# Post-Quantum Cryptography in VC
## PQ and PQ/T hybrid approaches

Andrea Vesco, Ph.D.

Head of Cybersecurity Research @ LINKS Foundation

QUBIP project coordinator

andrea.vesco@linksfoundation.com

FONDAZIONE links
PASSION FOR INNOVATION

QUBIP
Quantum-oriented Update to Browsers and Infrastructure for the PQ Transition

# Quantum computing: 2 sides of the same moon

- The quantum computer is a major revolution, paving the way for new discoveries in a wide range of disciplines.
- One small problem: the Cryptographically Relevant Quantum Computer (CRQC) will break most of the cryptography used to secure the Internet today. Asymmetric cryptography is at risk.

# Quantum threat mitigation: PQC

1. Cryptographers: PQC a new class of algorithms based on a set of hard problems which should be immune to quantum threats.
2. Engineers: integration of PQ algorithms in protocols, networks, and systems while ensuring crypto-agility, pliability and possibly using hybrid schemes.
    - Crypto-agility: ability to reconfigure software by plugging in PQC algorithms without extensive code refactoring.
    - Pliability: transition must be aligned with network management best practices, supports established network services, and be integrated through standardized approaches.
    - PQ/T hybrid schemes: combine both PQ and traditional algorithms in a single crypto scheme. It is secure as long as the the security of one of the algorithms holds.

# PQC Algorithms: key agreement and signatures

## Federal Register Notices

August 13, 2024 — The Secretary of Commerce approved three Federal Information Processing Standards (FIPS) for post-quantum cryptography:

- FIPS 203, Module-Lattice-Based Key-Encapsulation Mechanism Standard
- FIPS 204, Module-Lattice-Based Digital Signature Standard
- FIPS 205, Stateless Hash-Based Digital Signature Standard

draft FIPS 206 standard built around FALCON

| Code-Based | Lattice-Based | MPC-in-the-Head | Multivariate |
|---|---|---|---|
| **CROSS** | EagleSign | Biscuit | 3WISE |
| Enhanced pqsigRM | EHTv4 | **MIRA*** | DME-Sign |
| FuLeeca | HAETAE | **MiRitH*** | HPPC |
| **LESS** | **HAWK** | **MQOM** | **MAYO** |
| MEDS | HuFu | **PERK** | PROV |
| WAVE | Raccoon | **RYDE** | **QR-UOV** |
| | SQUIRRELS | **SDitH** | **SNOVA** |
| | | | TUOV |
| | | | **UOV** |
| | | | VOX |

| Other | Symmetric-Based | Isogeny-Based |
|---|---|---|
| ALTEQ | AIMer | **SQIsign** |
| eMLE-Sig 2.0 | Ascon-Sign | |
| KAZ-SIGN | **FAEST** | |
| PREON | SPHINCS-alpha | |
| Xifrat1-Sign.I | | |

# Transition to PQC

- Store-Now-Decrypt-Later: data encrypted today can be decrypted by a quantum computer in the future.
- Key agreement (urgent) prioritized over signatures for authentication (before CRQC).
- Transition takes time:
  - Very diverse: many different implementation with performance constraints and update cycles.
  - Protocol ossification: despite being designed to be upgradeable, any flexibility that isn't used in practice, is probably broken, because of faulty implementations [*by Bas Westerbaan – Cloudflare*].

# CRQC and decentralized identity

- Quantum computers threaten traditional asymmetric cryptography mainly through the Shor's algorithm for factoring integers and solving discrete logarithms and Grover's search.
- The main cryptographic algorithms in VCs are the digital signatures.
- Consequently, VCs are vulnerable to Shor's algorithm, but not to the store-now-decrypt-later attacks since key agreement for encryption is not involved in any interaction among Issuers, Holders, and Verifiers.
- Therefore, the transition of decentralized identity to PQC means switching from traditional to PQ identity key pair generation and digital signature for VCs and VPs.

# Pure PQ approach

Plaintext VC

# Two assumptions

- The transition to PQC involves the selection of the appropriate PQ signature algorithm to be used by the Issuer and Holder to sign the VC and VP respectively, and by the Verifier to verify the signatures for authentication purposes.
- In a target web scenario, a Holder will request access to a Verifier with a higher frequency than the frequency with which he/she requests the issuance of a VC to the Issuer.

# The requirements

Given a NIST security level, the requirements in order of priority are:

1. the signature verification time must be as short as possible to increase the number of Holders a Verifier can authenticate in a unit of time. A Verifier must verify the signature on the VC and on the VP;
2. the signature generation time must be as short as possible to (*i*) increase the number of VCs an Issuer can issue in a unit of time, and to (*ii*) decrease the time a Holder spends preparing the VP;
3. the signature size should be as small as possible to reduce the overall size of the VC and VP;
4. the public key size should be as small as possible to reduce the size of DID documents and of DIDs in case of DLT-less methods (*e.g.*, did:jwk).

| Algorithm | NIST Security Level | ④ JWK(Public Key) Size (Bytes) | ③ Sign Size (Bytes) | Key Pair Generate (ms) | ② Sign Generate (ms) | ① Sign Verify (ms) |
|---|---|---|---|---|---|---|
| Ed25519 | - | 145 | 64 | 0,042 | 0,042 | 0,070 |
| FALCON 512 | 1 | 1291 | 752 | 13,361 | 0,685 | 0,086 |
| SLH-DSA-SHA2-128f | 1 | 147 | 17088 | 2,745 | 61,541 | 3,686 |
| SLH-DSA-SHA2-128s | 1 | 147 | 7856 | 167,310 | 1263,700 | 1,331 |
| SLH-DSA-SHAKE-128f | 1 | 148 | 17088 | 3,802 | 86,079 | 5,023 |
| SLH-DSA-SHAKE-128s | 1 | 148 | 7856 | 238,820 | 1807,600 | 1,848 |
| ML-DSA-44 | 2 | 1845 | 2420 | 0,252 | 0,694 | 0,151 |
| ML-DSA-65 | 3 | 2698 | 3293 | 0,373 | 1,028 | 0,241 |
| SLH-DSA-SHA2-192f | 3 | 168 | 35664 | 3,892 | 100,110 | 5,375 |
| SLH-DSA-SHA2-192s | 3 | 168 | 16224 | 241,470 | 2250,100 | 1,925 |
| SLH-DSA-SHAKE-192f | 3 | 169 | 35664 | 5,575 | 141,390 | 7,815 |
| SLH-DSA-SHAKE-192s | 3 | 169 | 16224 | 348,670 | 3124,700 | 2,613 |
| FALCON 1024 | 5 | 2487 | 1462 | 40,792 | 1,140 | 0,168 |
| ML-DSA-87 | 5 | 3551 | 4595 | 0,503 | 1,263 | 0,405 |
| SLH-DSA-SHA2-256f | 5 | 190 | 49856 | 10,001 | 204,800 | 5,490 |
| SLH-DSA-SHA2-256s | 5 | 190 | 29792 | 159,610 | 1990,700 | 2,796 |
| SLH-DSA-SHAKE-256f | 5 | 191 | 49856 | 14,513 | 291,390 | 7,824 |
| SLH-DSA-SHAKE-256s | 5 | 191 | 29792 | 230,200 | 2766,500 | 3,815 |

# Selection of algorithms for PQ VCs / VPs

- ML-DSA-44 sl-2 equivalent to SHA-256/SHA3-256 collision search
- ML-DSA-65 sl-3 equivalent to AES192 key search
- ML-DSA-87 sl-5 equivalent to AES256 key search

# PQ/T hybrid approach

Plaintext VC

# The reason behind PQ/T hybrid

- Although the PQ signature algorithms standardized by the NIST have undergone rigorous reviews in recent years, they are not as much mature as the traditional algorithms.
- Hybrid schemes:
  - combine both PQ and traditional signature algorithms in a single cryptographic scheme;
  - are secure as long as the security of one of the algorithms holds;
  - hybrid authentication: authentication is achieved by the hybrid signature scheme, provided that at least one signature algorithm remains secure; an adversary must violate both signatures to forge a credential and impersonate another user's identity.
- PQ/T Hybrid in VC: if the PQ assumption is found to be flawed in the future, the composition of the signature algorithms ensures that the scheme is as secure as the traditional signature algorithm (option for the transition period)

# Hybrid scheme design

- There are several ways to combine algorithms to build a hybrid scheme.
- Hybrid signature scheme combining PQ and traditional signatures using the concatenation combiner to achieve the Weak Non-Separability (WNS) property.
- Non-Separability property defined for hybrid signatures prevents an adversary from removing the signature generated by the secure algorithms, forcing the Verifier to rely only on the signature generated by the broken algorithm (stripping attack).
- WNS property implies that an adversary cannot remove one of the signatures without the Verifier noticing.
- Through the adoption of an artifact, the evidence of the will of Issuers and Holders to hybridize their signature on VC and VP respectively.
- Our design places the artefact at the protocol level within the DID document.

**Composite ML-DSA For use in X.509 Public Key Infrastructure and CMS**
draft-ietf-lamps-pq-composite-sigs-03

# `compositeJwk` object

```
"id": "did:method_name:method_specific_id",
"authentication": [{
  "id": "did:method_name:method_spec_id#keys-1",
  "controller": "did:method_name:method_spec_id",
  // the new type in the verification method
  "type": "CompositeSignaturePublicKey",
  "compositeJwk": {
    "algId": "id-MLDSA44-Ed25519-SHA512",
    "pqPublicKey": { // contain a JsonWebKey
      "kty": "ML-DSA",
      "alg": "ML-DSA-44",
      "kid": ".. key thumbprint ..",
      "pub": ".. encoded public key .." },
    "traditionalPublicKey": { // contain a JsonWebKey
      crv": "Ed25519",
      "x": ".. x coordinate ..",
      "kty": "OKP",
      "kid": ".. key thumbprint .." }
}]
```

# Hybrid signature and verification

$$\sigma_h = (\sigma_{pq}, \sigma_t) \tag{1}$$

$$\begin{aligned} m' &= \text{hash}(m) \\ \sigma_{pq} &\leftarrow \text{Sign}(sk_{pq}, A_{pq}, \text{DER}(\texttt{OID})||m') \\ \sigma_t &\leftarrow \text{Sign}(sk_t, A_t, \text{DER}(\texttt{OID})||m') \end{aligned} \tag{2}$$

$$\begin{aligned} m' = \text{hash}(m) \\ \text{Verify}(pk_{pq}, \text{DER}(\texttt{OID})||m', \sigma_{pq}, A_{pq}) \\ \text{Verify}(pk_t, \text{DER}(\texttt{OID})||m', \sigma_t, A_t) \end{aligned} \tag{3}$$

https://github.com/Cybersecurity-LINKS/did-compositejwk/blob/main/spec.md

# did:compositejwk method

did-compositejwk / spec.md

andreavesco Update spec.md 46ec09d · 2 months ago History

Preview Code Blame 119 lines (85 loc) · 3.27 KB

Raw

## DID Method Specification

`did:compositejwk` is a deterministic transformation of a `compositeJwk` into a DID Document.

### compositeJwk

The `compositeJwk` is a new Verification Material property introduced to handle Post-Quantum/Traditinal (PQ/T) hybrid keys. This object contains the PQ and traditional public keys, both JWK encoded, and the algId string representing the name of algorithms used to generate the hybrid signature.

```
"compositeJwk": {
  "algId": ".. composite key OID ..",
  "pqPublicKey": {
     ".. PQ JWK encoded key .."
  },
  "traditionalPublicKey": {
    ".. Traditional JWK encoded key .."
  }
}
```

### DID Format

```
did-compositejwk-format   := did:compositejwk:<base64url-value>
base64url-value  := [A-Za-z0-9_-]+
```

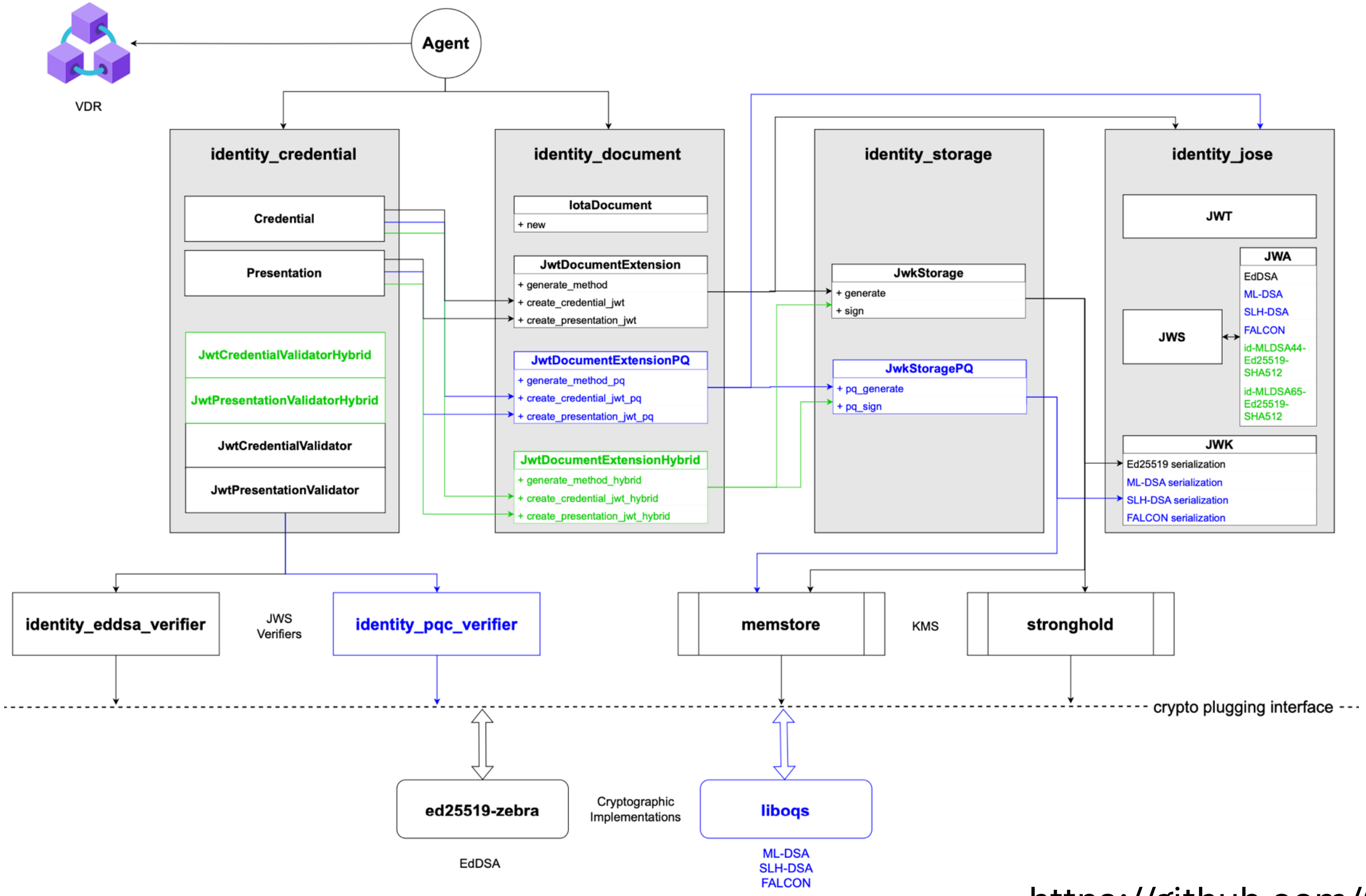
The `base64url-value` is a base64url encoded `compositeJwk`.

# Selection of algorithms for PQ/T hybrid VCs / VPs

The same requirements and principles for the selection of the PQ signature algorithms discussed for PQ apply to PQ/T hybrid case:

- id-MLDSA44-Ed25519-SHA512
- id-MLDSA65-Ed25519-SHA512

# Implementation in IOTA Identity framework




https://github.com/QUBIP/pq-identity.rs

Coherent with
key pair generation time

Coherent with
signature generation time

| Algorithm | DID Doc Size (Bytes) | DID Create (ms) | JWT(VC) Size (Bytes) | JWT(VP) Size (Bytes) | JWT(VC) Generate (ms) | JWT(VC) Verify (ms) | JWT(VP) Generate (ms) | JWT(VP) Verify (ms) |
|---|---|---|---|---|---|---|---|---|
| Ed25519 | 778 | 0,063 | 871 | 1894 | 0,091 | 0,115 | 0,100 | 0,223 |
| ML-DSA-44 | 2478 | 0,289 | 4018 | 9236 | 0,741 | 0,178 | 0,762 | 0,403 |
| ML-DSA-65 | 3331 | 0,425 | 5182 | 11952 | 1,118 | 0,268 | 1,129 | 0,607 |
| ML-DSA-87 | 4184 | 0,561 | 6918 | 16003 | 1,306 | 0,431 | 1,358 | 0,954 |
| AlgID-1 | 2766 | 0,347 | 4189 | 9621 | 0,805 | 0,275 | 0,847 | 0,582 |
| AlgID-2 | 3619 | 0,486 | 5353 | 12337 | 1,183 | 0,361 | 1,220 | 0,888 |

Coherent with
size of PQ public keys

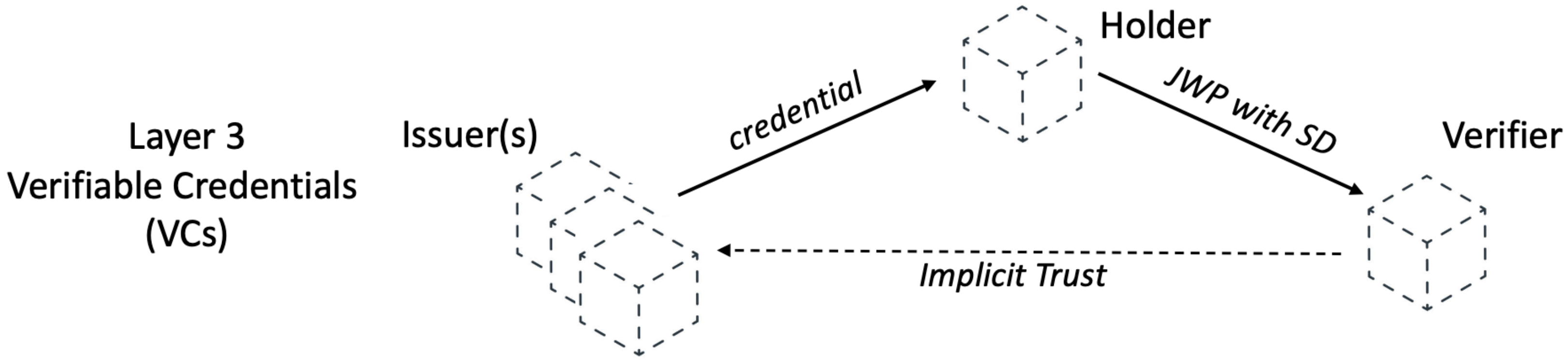
Coherent with
size of PQ signature

Coherent with
signature generation time

# Conclusions

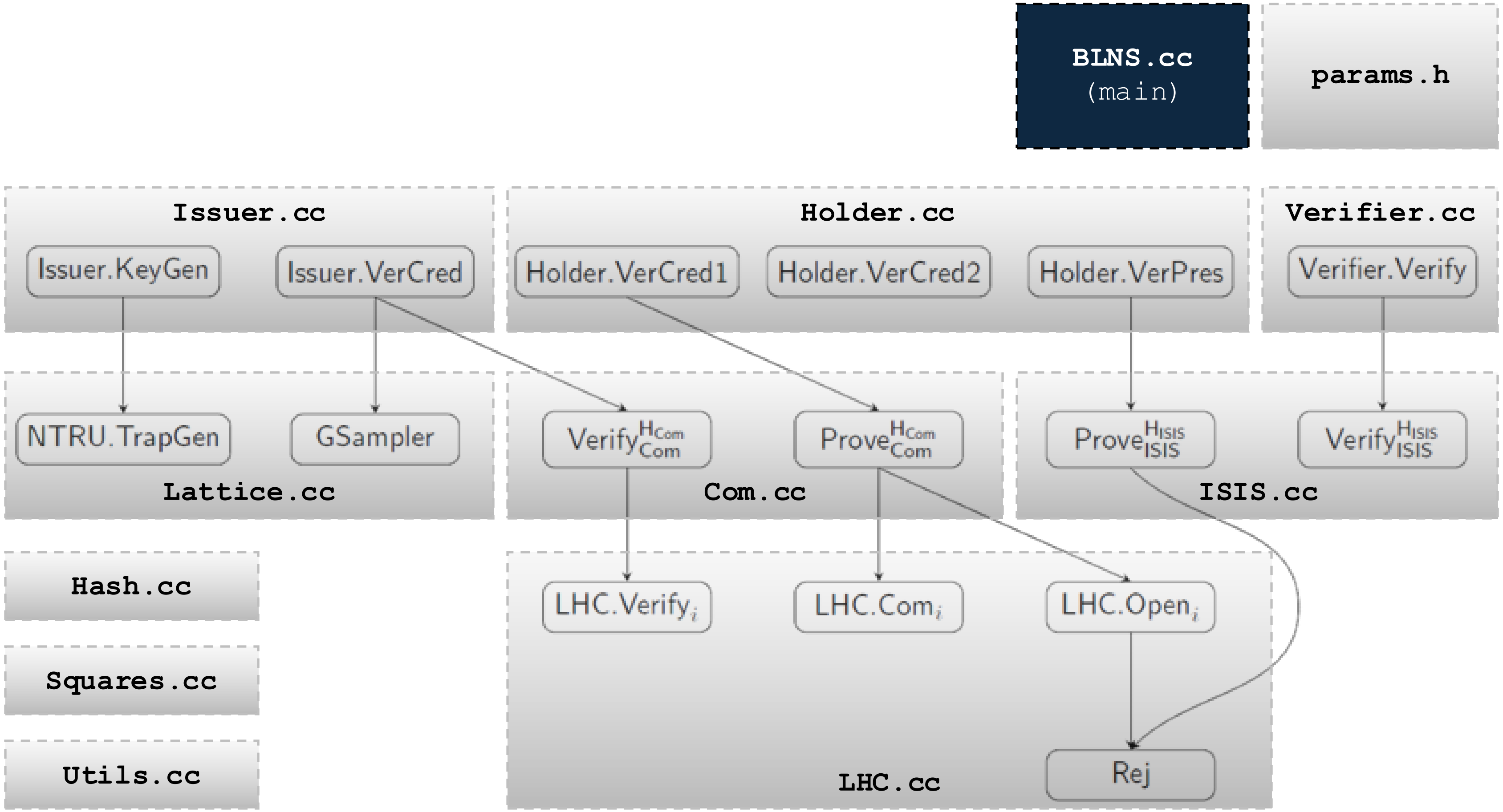
- Transition of VCs / VPs to PQC using NIST selected algorithms with a practical implementation.
- Both PQ and PQ/T hybrid approaches.
- PQ/T hybrid leverage a new `compositeJwk` object as an artifact to achieve WNS.
- PQ/T hybrid performance comparable to pure PQ performance while ensuring today's level of security in case ML-DSA (lattice in general) is found to be theoretically flawed in the future.

# Zero-Knowledge VC




- BBS+
- https://github.com/Cybersecurity-LINKS/json-proof-token
- https://github.com/Cybersecurity-LINKS/zkryptium
- adopted by 3 SSI frameworks: IOTA Identity, SpruceID, Hushmesh Inc.

# Ongoing work on PQ ZK VC with SD




A Framework for Practical Anonymous Credentials from Lattices
https://eprint.iacr.org/2023/560

# Efficient, privacy preserving, and quantum-resistant revocation