Post-Quantum Cryptography

Panel: Mike Jones and Joseph Heenan



Why post-quantum, and why the hurry

RSA, EC	**A large quantum computer breaks our public keys.** Every signature and key exchange in OAuth, OpenID Connect, JOSE and TLS. Symmetric crypto survives. NIST replacements: ML-KEM and ML-DSA (2024).
Today	**Encryption is at risk now: store now, decrypt later.** Traffic recorded today is readable once a machine exists. So hybrid TLS key exchange went first and is already on by default.
Q-Day	**Signatures must move before the machine exists.** One trusted quantum-vulnerable key is enough for an attacker. Long-lived keys and trust anchors first, then libraries, HSMs, wallets and profiles.
2029	**The estimates moved, and so did the targets.** RSA-2048: 20 million qubits (2019) to under one million (Google, 2025); P-256 about 10,000 neutral-atom qubits (Oratomic, 2026). Google and Cloudflare now say 2029.

OpenID Connect: The Good News

- OpenID Connect Key Rotation will enable OPs to provide updated signing keys
  - https://openid.net/specs/openid-connect-core-1_0.html#RotateSigKeys
- Larger ID Tokens will not hit size limits when using the Code Flow
  - Since passed in Token Endpoint response HTTP body
- ML-DSA JOSE algorithms already standards
  - RFC 9964 — ML-DSA for JOSE and COSE

OpenID Connect: The Bad News

- No means defined for RPs to signal to OPs they support new algorithms
- Post-quantum encryption algorithms for JOSE not yet standards
- Implementations need to be updated to support PQ algorithms
- Some implementations currently break when PQ keys published
  - Presence of new “akp” key type in JWK Set can cause errors
  - Described in blog post https://openid.net/post-quantum-openid-connect/
  - However certification suite now fails RPs with this bug

Libraries, platforms and PKI are the bottleneck

One AKP key broke 19 of 46 parsers

Raidiam's July to August survey of jwt.io libraries: 19 of 46 set parsers dropped the whole set, so RS256 tokens stopped verifying. Fixes are landing.

Java platform

ML-DSA and ML-KEM in the JDK since 24; hybrid TLS in the JDK's own TLS stack from JDK 27 (JEP 527), in BouncyCastle today. Nimbus 10.10 has no ML-DSA yet.

ML-DSA in JOSE libraries: patchy

Released: jose (JS), .NET IdentityModel, jwx (Go). Merged, unreleased: web-token (PHP). Planned: Nimbus, jose4j, JJWT. Not yet: PyJWT, java-jwt.

Hybrid TLS is widely deployed

X25519MLKEM768 is on by default in Chrome, Firefox, OpenSSL 3.5 and at Cloudflare. It protects key establishment only, not signatures or tokens.

Browsers

Chrome 150: ML-DSA for private PKI only. WebCrypto ML-DSA and ML-KEM: origin trial in 151 to 154, ship target 155. No public ML-DSA certificates yet.

Wallet hardware: here, unevenly

ML-DSA keys in Apple's Secure Enclave since iOS 26 and Android 17 Keystore since June 2026; PQ attestation, certified secure elements for EUDI and the installed base lag behind.

Conformance suite: what is already done

RP tests publish unusable keys

The OpenID Connect, FAPI 1, FAPI 2 Final and FAPI-CIBA client tests publish an ML-DSA-shaped AKP key and four other unusable keys in the server JWKS. Live since 17 September.

Suite recognizes ML-DSA names

ML-DSA-44/65/87 accepted in OpenID4VCI metadata and COSE; key selection from a counterparty JWKS skips keys it cannot parse, after an implementer's bug report.

Approved by the working groups

Connect WG consensus on 27 August: the RFC 7517 "SHOULD ignore" is a MUST for certification. FAPI WG agreed in July and August.

TLS 1.3 checks

The suite checks TLS 1.3 where profiles require it and probes mTLS endpoints for TLS 1.2. TLS 1.3 is necessary for hybrid key exchange, but not proof of it.

OpenID4VP wallets too

A dedicated wallet test surrounds the usable encryption key with an ML-KEM-shaped key and an unknown key type; the wallet must still encrypt its response.

Conformance suite: still to do

Unusable keys in every JWKS

Client JWKS in dynamic registration, OpenID4VCI wallets, Federation, SSF transmitters and receivers. Tracked as issue #1925.

Positive post-quantum tests

Optional ML-DSA tests are possible now against RFC 9964; making them certification requirements needs a profile that names the algorithm.

Unknown members and unknown algs

An unknown member on the usable key (an RFC 7517 MUST) and unknown alg values in metadata; proposed on the Connect issue, not yet designed.

Hybrid TLS and size limits

Record the negotiated group with the BouncyCastle TLS probe we already have; test real encoded ML-DSA tokens and requests (3 KB+ signatures) end to end.

ML-DSA in the suite itself

Nimbus has no ML-DSA; wire BouncyCastle so the suite can sign and verify ML-DSA JWS and COSE, then issue PQ-signed test credentials.

Help ecosystems measure readiness

A readiness view in the suite and reports, so Brazil, KSA and UAE can see which certified implementations ignore unknown keys.

Next: what you can do now, and what we are doing

Implementers: do now

Make JWKS handling tolerant
Put an AKP key in a test JWKS; the client must keep using the other keys. Certification already requires it.

Move to TLS 1.3 and turn on hybrid key exchange
X25519MLKEM768 is in current browsers, OpenSSL and CDNs; it protects against harvest-now-decrypt-later.

Know your inventory
Where RS256 is hard-coded, token and URL size limits, HSM and library roadmaps.

WGs, suite and staff: in progress

Suite
Unusable keys in every JWKS (#1925); ML-DSA signing and verification; hybrid TLS probe; real-size tests.

Working groups
Name a PQC champion; decide hybrid or pure ML-DSA per profile; then name algorithms in FAPI 2, HAIP, CAEP.

Staff
Ecosystem briefs, Brazil, KSA and UAE first; a "now and later" template for implementers.

Ask: test your library against an AKP key this week and tell your WG, or #interest-group-pqc-readiness on Slack, what breaks. Ecosystems: tell us your post-quantum dates so the tests can get there first.

Post-Quantum Resources

- Slack channel #interest-group-pqc-readiness
- Blog post https://openid.net/post-quantum-openid-connect/