BRewSKI

Bootstrapping Remote Secure Key Infrastructure With RUST... on RIOT-OS...

RIOT-OS 2019
Helsinki

By

Michael Richardson
<mcr@sandelman.ca>

https://www.sandelman.ca/SSW/talks/riotos2019-brski/html

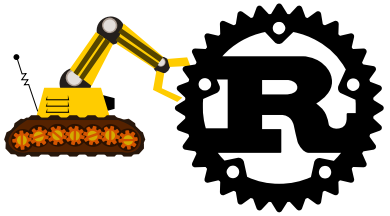
This Talk

Who am I?

Why am I doing this?

What is the problem?

What about RUST?



What about RIOT-OS?

Who am I?

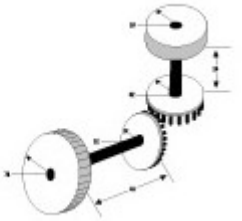
Internet technologist, doing IP since 1988. “Garage Entrepreneur”

Xelerance Corp 2003-2007,2014-

xelerance

SSH.COM
1997-1998

SANDELMAN SOFTWARE WORKS



1996-

SOLIDUM
(1998-2001)

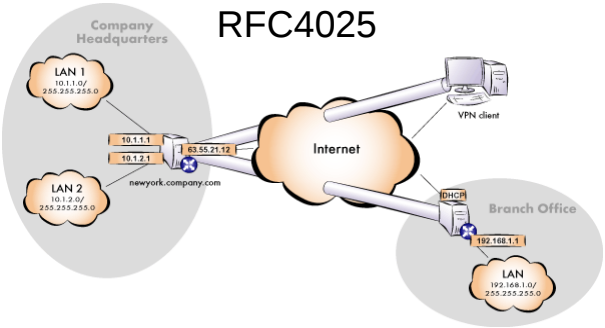
FreeS/WAN (2001-2004)

Linux FreeS/WAN

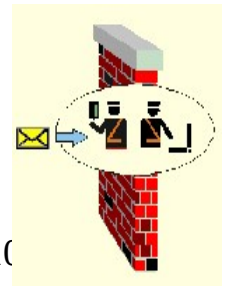
SIMtone CORPORATION

CENTRE DE RECHERCHE ET DÉVELOPPEMENT EXPÉRIMENTAL EN INFORMATIQUE LIBRE
CREDIL
CENTRE FOR RESEARCH AND EXPERIMENTAL DEVELOPMENT IN INFORMATICS LIBRE

RFC4322
RFC4025



#4 at Milkyway Networks (1994)



ROLL – RFC6550
2012-

6TiSCH

IETF standard security:IPsec/VPN

What else do I do?
--> IETF things

- minimal-security
- zerotouch-join

- RFC8520, Manufacturer Usage Description

- ANIMA
  - RFC8366 vouchers
  - BRSKI
  - constrained-voucher

Zero-Touch IoT onboarding

What is the problem?

What is the problem?

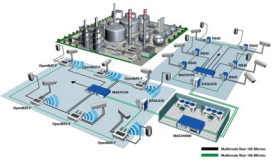
New Device

What is the problem?

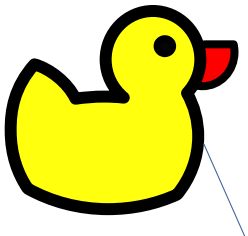
New Device

What is the problem?

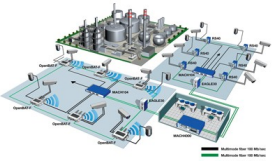
New Device



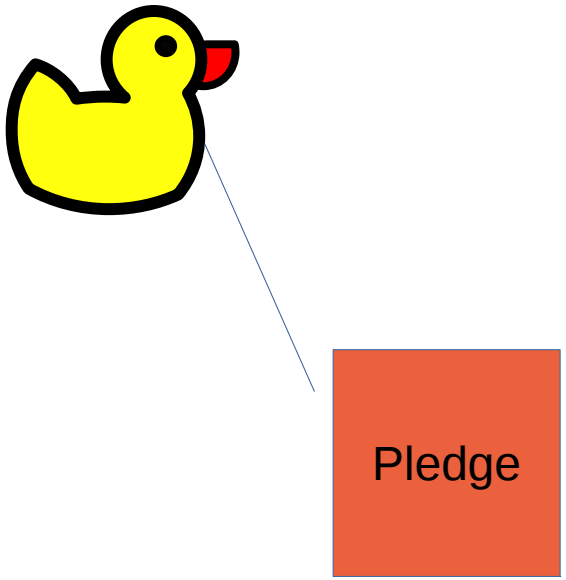
What is the problem?



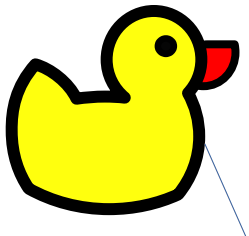
Pledge



What is the problem?



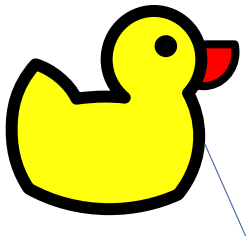
What is the problem?



Pledge

Network Operator (owner/ resident)

What is the problem?

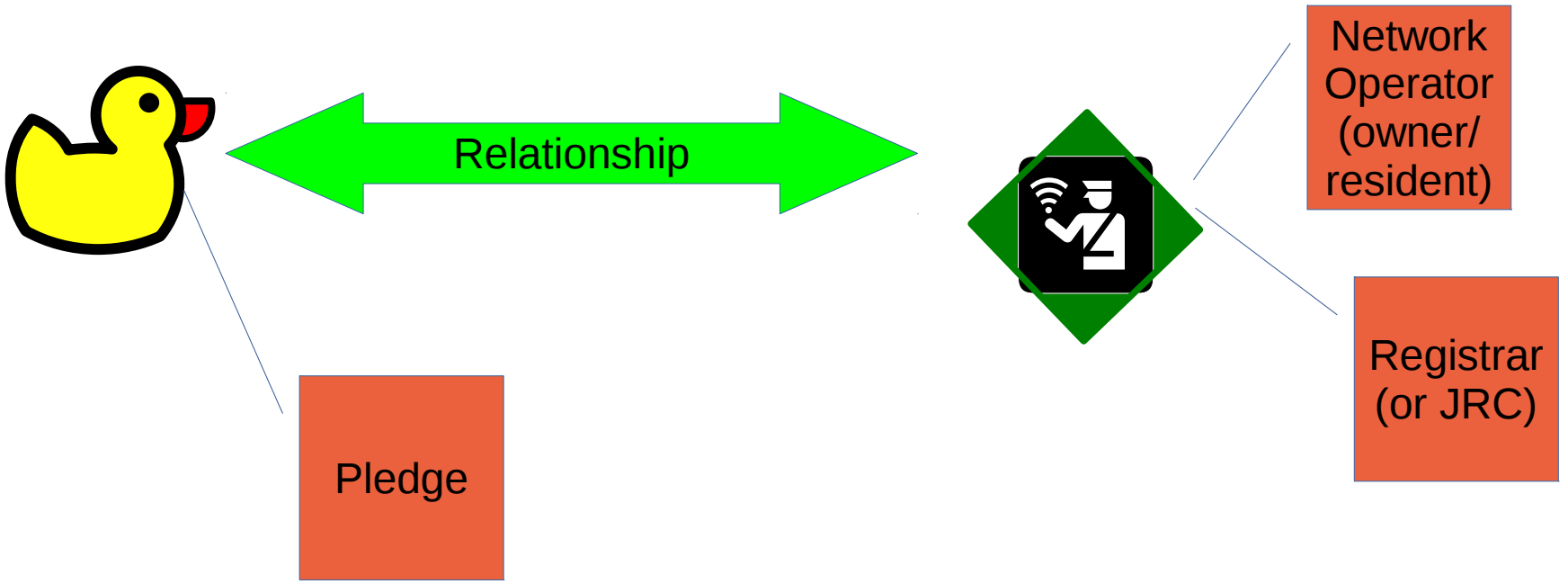


Pledge

Network Operator (owner/ resident)

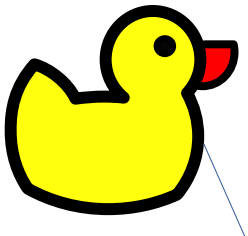
Registrar (or JRC)

What is the problem?



What is the problem?

Finally! I'm in a
Relation Ship



Relationship

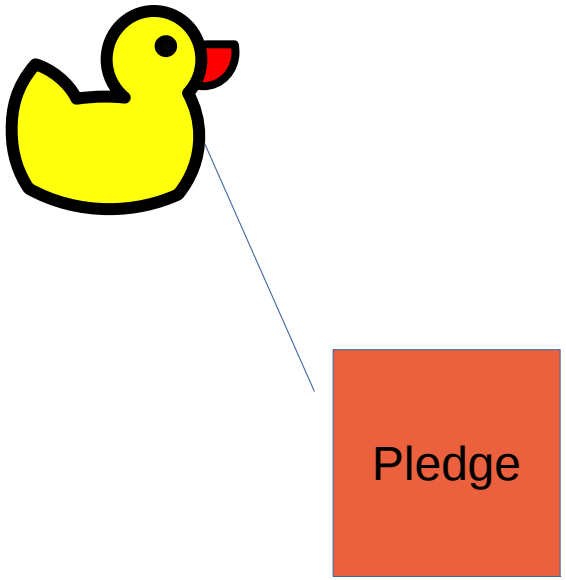
Network Operator (owner/ resident)

Registrar (or JRC)

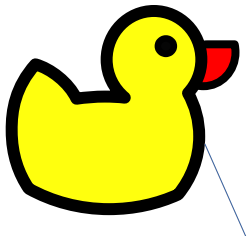
Pledge

BRSKI in animation

BRSKI in animation



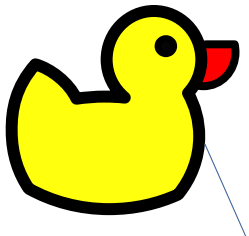
BRSKI in animation



Pledge

Network Operator (owner/ resident)

BRSKI in animation

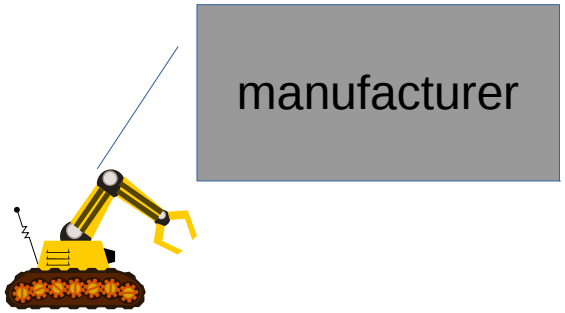


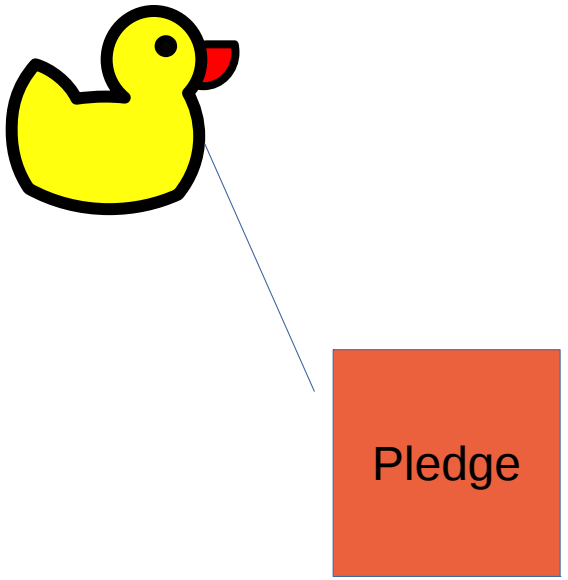
Pledge

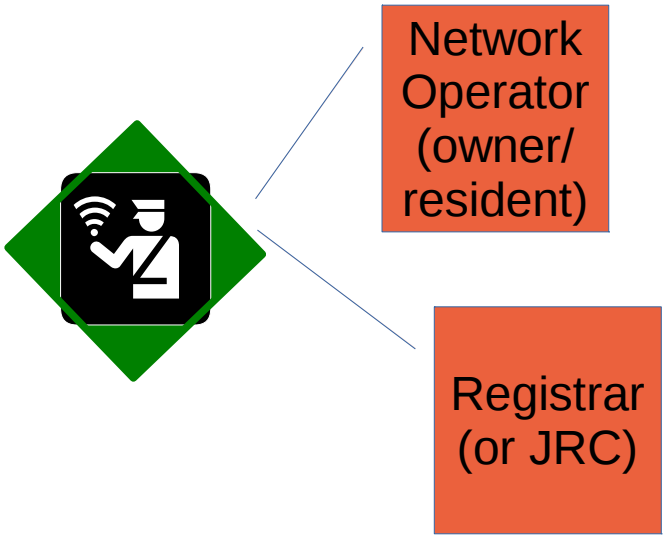
Network Operator (owner/ resident)

Registrar (or JRC)

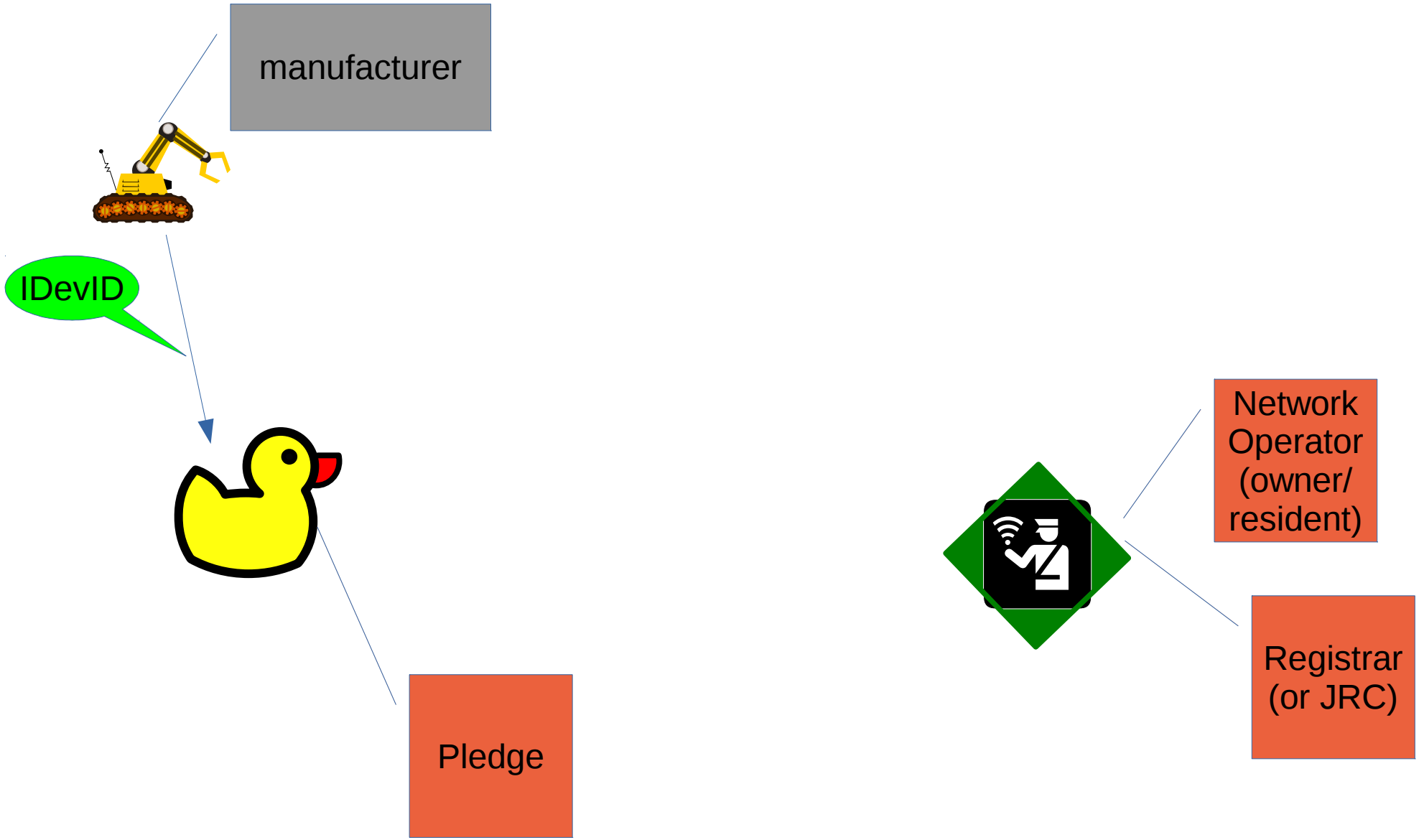
BRSKI in animation



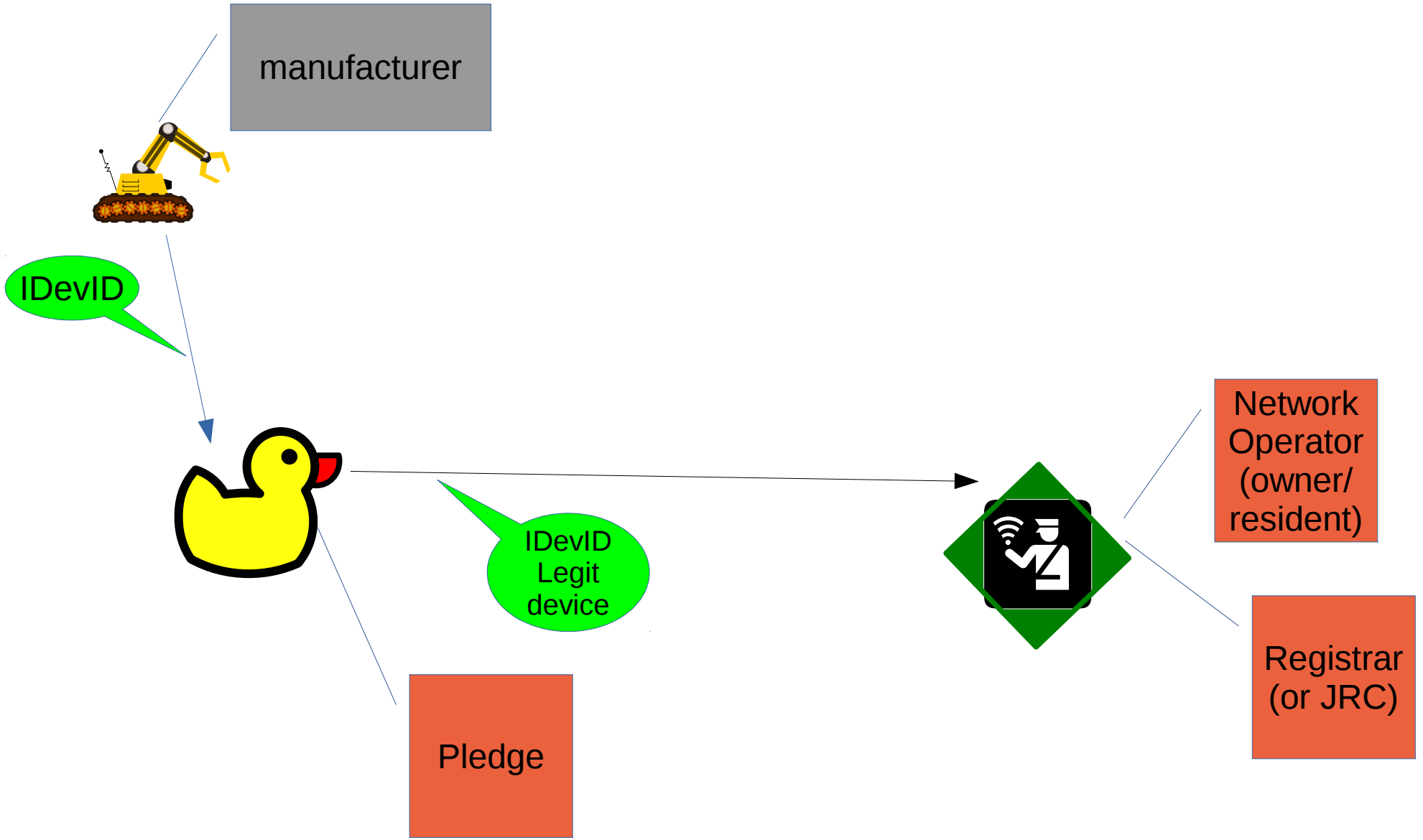




BRSKI in animation

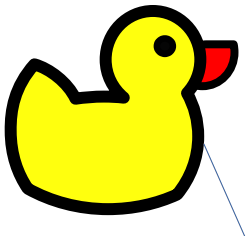


BRSKI in animation



BRSKI in animation

MASA
(manufacturer Authorized Signing authority)

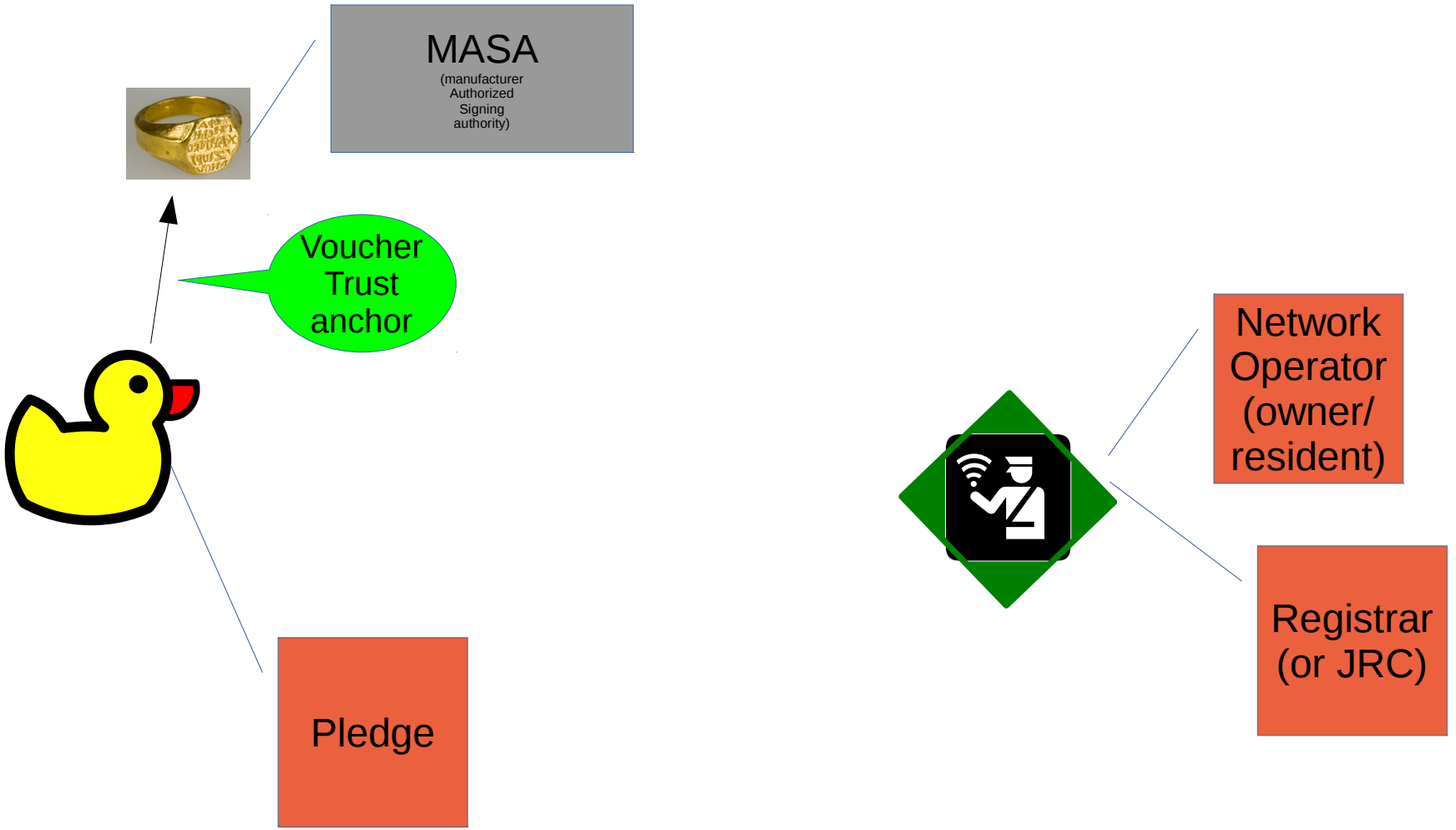


Pledge

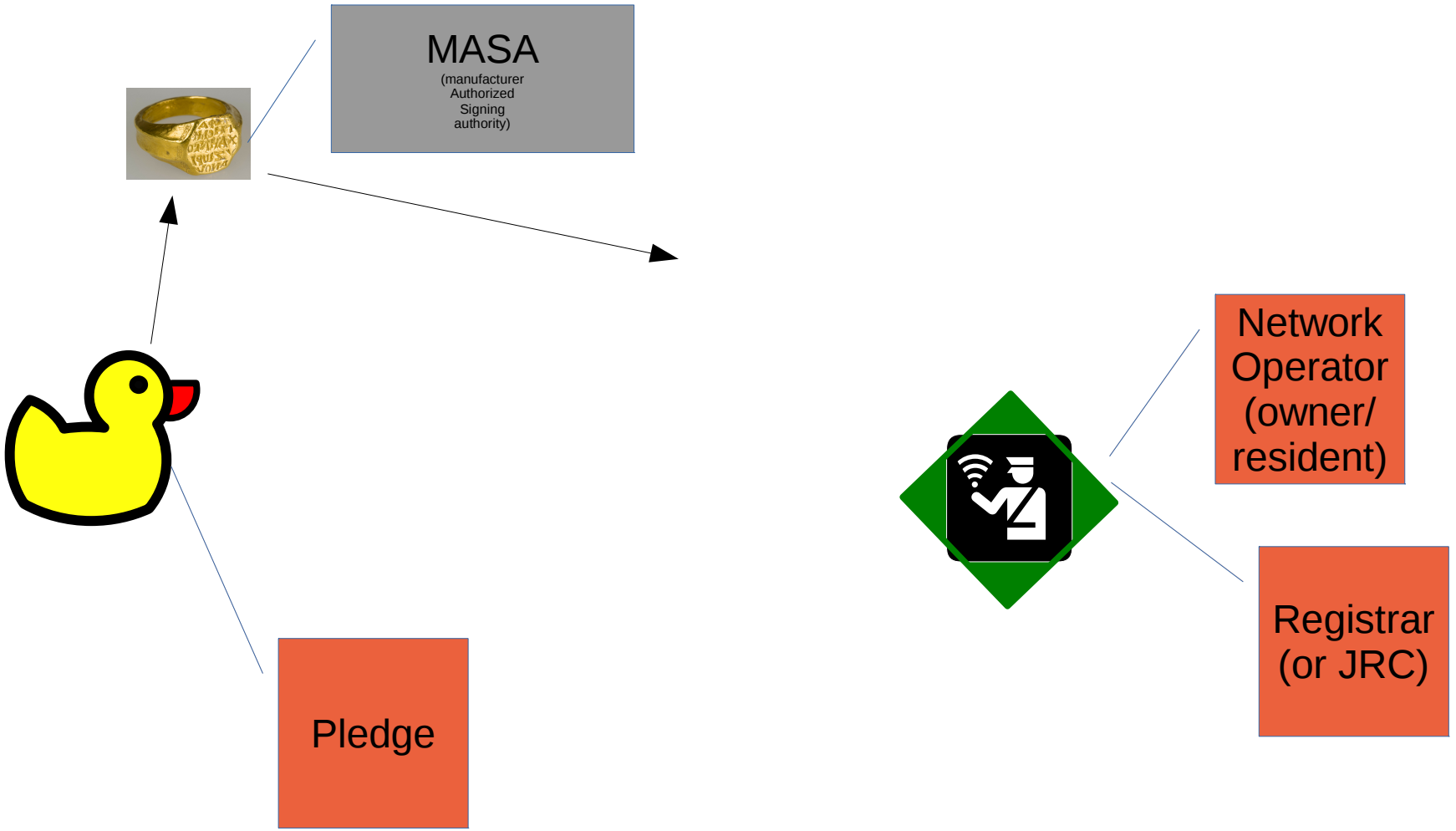
Network Operator (owner/ resident)

Registrar (or JRC)

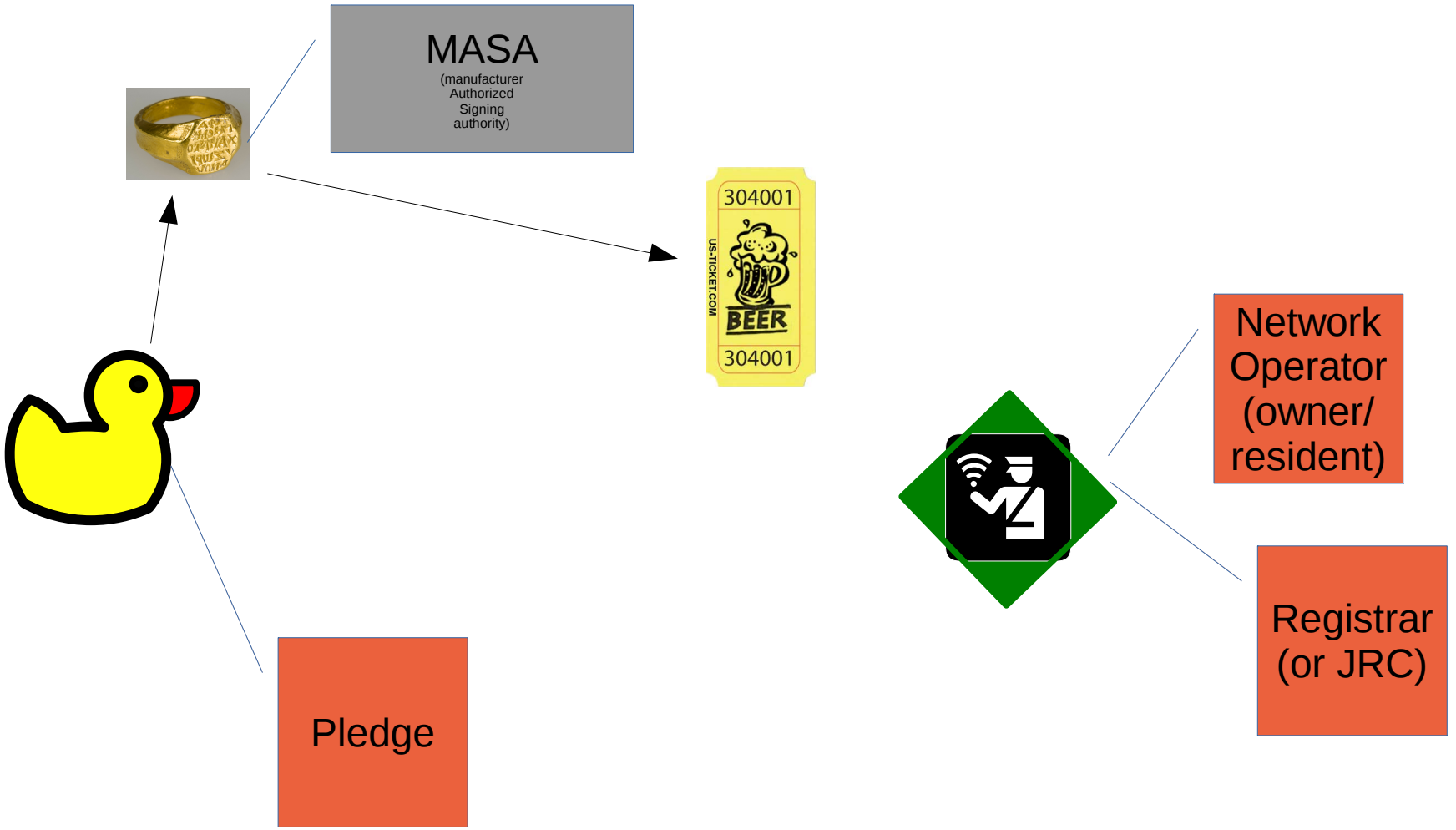
BRSKI in animation



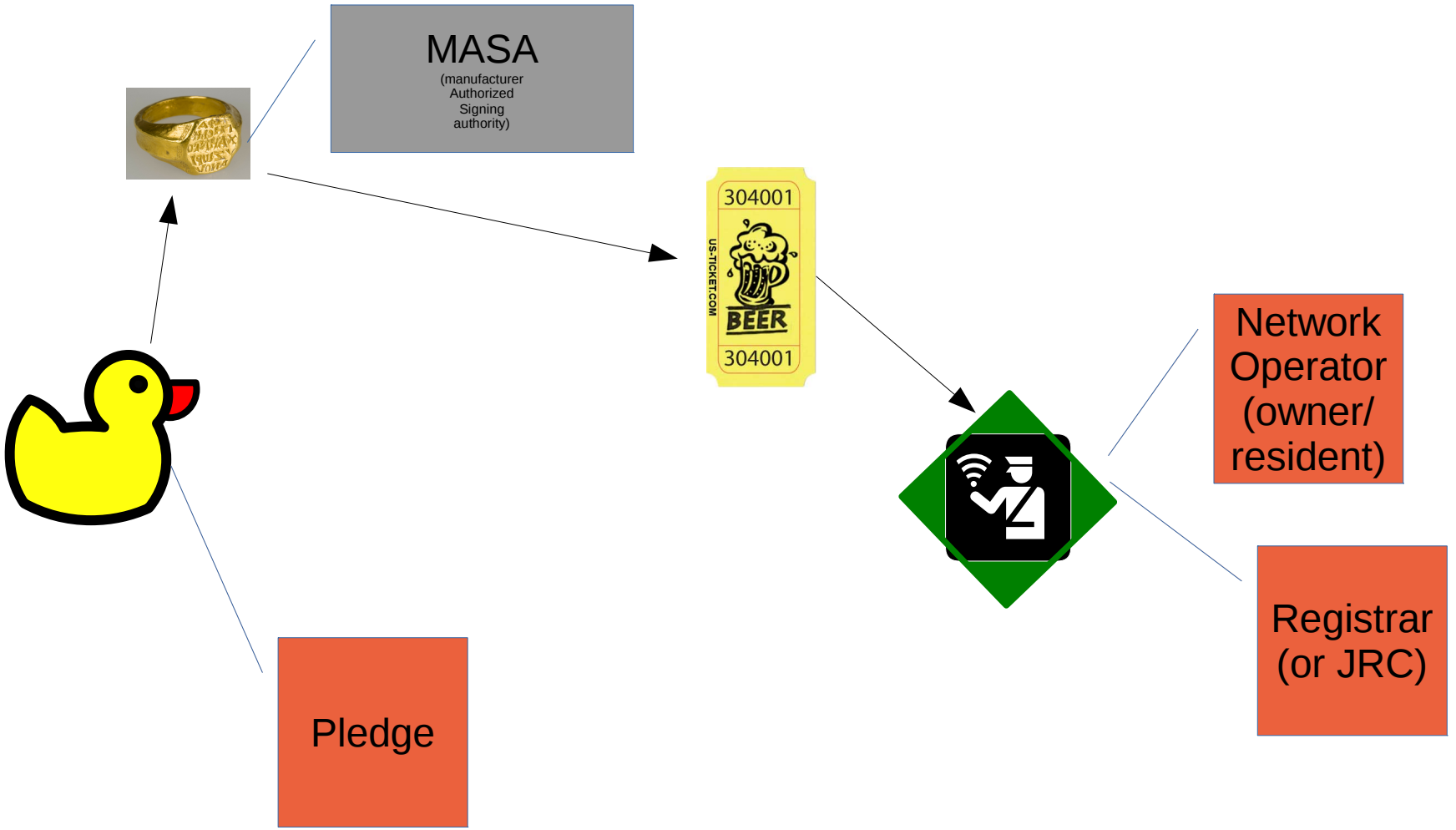
BRSKI in animation



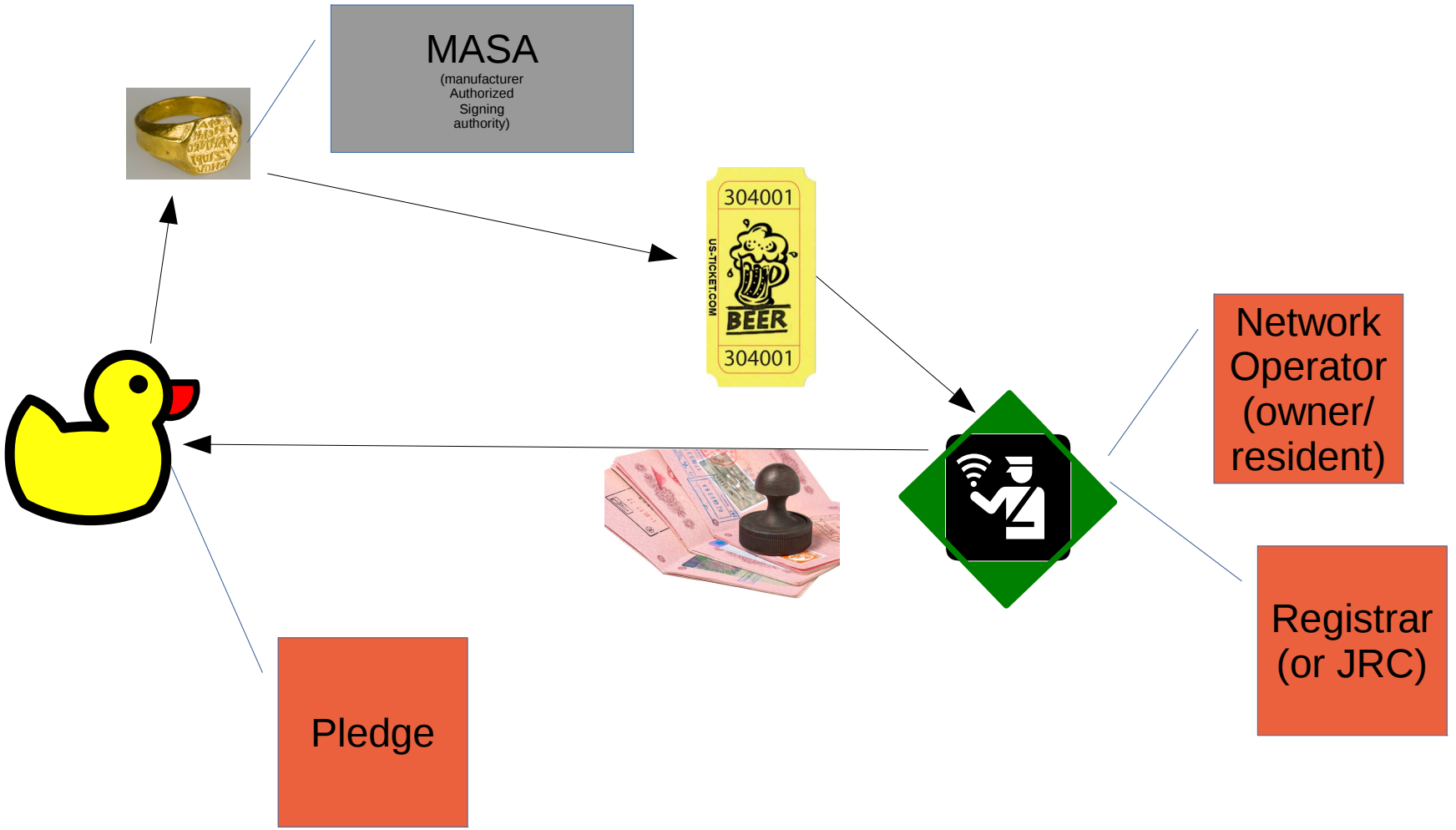
BRSKI in animation



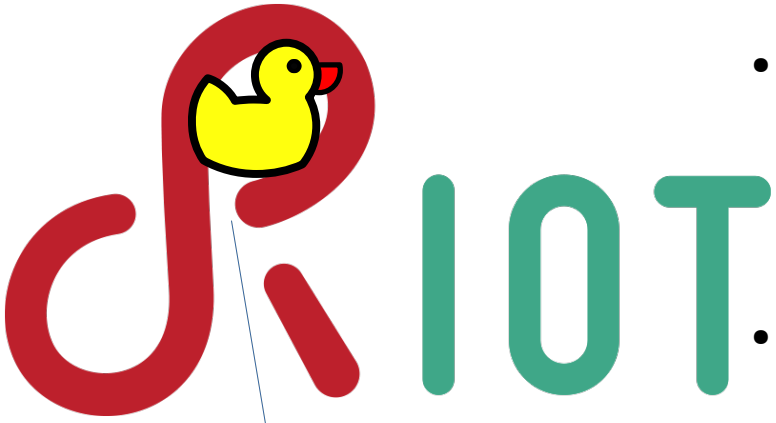
BRSKI in animation



BRSKI in animation



Goal: Pledge Library for RIOT-OS



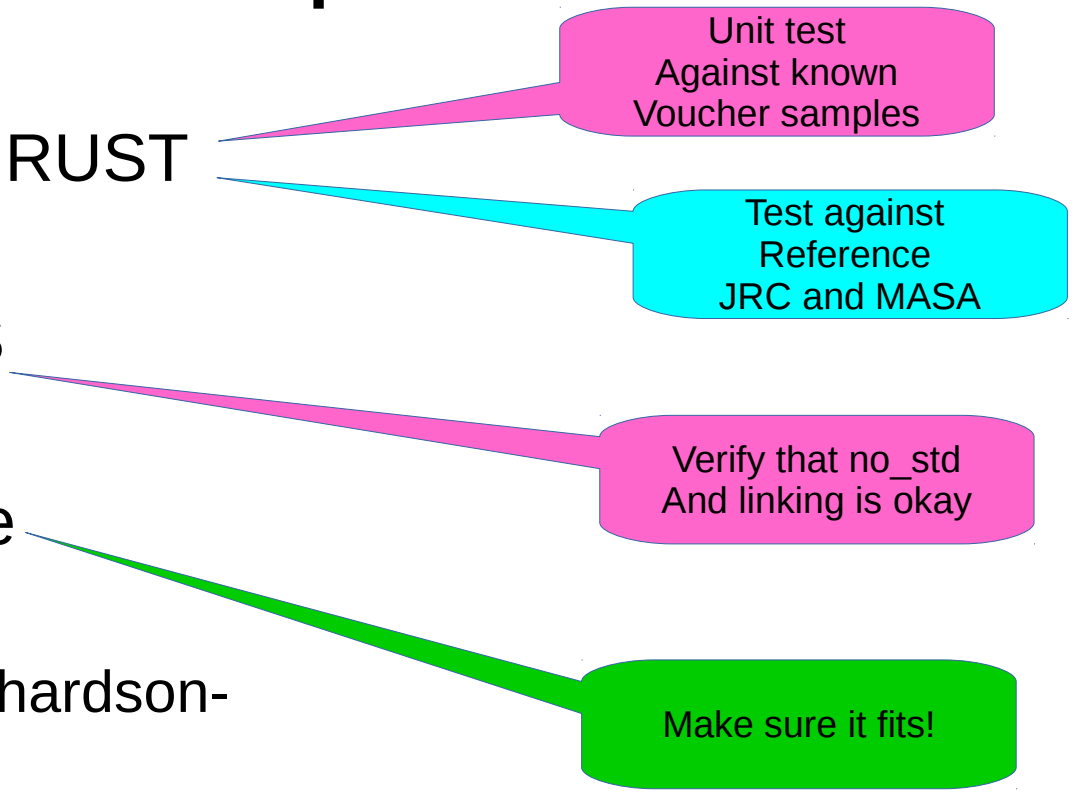
- Available for all devices, networks (wifi, 802.15.4)
  - Should include proxy functionality.
- Sharing common core code with “bigger” platforms, e.g. Android IoT, OpenWRT, generic Linux
- Written in a memory-safe language.
- Leveraging as many native libraries as possible
- In as few bytes of code as possible

I have done C-code since I was 14, 34 years ago. For first 5 years, on machines without MMUs, where NULL deference was “safe”: enough!

Unrelated humour about Ducks and ping(8)
https://www.sandelman.ca/mcr/humour/storyofping.html

Work plan

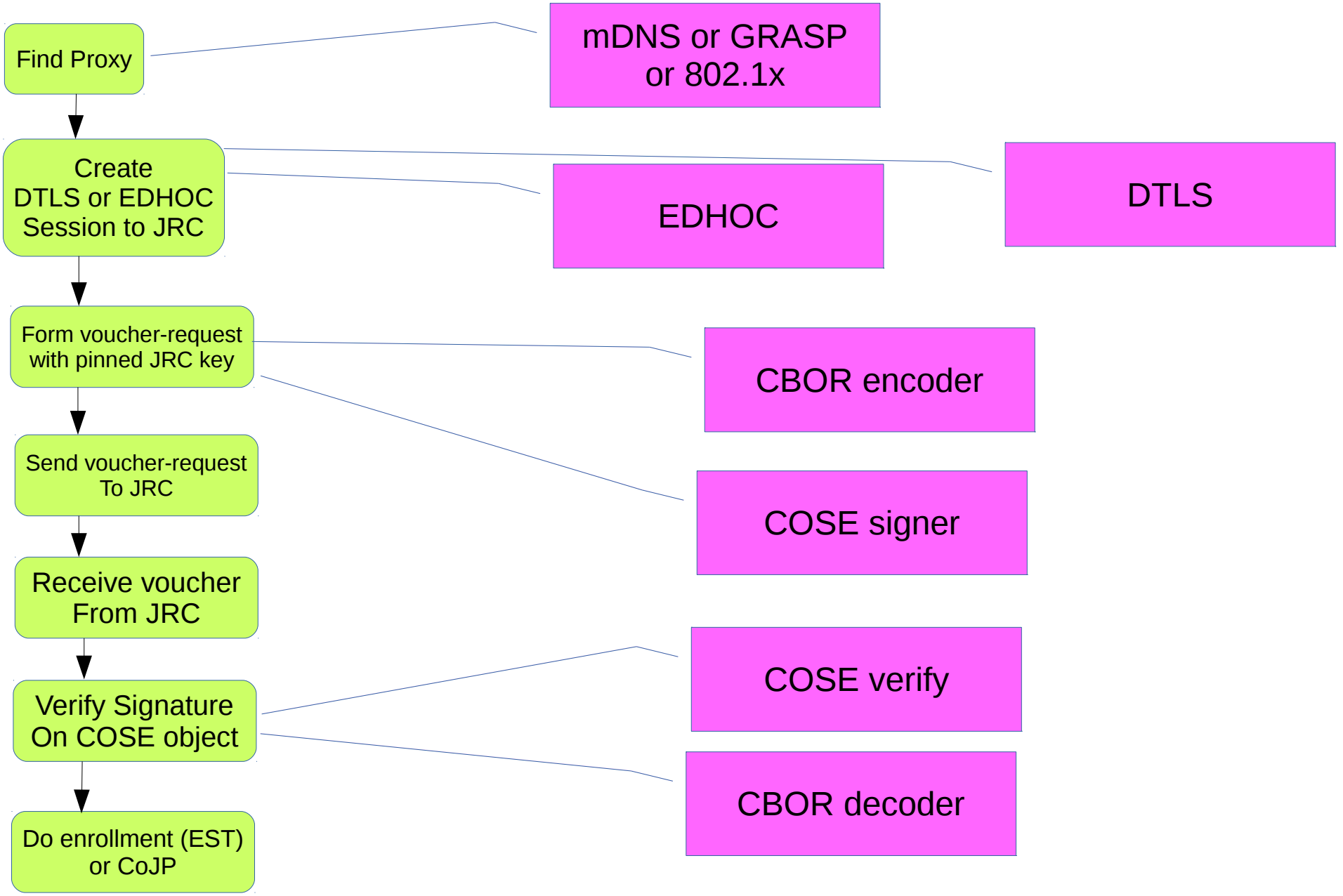
- Build client library in RUST on Linux
- Compile to RIOT-OS “native” regularly
- Target ESP32 device periodically
  - WiFi initially, draft-richardson-anima-smarkaklink
  - 802.15.4, draft-ietf-6tisch-dtsecurity-zerotouch later (requires proxy code too)



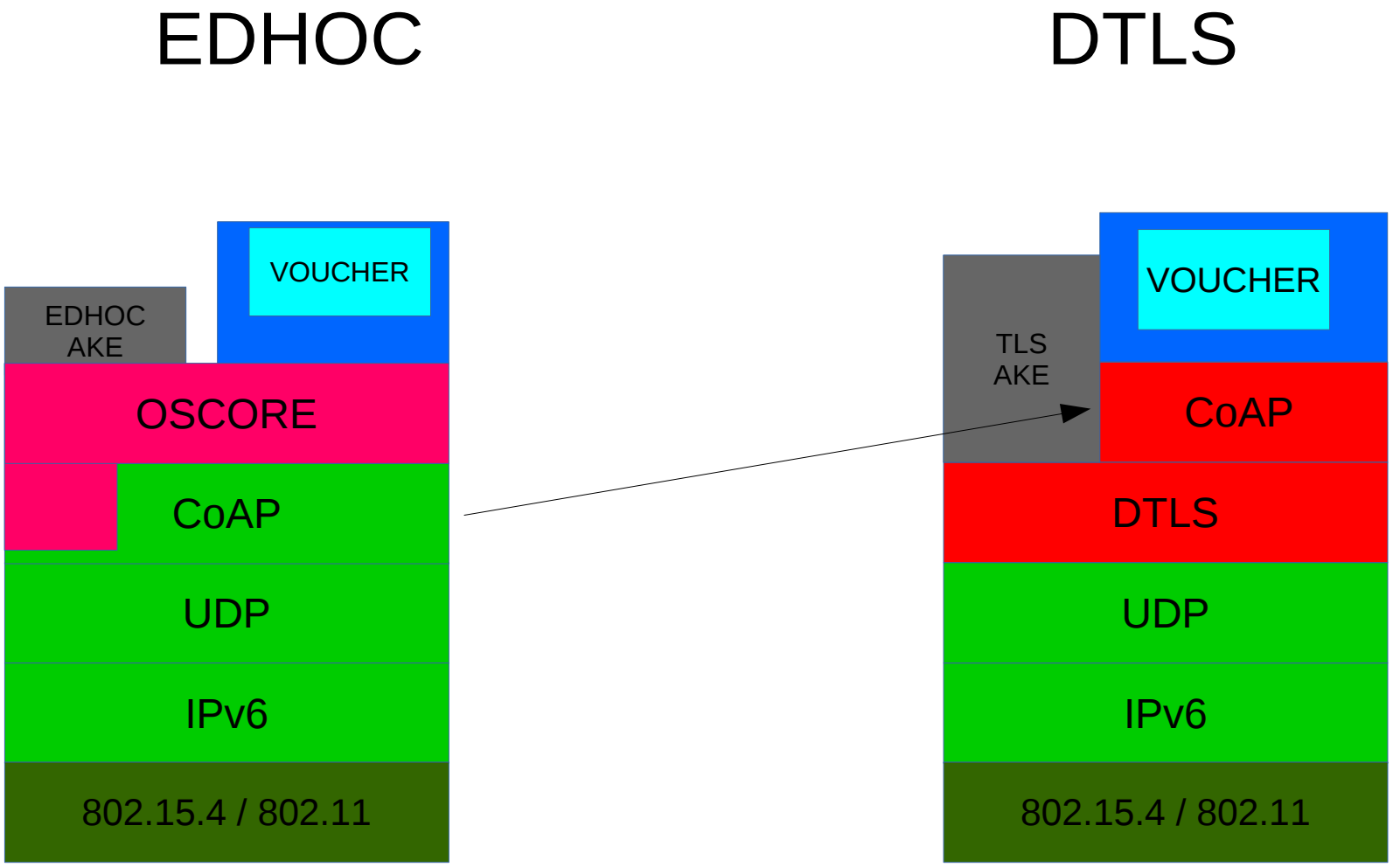
Reality!

- Still a very early work in progress
  - Many decisions to make, which I'll share
- Not the walk in the park I had hoped for.
  - Little of this is RIOT-OS, though!

Anatomy of BRSKI pledge



Network differences: DTLS/EDHOC



CBOR library

RUST options

- cbor_no_std
- tokio_serde_cbor
- serde_cbor
- cbor_event(*)

RIOT-OS options

- cn-cbor
- Nanocbor (*)
- tinycbor

COSE library

RUST options

- Crate COSE

RIOT-OS options

- Libcose
  - Requires `nanocbor`
  - Either `HACL-C`, `libsodium` or `mbed TLS` as crypto library

DTLS library

- tinydtls
- ~~RUST-openssl~~
- RUST wrapped mbedtls
  - *“Additionally, building on MbedTLS's focus on embedded use, this crate can be used in a no_std environment”*

- RIOT-OS *mbedtls* – not yet ported, AFAIK.

EDHOC library

- needs cose library!
- plus some crypto libraries

Would really prefer to write this in RUST.

OSCORE library

Not even sure!

Stack Options for EDHOC

More RUST

- cbor_event
- crate cose
- tweetnacl (RUST-wrapped C)

Less RUST

- nanocbor
- libcose
- tweetnacl

Stack Options for DTLS

More RUST

- cbor_event
- crate cose
- RUST-wrapped mbedtls

Less RUST

- nanocbor
- libcose
- tinydtls
  - need to check crypto requirements for this.

Conclusions and next steps

- the pkg system is nice.
- previous versions had more code “inline”, which makes comprehension and “grep -R” easier
  - less flexibility reduces choices
- *EDHOC still in IETF limbo until Oct.2019*

Next Steps

1) focus on RUST voucher library
2) get some kind of DTLS interaction with native and ESP32 devices
3) figure out OSCORE pieces needed for EDHOC

Questions!!!!

This presentation

https://www.sandelman.ca/SSW/talks/riotos2019-brski/html

BRSKI and ANIMA

http://datatracker.ietf.org/wg/anima

https://datatracker.ietf.org/doc/draft-ietf-anima-bootstrapping-keyinfra/

Minerva reference implementation (MASA, JRC, Pledge in Rails)

https://minerva.sandelman.ca/