
Post-Quantum Authentication: Preparing Digital Identity for the Quantum Era

Lessons from the Field — A CA Vendor Perspective

The Urgency of Acting Now

Organizations that delay PQ authentication migration risk compounding security, operational, and compliance challenges.^[1]

- CA vendors and operators face long deployment lifecycles spanning many years
- Delayed action creates security, operational, and compliance risk
- Quantum-safe authentication must be prioritized now, not treated as a future problem
- Entrust's experience updating PKI infrastructure revealed 3 main challenges:



Challenge 1 - TLS Client Authentication

Many third-party TLS software stacks were not ready: almost none supported basic ML-DSA authentication, let alone more conservative PQ options. [\[1\]](#)



Third-Party TLS Stacks

- Many third-party TLS stacks did not support ML-DSA client authentication
- Web browsers and servers using different crypto libraries also lacked ML-DSA support
- Configuring a stable, PQ-secure environment is extremely difficult today



Java LTS Releases

- Java LTS releases had no ML-DSA authentication support
- Enterprise deployments reliant on Java face an extended wait for viable PQ authentication



Conservative PQ Options

- SLH-DSA and Composite ML-DSA have even less adoption
- More conservative algorithm choices offer better security, but had limited support
- Organizations may need to decide between security strength verses implementation reality

Challenge 2 - HSM Systems

HSMs are adding the new algorithms, but fragmented support for algorithms and critical operational features creates real-world deployment challenges. [\[1\]](#)

Issue 01



Algorithm limitations

SLH-DSA and Composite algorithm support is not uniform across HSM systems leaving organizations without consistent PQ options.

Issue 02



PKCS#11 Interoperability

Early PKCS#11 adopters used vendor-specific defines, breaking interoperability between systems and complicating cross-vendor deployments.

Issue 03



ML-DSA Pure Mode Signing

ML-DSA pure mode required in certificates creates challenges when signing large data.

Issue 04



Missing Streaming APIs

Not all HSM vendors support "external mu" hashing or streaming APIs needed for deployment efficiency which limits performance at scale.

Challenge 3 - Cryptographic Message Syntax (CMS)

Internal control enabled PQ CMS upgrades and a workaround for TLS limitations. [1]

1 Direct PQ Upgrade - CMS Layer

Available CMS standards allowed internal software stacks to add support for PQ algorithms, including ML-DSA, SLH-DSA and Composites

 PQ CMS Authentication

ML-DSA / PQ algorithm

Temporary Workaround Solution

- This was not the preferred approach, but it was the best available given current third-party TLS stack limitations

2 Hybrid Tunnel - TLS limitation

When TLS client authentication could not be made PQ-secure, a layered hybrid approach was used as a work-around.

Classical TLS handles transport authentication, with PQ CMS authentication layered on top as part of the application.

 Classical TLS Transport

Outer layer

 PQ CMS Authentication

Inner layer



Conclusion: Control Your Cryptographic Layers

We could not rely on third parties to deliver PQ authentication

✕ Third-Party Dependency

- ✕ **TLS Clients & Browsers**
No ML-DSA authentication support across major TLS stacks stalled migration progress
- ✕ **Java LTS Releases**
No ML-DSA in current Long-Term Support versions causes issues for web apps
- ✕ **HSM Fragmentation**
Vendor-specific PKCS#11 defines and incomplete advanced algorithm support

✓ Internal Control

- ✓ **CMS Layer Upgrades**
Internal CMS stacks were upgraded to PQ algorithms without issue
- ✓ **Hybrid Workaround**
PQ CMS authentication layer added on top of classical TLS allowed PQ authentication
- ✓ **Prioritize What You Own**
Identify stack layers under internal control and migrate those first

📌 **Practical guidance:** identify what you control vs. what depends on third parties. Do not wait for ecosystem-wide readiness; architect hybrid approaches to work-around limitations.

References

- [1] [EntrustPositionPaper.pdf](#)