Funded by the European Union under Grant Agreement No. 101087529. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Research Executive Agency. Neither the European Union nor the granting authority can be held responsible for them.

Co-funded by the European Union

Migrating Some Legacy e-Governance Applications to Post-Quantum Cryptography

Cyber-security Excellence Hub in Estonia and South Moravia

NIST - Fifth PQC Standardization Conference

April 10, 2024

Rockville, MD

Petr Muzikant, Jan Willemson, Peeter Laud

Information Security Research Institute @ Cybernetica AS, Estonia

Presentation Outline

- **Introduction**
- **Our Methodology**
- **e-Governance Applications**
- **PQ Engineering Obstacles**
- **Conclusions**

Presentation includes hypertext links.
PDF available at https://csrc.nist.gov/Presentations/2024/migrating-legacy-e-governance-applications-to-pqc

Introduction

- **Standardization of PQC** = first step in the process of actual deployment
- **Existing work:**
  - Timelines, "Migration Challenges", PQC Migration Handbook
- **The Issue:**
  - How to actually migrate? Does a general security engineer have everything in his disposal for that?
- **Our work:**
  - Exploring and supporting current FOSS state-of-the-art
  - Focus on engineering aspects of PQ implementations
  - Gather experience, problems, and remarks

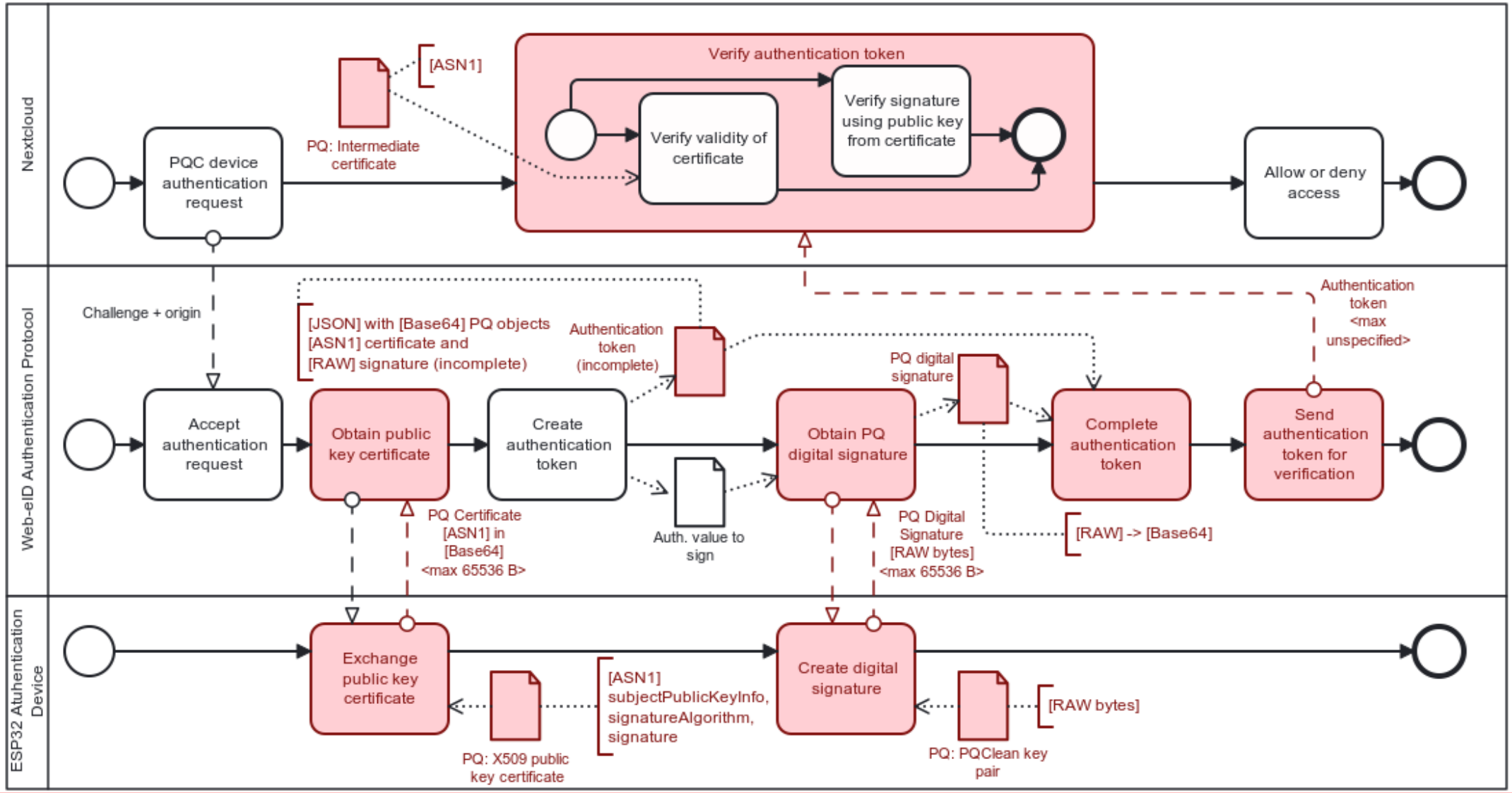
Our Methodology

Identify, Prepare, Implement, Adapt

Our Methodology

- **Identify**
  - All PKI objects and their lifetime in the system
- **Prepare**
  - MTUs, data formats (DER, PEM, JOSE, serialized, etc.)
  - Technological constraints
- **Implement**
  - Utilize available open source tools step-by-step
- **Adapt**
  - Custom wrappers, pull requests
  - Expect future changes

Our Methodology: BPMN Example

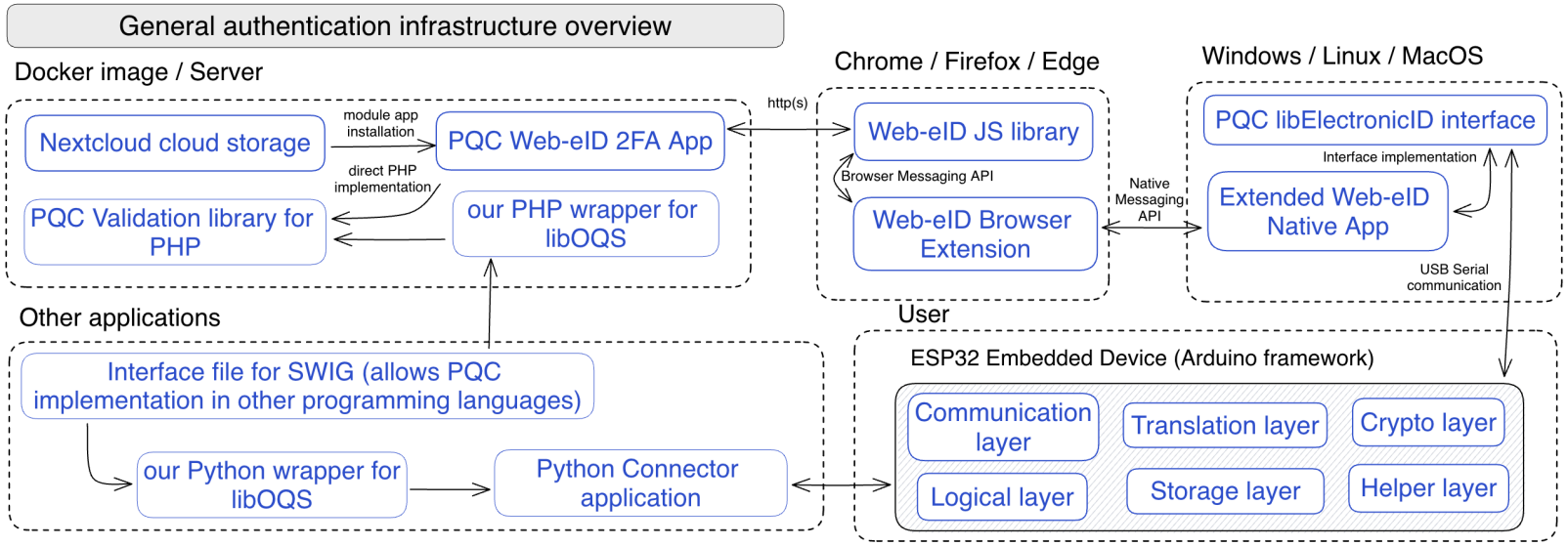


e-Governance Applications

Authentication, Encryption, Digital Signature, and e-Voting Frameworks

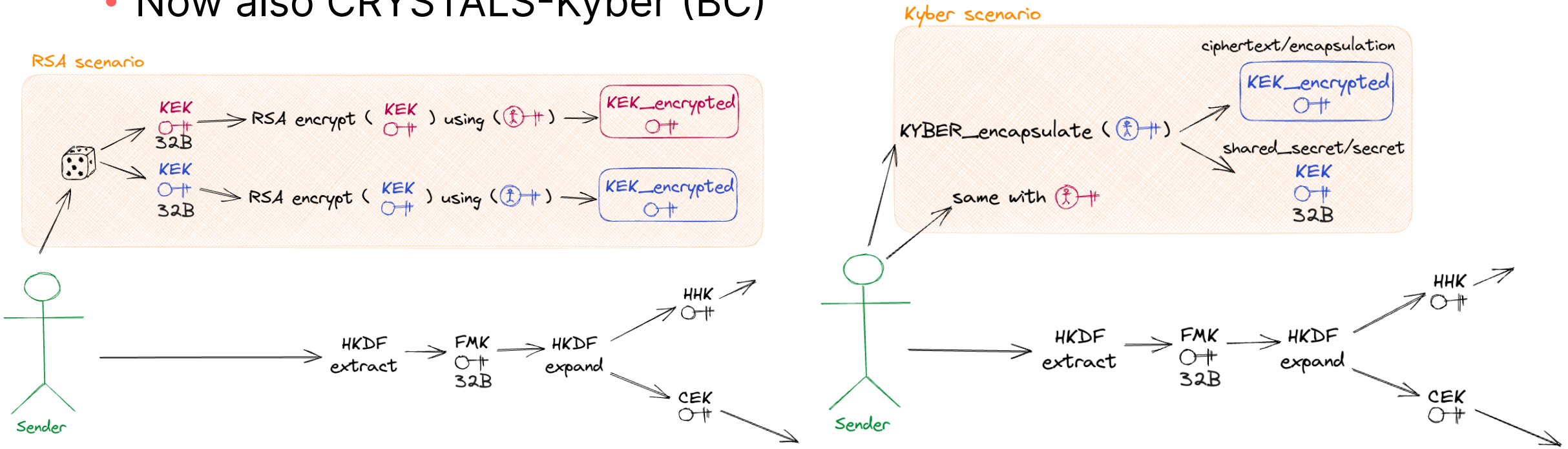
Web-eID (Authentication Framework)

- Authenticate users to **web application via browser interface**
- Web service + authentication framework + authentication device
  - Nextcloud file server
  - ID cards → ESP 32 embedded device (Dilithium5 or Falcon1024)



CDOC2 (Encryption Framework)

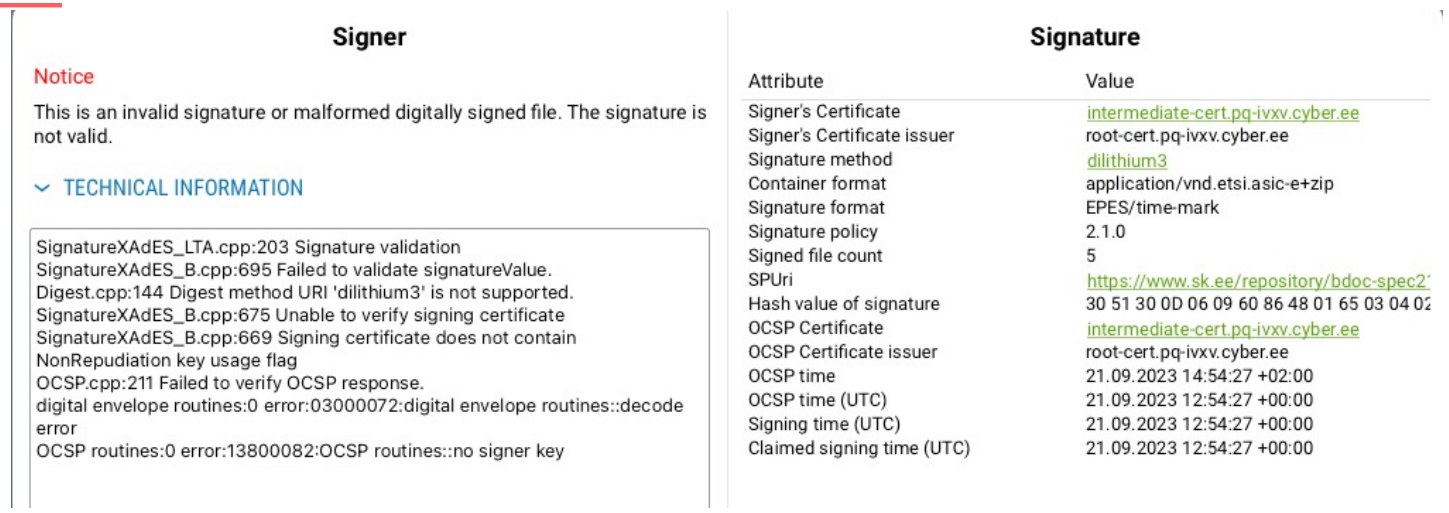
- = specification defining the process of **securing**, **storing**, and **exchanging encrypted messages** (.cdoc files)
- 1-to-many, expects RSA, EC or symmetric keys
  - Now also CRYSTALS-Kyber (BC)



CYBERNETICA

ASiC-E (Digital Signature Framework)

- = file and structure (container) specification **binding one or more digitally signed objects together** (.asice files) under eIDAS regulation
  - ETSI TS 102 918 v1.2.1
- Long-term availability and integrity of files within the container
- PQ version of thorgate/pyasice
  - Custom wrapper of libOQS
  - CMD app
  - Container creation



Signer

Notice

This is an invalid signature or malformed digitally signed file. The signature is not valid.

TECHNICAL INFORMATION

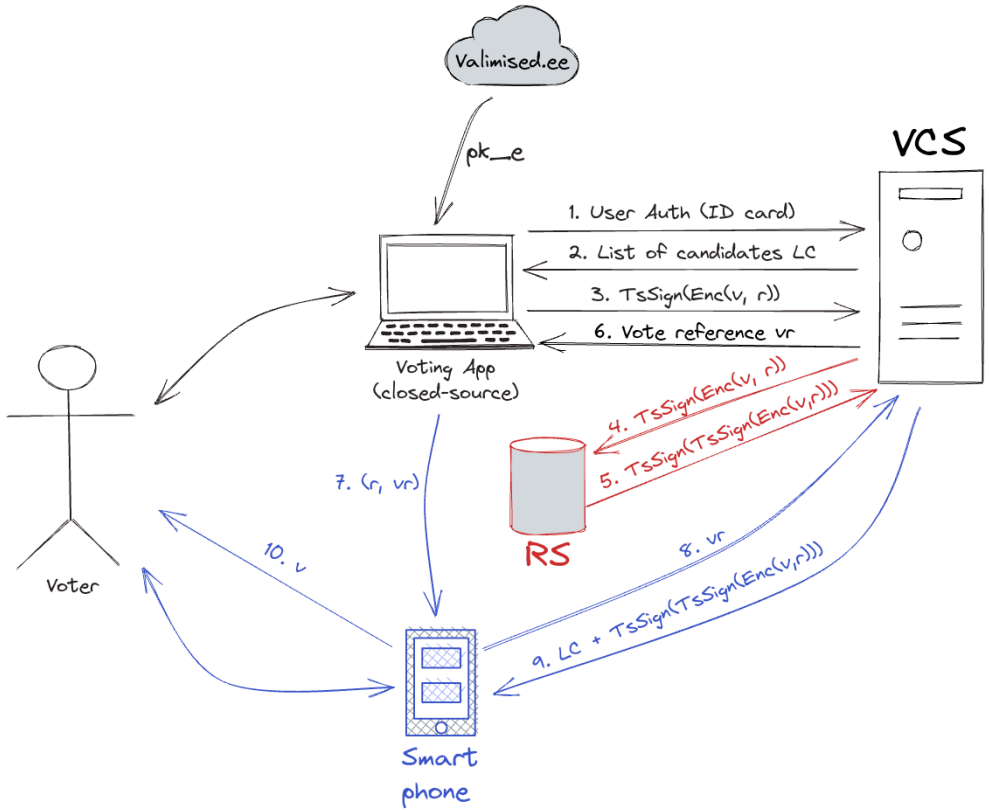
```
SignatureXAdES_LTA.cpp:203 Signature validation
SignatureXAdES_B.cpp:695 Failed to validate signatureValue.
Digest.cpp:144 Digest method URI 'dilithium3' is not supported.
SignatureXAdES_B.cpp:675 Unable to verify signing certificate
SignatureXAdES_B.cpp:669 Signing certificate does not contain
NonRepudiation key usage flag
OCSP.cpp:211 Failed to verify OCSP response.
digital envelope routines:0 error:03000072:digital envelope routines::decode
error
OCSP routines:0 error:13800082:OCSP routines::no signer key
```

Signature

Attribute	Value
Signer's Certificate	intermediate-cert.pq-ivxv.cyber.ee
Signer's Certificate issuer	root-cert.pq-ivxv.cyber.ee
Signature method	dilithium3
Container format	application/vnd.etsi.asic-e+zip
Signature format	EPES/time-mark
Signature policy	2.1.0
Signed file count	5
SPUri	https://www.sk.ee/repository/bdoc-spec2
Hash value of signature	30 51 30 0D 06 09 60 86 48 01 65 03 04 02
OCSP Certificate	intermediate-cert.pq-ivxv.cyber.ee
OCSP Certificate issuer	root-cert.pq-ivxv.cyber.ee
OCSP time	21.09.2023 14:54:27 +02:00
OCSP time (UTC)	21.09.2023 12:54:27 +00:00
Signing time (UTC)	21.09.2023 12:54:27 +00:00
Claimed signing time (UTC)	21.09.2023 12:54:27 +00:00

IVXV (e-Voting Framework)

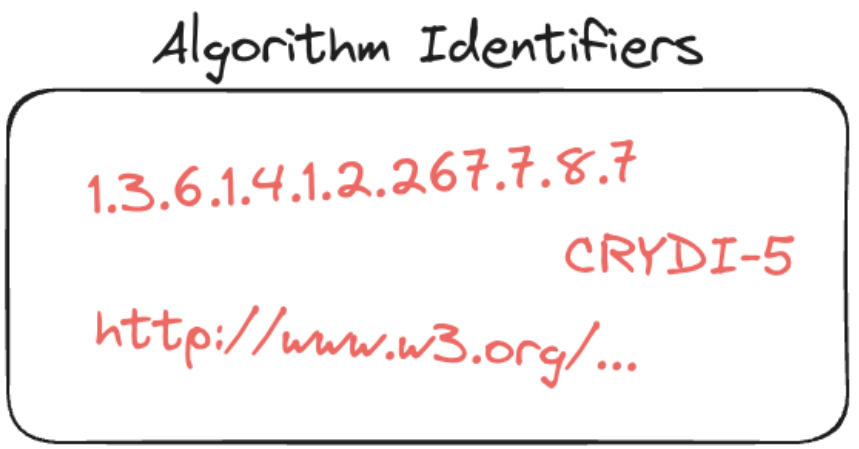
- = ongoing development of **Estonian e-Voting framework** named IVXV
- Components:
  - PQ-IVXV source distribution
  - PQ-ASiC-E
  - Vote application, Vote verification
  - Registration service TSA
  - Citizen PKI, OCSP, and TSA
- Custom lattice-based protocols
  - Vote Encryption, MixNets, MPC, ...



PQ Engineering Obstacles

Algorithm Identifiers, Object Encoding, Interoperability Awareness, Cryptographic Tokens, Maturity of PQC, and Misc.

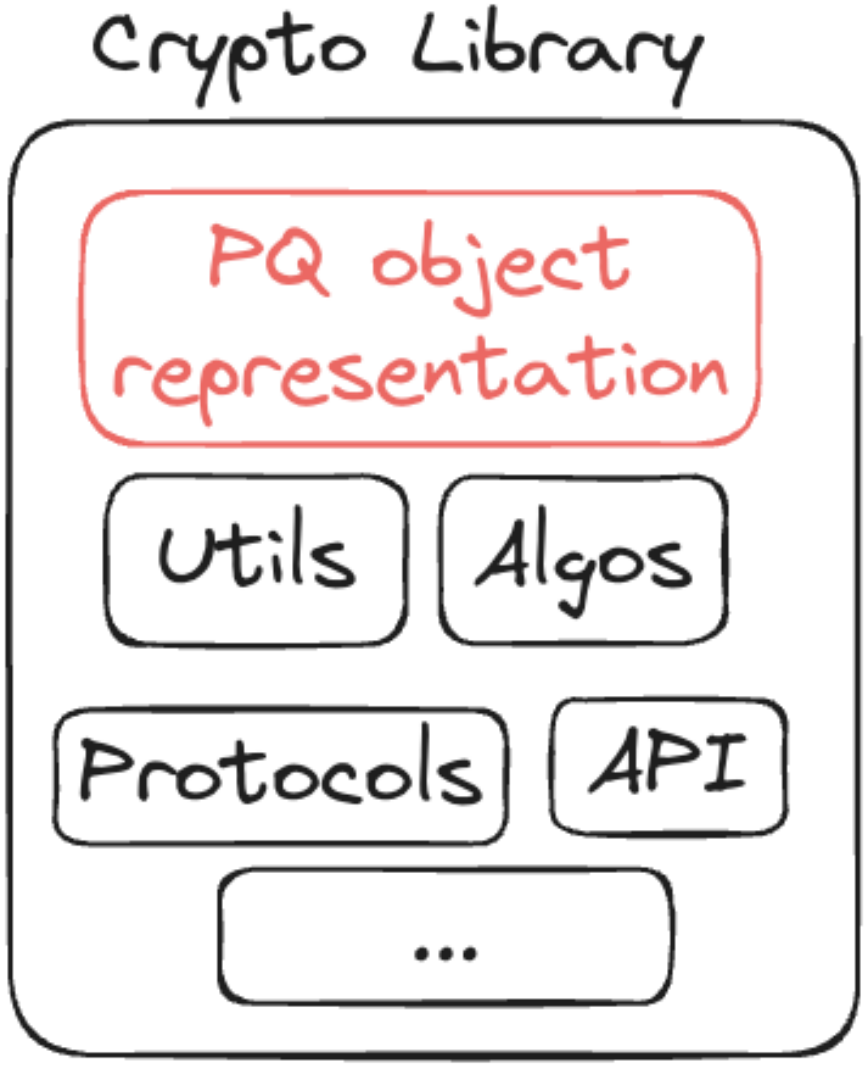
Algorithm Identifiers



- **ASN.1 Object Identifiers**
  - Wild West
  - OQS → BouncyCastle → OQS → IETF Hackathon → ???
  - ML-KEM vs CRYSTALS-Kyber?
- **JSON Web Algorithms**
  - PQ alternative to *ES256*?
  - Recent RFC, but only for KEMs
- **XML Signature Syntax Algorithms**
  - Defined URLs for digital signature algos
  - E.g.: http://www.w3.org/2001/04/xmldsig-more#rsa-sha256

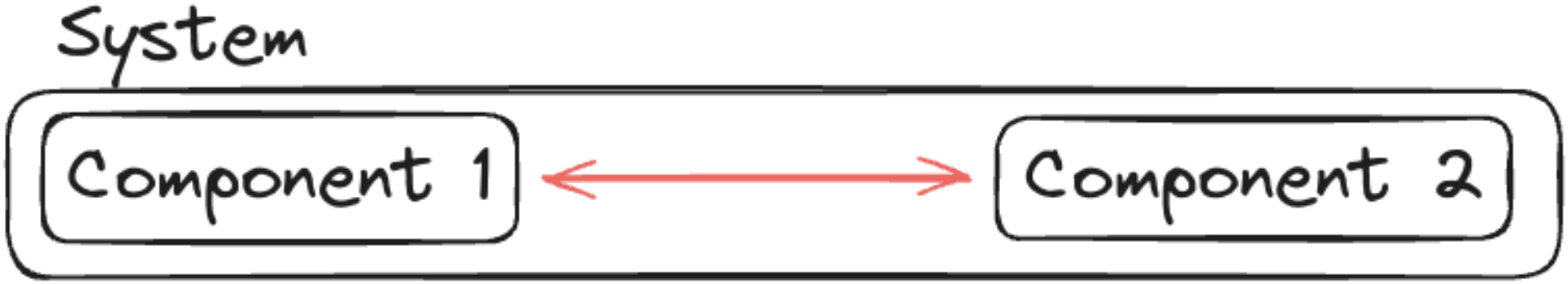
Object Encoding

- **Raw bytes** → interoperability heaven
- Already implemented **ASN.1 RFC drafts**? Why?
- Most crypto libraries have classical algorithms **hard-coded**
  - PHP extension for OpenSSL, PHPSecLib
  - cryptography, asn1crypto (Python)
  - crypto (Go)
- **Two options**:
  - **Extend** vs **circumvent**



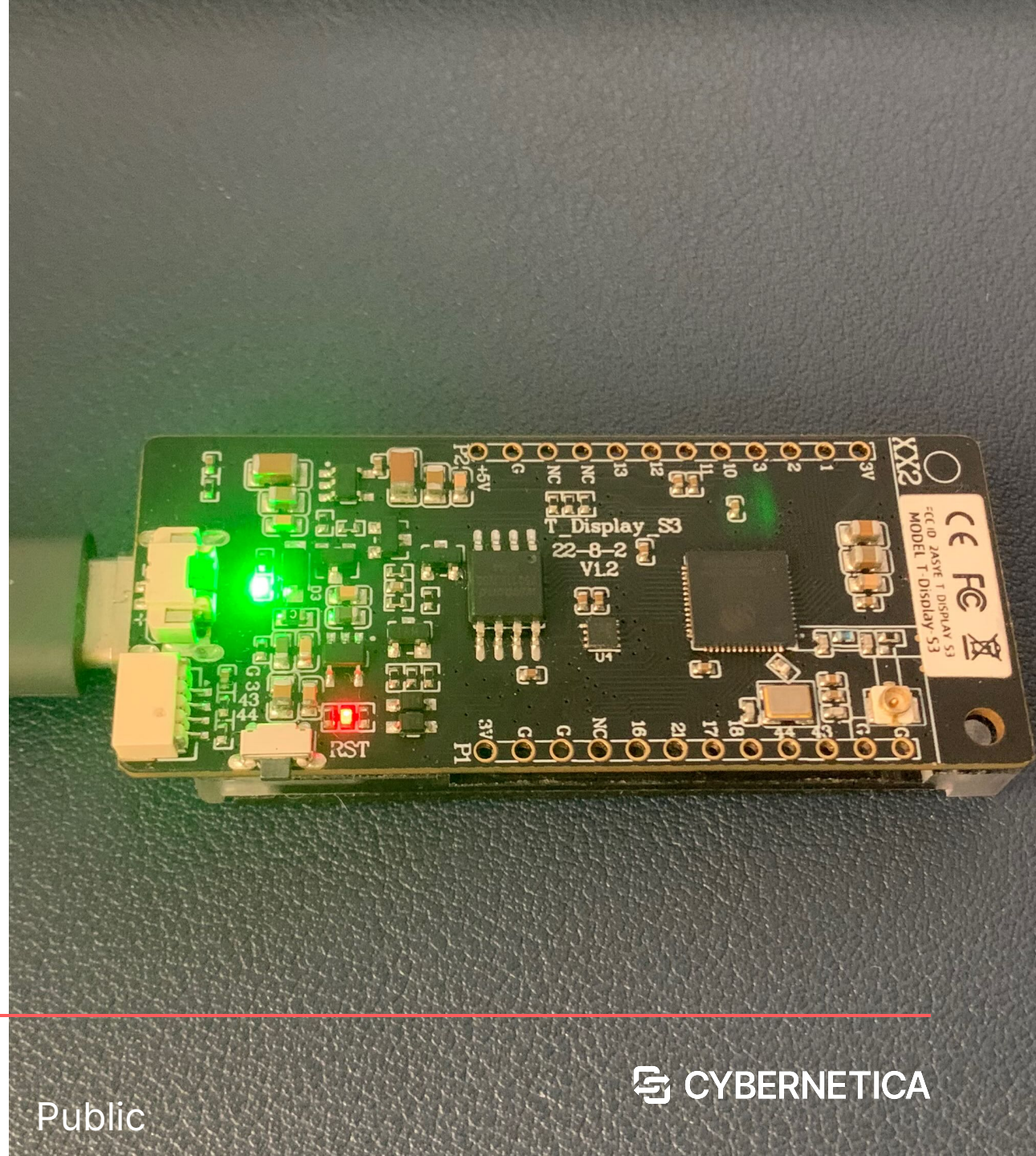
Interoperability Awareness

- Growing with system complexity
- **Active thinking** about all components
  - Identifiers, encoding, MTU, processing



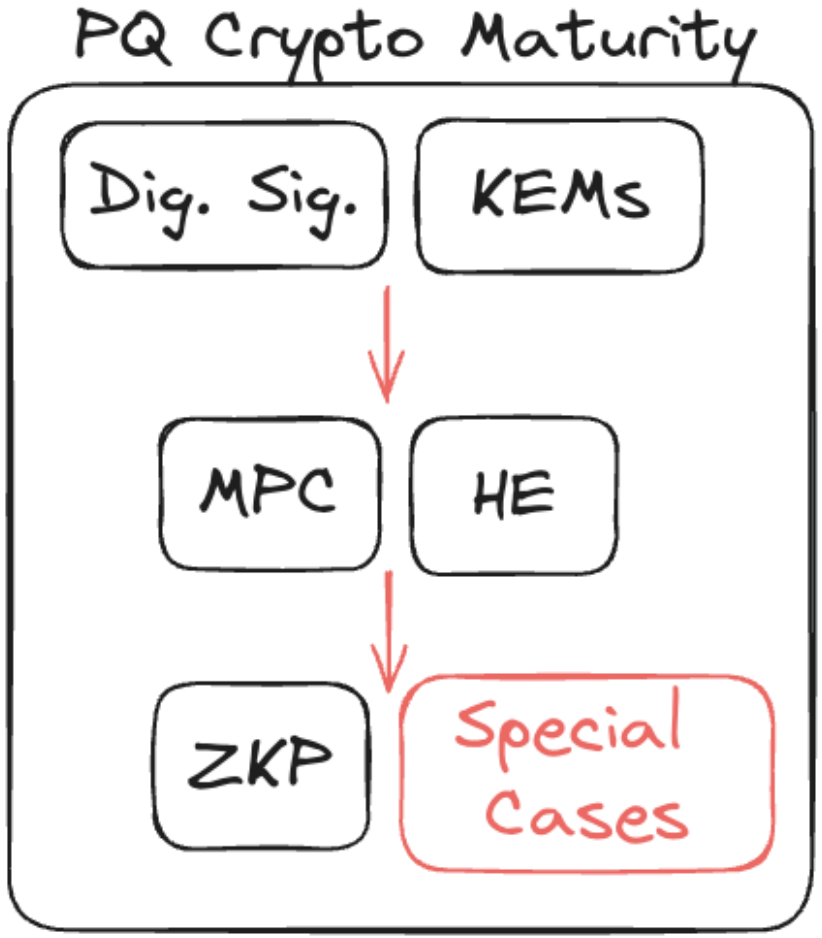
Cryptographic Tokens

- ~~Smart cards~~
- Chip manufacturers?
- Embedded devices for local testing purposes
  - Performance OK (ESP32-S3)
  - Memory OK, but complicated
  - Safety not OK (no HSM, TPM, not certifiable)
- Protocol adjustments might be required (~~stack~~ → heap, streaming)



Maturity of PQ Custom Crypto

- What if application requires:
  - **Multi Party Computation**?
  - **Homomorphic Encryption**?
  - **Special features**?
    - e.g. *ElGamal* in vote encryption - special decryption without private key
- Still lot of **R&D to be done**



Miscellaneous PQ Engineering Efforts

- **All the little things**
  - (OQS-)OpenSSL encodes private keys as:
    - *0×04 or 0×03 || length || private_key || public_key*
  - Custom wrappers → data type conversions
  - Adding single lines into dependencies' files to support PQ
  - Build issues, insufficient or confusing documentation
- **State of PQ-TLS server and client implementations**
- **Hybrid Mode**

Conclusions

- **Experience report**
  - Multiple e-Government projects of various sizes
  - PQC implementation is far from straight-forward
  - Engineering issues that need collective resolution
- **Standardization is crucial**, but considering the general advice is to **start implementing now**:
  - Brace yourself (or your engineers/developers)
  - Document
  - Share
  - Contribute

Thank you for listening!

Resources:
- links in presentation slides
- references in the paper
- Notes on PQC in PHP

Petr Muzikant, petr.muzikant@cyber.ee
Jan Willemson, jan.willemson@cyber.ee
Peeter Laud, peeter.laud@cyber.ee

https://cyber.ee/
info@cyber.ee
cybernetica
CyberneticaAS
cybernetica_ee
Cybernetica