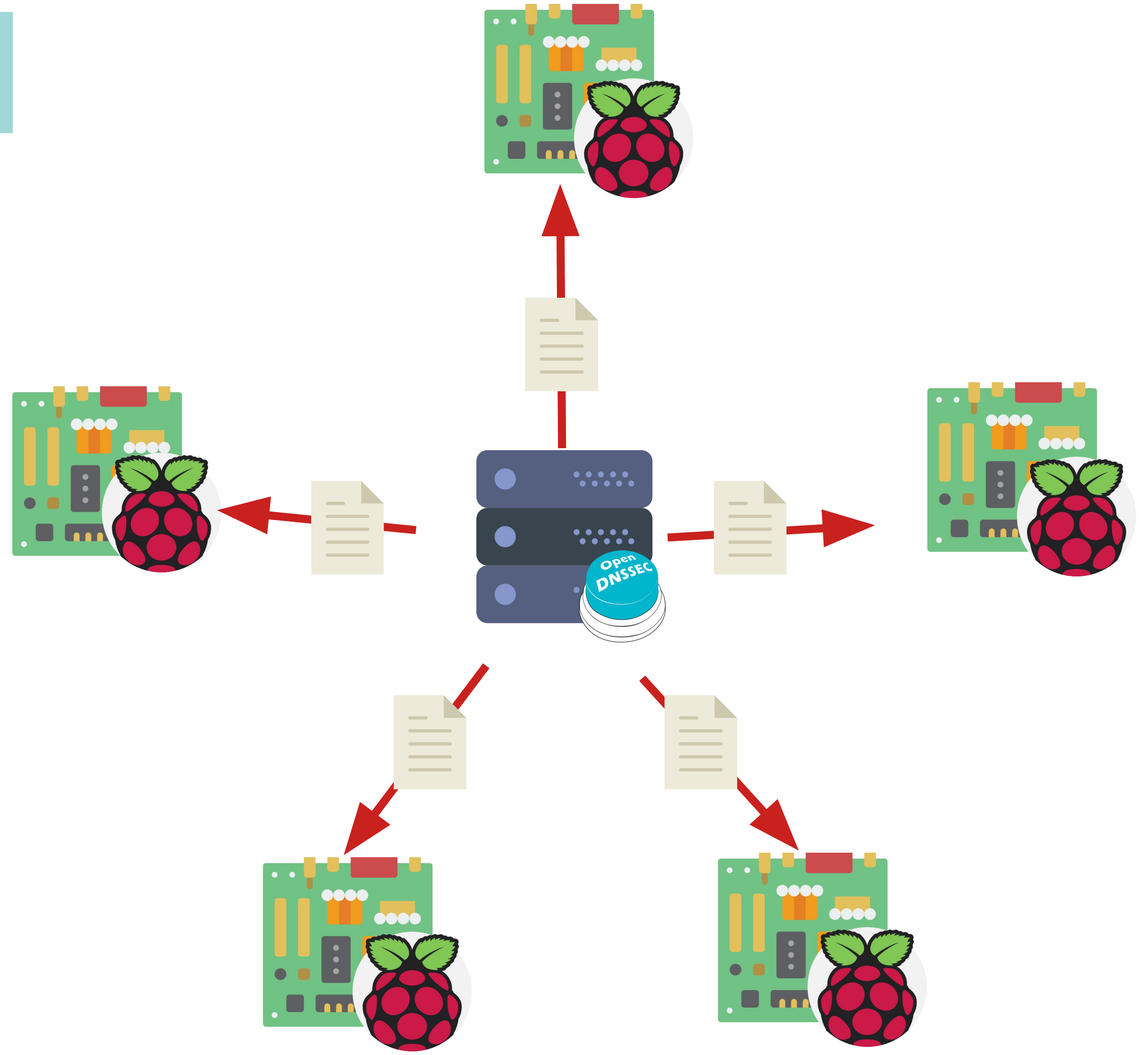


NIC CHILE RESEARCH LABS

How does it work?

Signing Process (ECDSA Case)

- OPENDNSSEC asks to the library to sign a RR.
- Library sends the RR to each node.

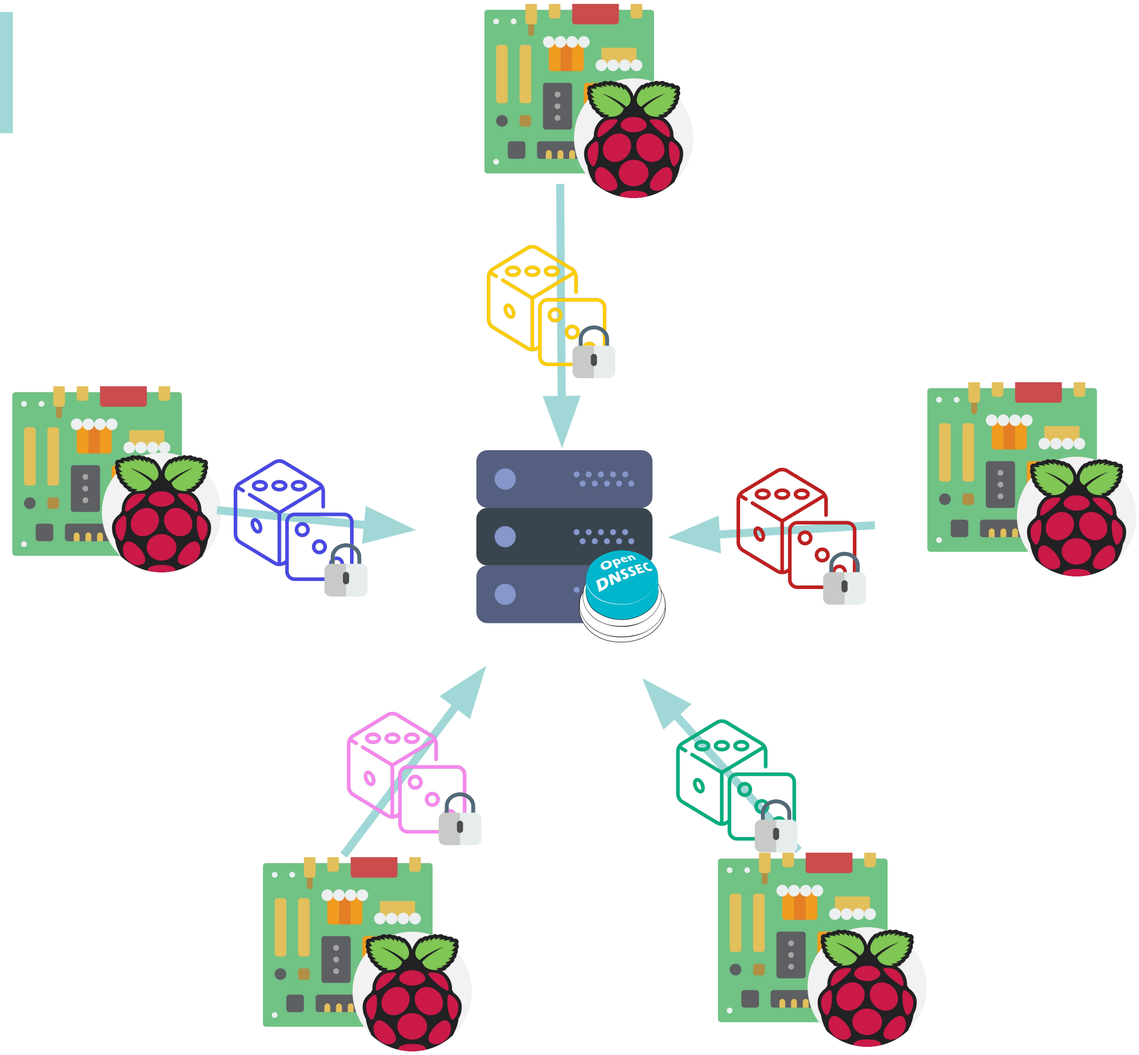


NIC CHILE RESEARCH LABS

How does it work?

Signing Process (ECDSA Case)

- Each node returns its own encrypted randomness to be used in the process.

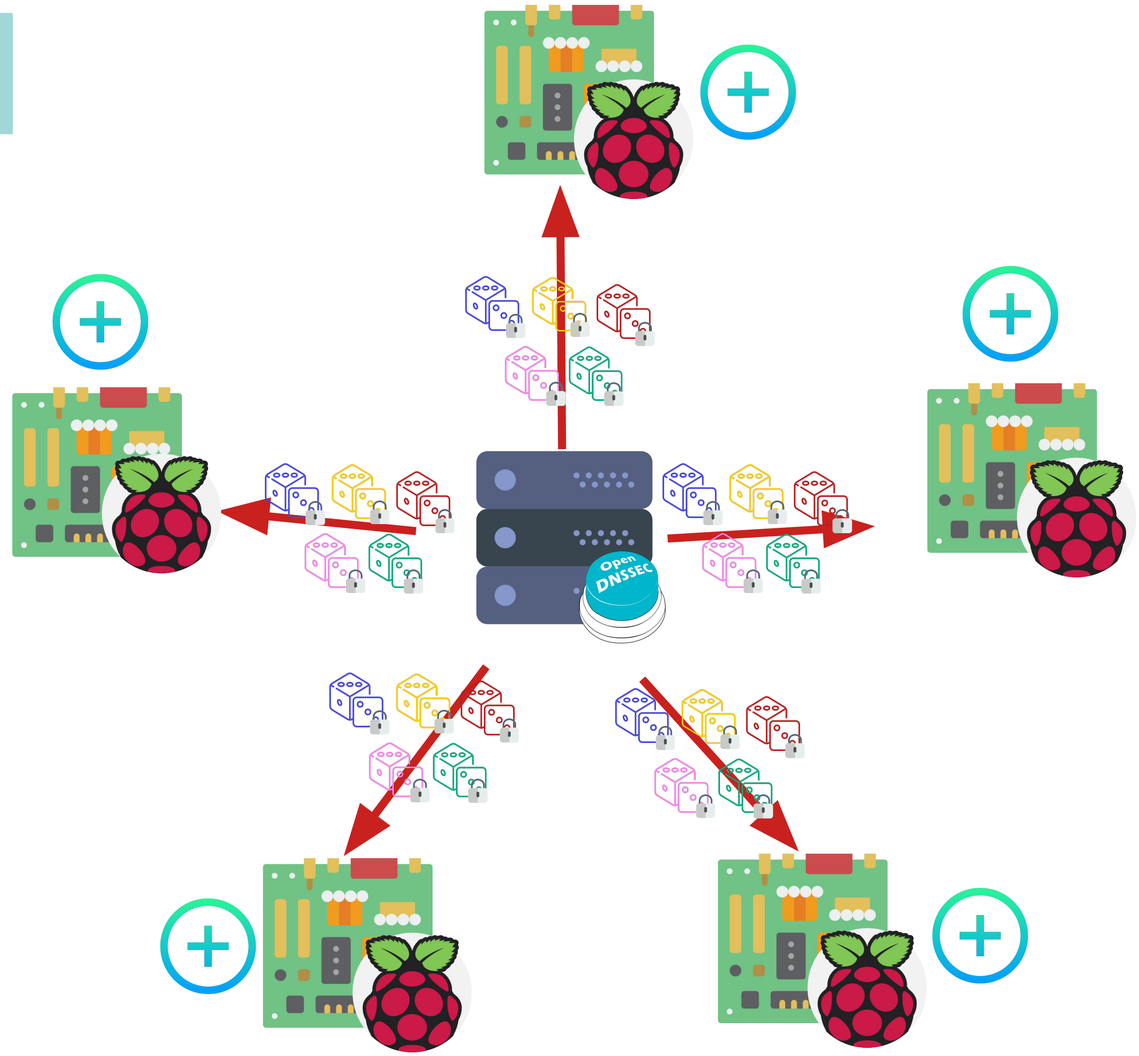


NIC CHILE RESEARCH LABS

How does it work?

Signing Process (ECDSA Case)

- The values are broadcasted by the client to other nodes and aggregated using Level 2 Paillier-based homomorphic encryption in 4 rounds.

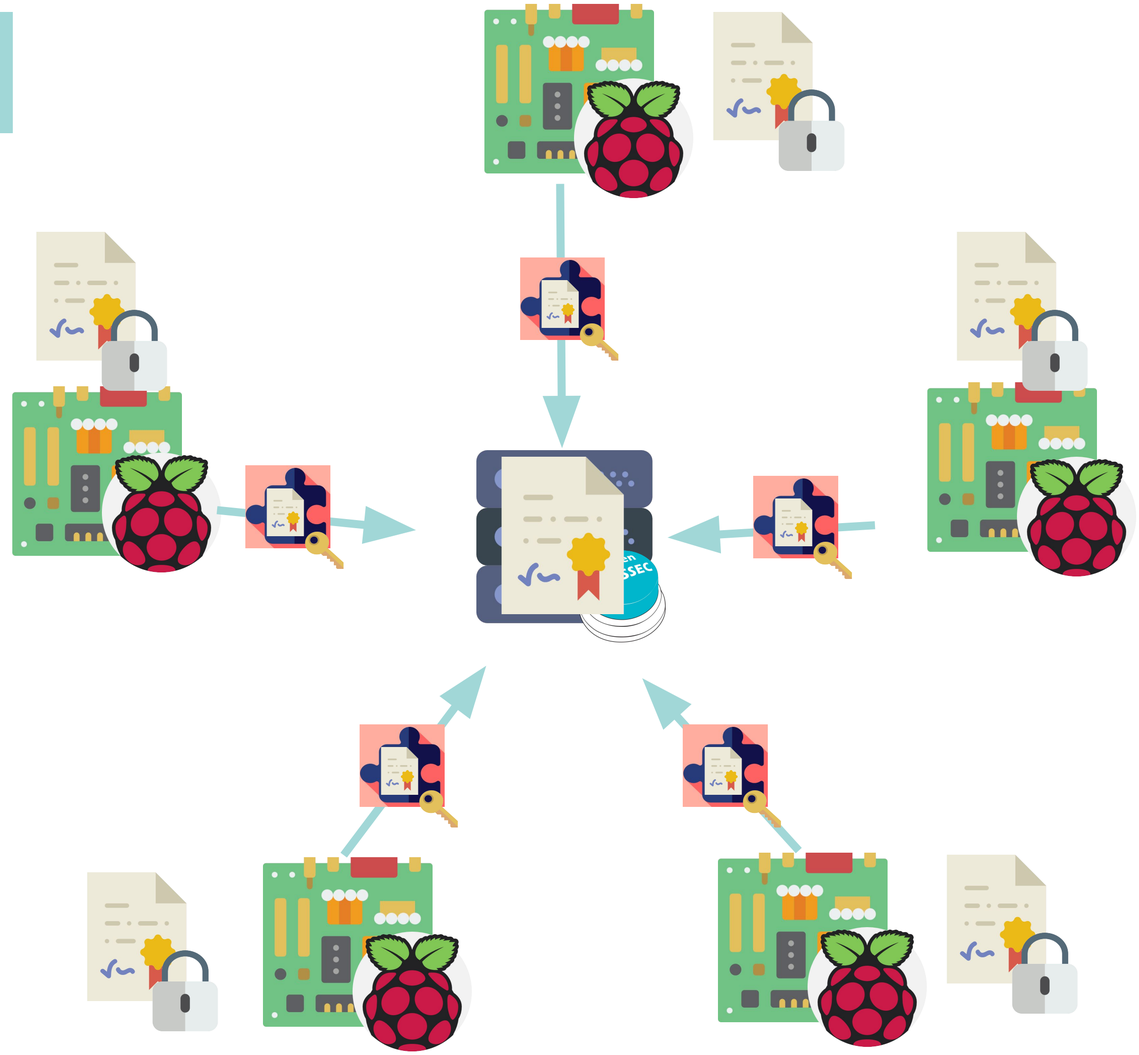


NIC CHILE RESEARCH LABS

How does it work?

Signing Process (ECDSA Case)

- After the four rounds, the client and all the nodes generate a Paillier-encrypted signature.
- The Nodes send decryption shares to the client to decrypt the signature.



NIC CHILE RESEARCH LABS

Components

All of them are available at
https://niclabs.cl/tchsm

Wiki and Documentation:
https://github.com/niclabs/dtc/wiki

TCRSA

Uses Go native libraries only

DTCNODE

Process that runs on each node

TCPAILLIER

Threshold Paillier Cryptosystem impl.

TCECDSA

Threshold ECDSA Signing impl.

DTC

PKCS#11 library

HSM-SIGNER

Go implemented zone signer

NIC CHILE RESEARCH LABS

Features

Measurements on PC with seventh gen Intel Core i7 processor, Ubuntu 18.04 and 8 GB of RAM

Compile Time

~3 minutes

Key Generation

~10s RSA/1024
~3m RSA/2048
~3m ECDSA/256

Signing Time

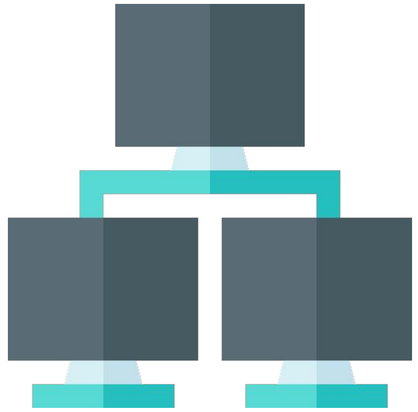
~100 signatures/second RSA
~25 signatures/second ECDSA

NIC CHILE RESEARCH LABS

Future Work

Implement other cryptographic schemes

(That support threshold crypto)



Implement other communication and storage platforms

(Besides ZMQ and SQLite)

Try other uses for the library

(Apart from DNSSEC)

NIC CHILE RESEARCH LABS

Acknowledgements

Fco. Cifuentes y Fco. Montoto

Firsts designs and implementations of DTCHSM.

Eduardo Riveros

Implementation and design of current version of DTCHSM.

Flaticon

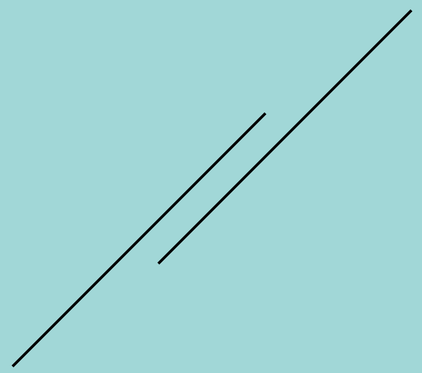
Icons from *Smashicons*, *itim1201* and *Freepik*, Creative Commons on https://flaticon.com

NIC CHILE RESEARCH LABS

Thanks

Hugo Salgado
hugo@nic.cl

Eduardo Riveros
Slides author
eduardo@niclabs.cl



NIC CHILE RESEARCH LABS