

Post-Quantum

Cryptography Conference

PKI deployments are as unique as any snowflake; how to build equally flexible PQ migration strategies

Greg Wetmore

Vice President Product Development at Entrust

PKI deployments are as unique as any snowflake – how to build equally flexible PQ migration strategies

Greg Wetmore

Vice President Software Development, Entrust



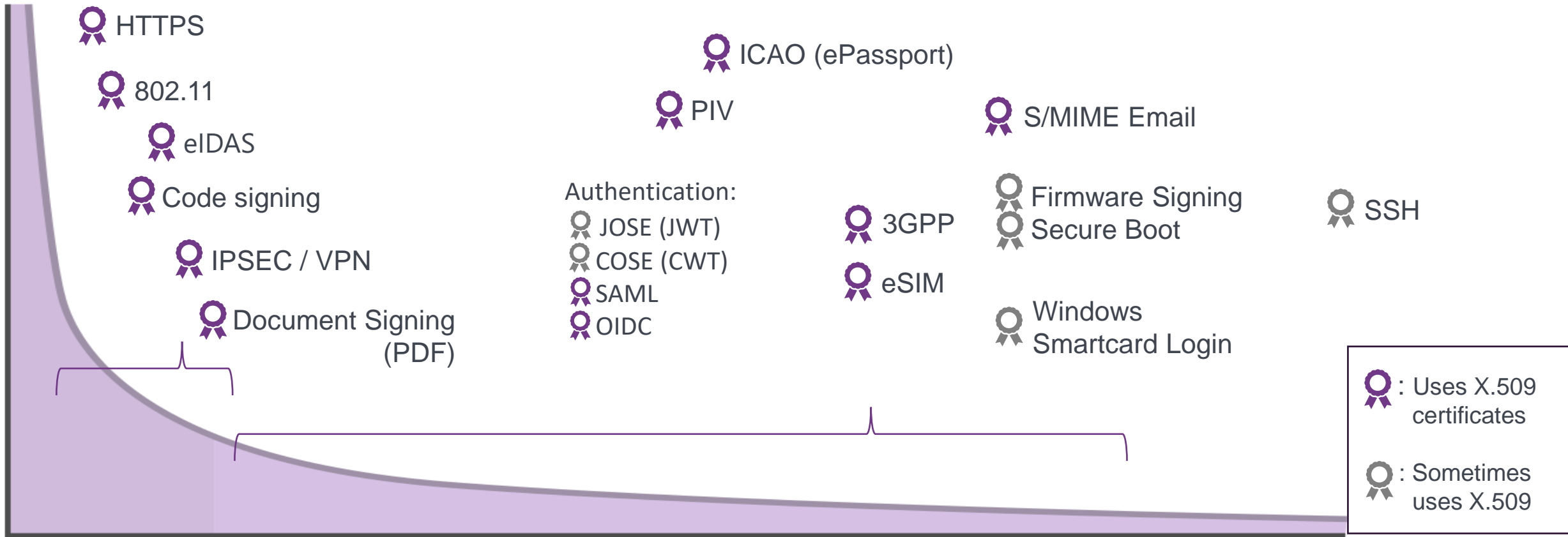
ENTRUST

SECURING A WORLD IN MOTION

The Long Tail of X.509 Usage

“CORE” X.509 USES

“SPECIALIZED” X.509 USES



Migration for “Core” X.509 Use Cases

- Core X.509 use-cases will likely have prescribed migration paths provided by platform providers: Cloud IaaS, Microsoft, CA/B Forum, Networking & VPN, etc.

But only:

IF you can reach all your client devices to upgrade them.

IF you can easily stand up new PKIs and distribute their root certificates.

IF you use standardized and crypto-agile network protocols and don't have any custom crypto code anywhere.

IF you can find copies of your code-signed .exe's and digitally-signed contracts to re-sign them.

IF you ...

- Experts say that this will be the **most challenging** cryptographic migration that we have ever done.

... and this is the easy half of the problem.



ENTRUST

Migration for “Specialized” X.509 Use Cases

- Specialized X.509 use-cases will require careful consideration and will benefit from flexible *Tools*.
- PKIs and Certificates are everywhere – used for all kinds of reasons and applications:
 - IoT Devices: roots of trust, device identity, firmware integrity
 - Smartcards and Identity documents: physical/logical access cards, National IDs, ePassports
 - Short lifetime & single-use certificates
 - Non-repudiation: digital signing of legal documents and contracts.
 - Certificate-based client-authentication:
 - Device-based (ex.: WiFi access point, smartcard Windows login),
 - TLS-based (ex.: cert-based REST API),
 - File-based (ex.: disk encryption, encrypted backup files).
- **Takeaway:** Simple PQ migration strategies will not necessarily work for
- specialized uses of X.509.

Classification of PQ Migration Challenges



ENTRUST

The Million-Dollar Question

Is your PKI “Core” or “Specialized”?

- Your PQ migration is “Core” if you can answer **YES** to the following questions:
 - X.509 certs are used exclusively with standardized negotiated network protocols (TLS, IPSEC, etc).
and
 - You can reach and patch all your servers to introduce PQ by the 2025 “Support and prefer” deadline.
and
 - You can reach and patch all your clients to introduce PQ by the ~ 2030 sunset date for RSA, DH, and ECC, and then re-patch all servers to remove them completely.
and
 - You can easily stand up new PKIs and re-issue certs to all devices.

National Security Agency – Timelines

- **Software and firmware signing:** begin transitioning immediately, support and prefer CNSA 2.0 by 2025, and exclusively use CNSA 2.0 by 2030.
- **Web browsers/servers and cloud services:** support and prefer CNSA 2.0 by 2025, and exclusively¹ use CNSA 2.0 by 2033.
- **Traditional networking equipment** (e.g., virtual private networks, routers): support and prefer CNSA 2.0 by 2026, and exclusively use CNSA 2.0 by 2030.
- **Operating systems:** support and prefer CNSA 2.0 by 2027, and exclusively use CNSA 2.0 by 2033.
- **Niche equipment** (e.g., constrained devices, large public-key infrastructure systems): support and prefer CNSA 2.0 by 2030, and exclusively use CNSA 2.0 by 2033.
- **Custom applications and legacy equipment:** update or replace by 2033.



Classifying the “Specialized” – Data Requirements

If you answered **NO** to any of the previous questions, then your PQ migration will require further study and planning.

Some starter-questions

- Do you have a cryptographic inventory – ie do you know where all your crypto is?
 - Are you using mainly open standardized crypto layers (IETF, ISO, X9, transparent platform-provided crypto)?
 - PQ transition may be prescriptive, BUT
 - Configuration changes, Key & Certificate Management, Legacy Environments
 - Do your applications have direct crypto dependencies: that will engineering effort to find and upgrade?
- What is the data sensitivity lifetime?
 - If seconds, like point-in-time transactions (ex.: money transfer), then less urgency.
 - If decades, like personal data or signed contracts, then it’s important to get it right.

Classifying the “Specialized” – Infrastructure Requirements

- Some starter-questions
 - Is it servers or clients that you can't easily patch?
 - Hardware and vendor lock-in, or 10 year hardware replacement schedules may need to be re-evaluated.
 - Are your clients and servers talking over private trusted networks, or over the public Internet?
 - Is your environment “homogeneous” (clusters of things upgraded together), or “heterogeneous” (new things need to continue talking to old things)?
 - Will non-upgraded components continue producing sensitive data that requires long-term protection?
 - Can your protocols gracefully handle adding hybrid PQ/T encryption, or does this need engineering to modify protocols or add custom “shim” or “wrapping” layer before that data crosses untrusted networks?

Classifying the “Specialized” – Cryptography Requirements

Some starter-questions

Do you have requirements for the choice of cryptographic algorithms?

- Compliance with national standards under which you must operate (ex.: NIST/FIPS, BSI, ENISA, China, etc).
- Do you have high-value long-term data where new cryptography introduces too much risk?
 - Do you have long-lived data or cryptographic infrastructure that would be difficult to replace and re-protect in an emergency?
- Do you have size, bandwidth, or CPU constraints that would be violated by a naïve application of PQC algorithms?







Classification of PQ Migration Solutions



ENTRUST



What's in the Toolbox?

TOOL		CORE	15+ YR DATA / INFRA LIFE	5 – 15 YR DATA / INFRA LIFE	HETEROGENEOUS ENV BACKWARDS COMPATIBILITY
Hard Flag Day (cutover)		<input checked="" type="checkbox"/>			
Soft Flag Day (Negotiated Protocols)		<input checked="" type="checkbox"/>			<input checked="" type="checkbox"/>
Multiple Signatures (CMS)		<input checked="" type="checkbox"/>			<input checked="" type="checkbox"/>
High-Assurance algorithms (Hash-based, C. McEliece)			<input checked="" type="checkbox"/>		
Hybrids: Composite				<input checked="" type="checkbox"/>	
Hybrids: Multi-cert AltPublicKey Chameleon					<input checked="" type="checkbox"/>



Flag Day



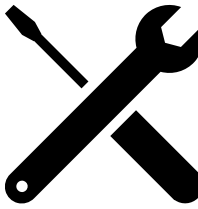
- Hard Flag Day
 - All systems do a hard switch-over at a designated time and no longer accept the old algs.
- Soft Flag Day
 - Protocol-level cryptographic agility allows for staged migration;
 - Upgraded systems can gracefully accept old crypto until the ecosystem hits 100% adoption.
 - Examples:
 - TLS
 - IPSEC
 - For some time period, servers support both the PQ and traditional cipher suites.
 - Data transmitted by non-upgraded clients is at risk “Harvest now, decrypt later”, but at least can still connect.

Multiple Signatures (CMS)

Core



Heterogeneous env
Backwards compatibility

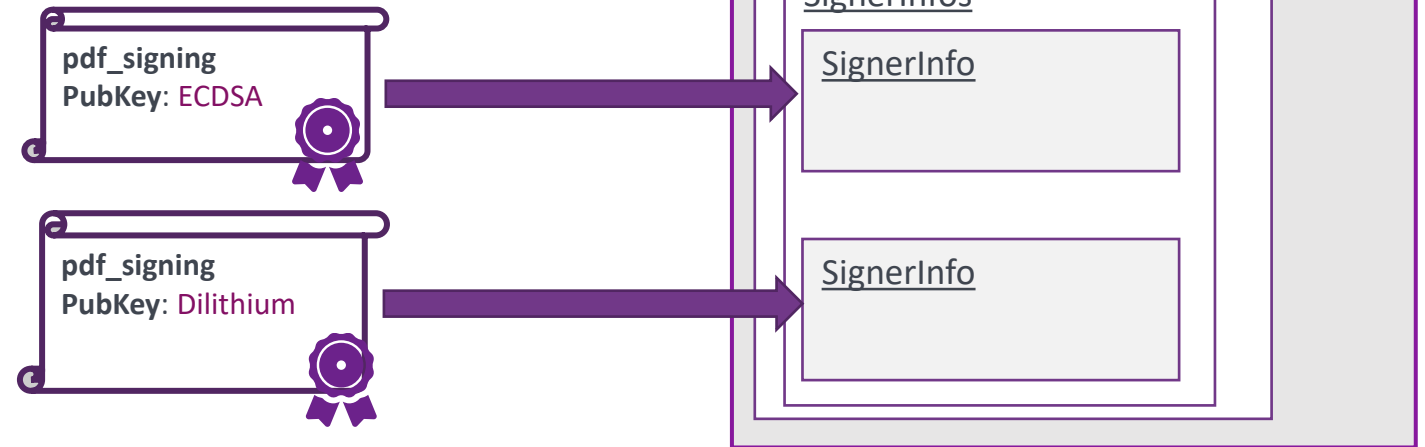


Example of applying hybrid PKIs to **Cryptographic Message Syntax (CMS)**

(CMS – aka PKCS#7 – underlies all sorts of protocols: S/MIME, code-signing, PDF signing)

RFC5652 (CMS) - SignerInfos:

“When the collection represents more than one signature, the successful validation of one of the signatures from a given signer ought to be treated as a successful signature by that signer...”



- › Backwards compatibility: CMS clients (S/MIME, code-signing, PDF) can already handle multiple *SignerInfos* today.
 - So legacy clients **should** gracefully skip the PQ signature that they can't parse.
- › Redundancy gives migration flexibility. PQ-aware clients can validate either:
 - PQ signature only, or
 - Both signatures.

High-Assurance Algorithms

15+ yr
data / infra life

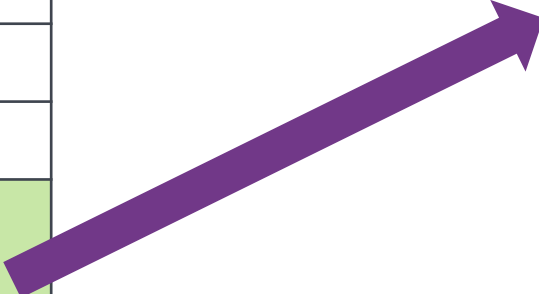


- Lattice schemes (ML-DSA, FN-DSA, ML-KEM) offer reasonable performance and key sizes, but there is concern that the algorithms will be broken by yet-to-be discovered mathematical attacks within the next decade.
- **Hash-based signature algorithms** are based only on hash functions, which are very well studied, so we trust their security completely.
 - But they have large signatures (~20 kb), slow signing times (200x ECDSA), and extremely large private keys and keygen times.
- **Classic McEliece encryption algorithm** has been studied as a cryptosystem since 1978 without any significant breaks or cryptanalytic advancements.
 - It has very small ciphertexts, but extremely large public keys (0.25 – 1.3 mb).
 - Any Classic McEliece deployment will want some sort of out-of-band public key distribution.

Aside: Addressing certificate size with Externalized Public Keys

New idea to explore: External Public Keys ¹ in order to keep the certificates themselves small.

CERTIFICATE
Subject: cn=joe
Issuer: CACorp
Serial: 07
SPKI: { location: http://joe.com/pubkey hash: 8eff38e8... }
Extensions:
SANs: joe.com
Sig: SLH-DSA {a620bf96d6b..}

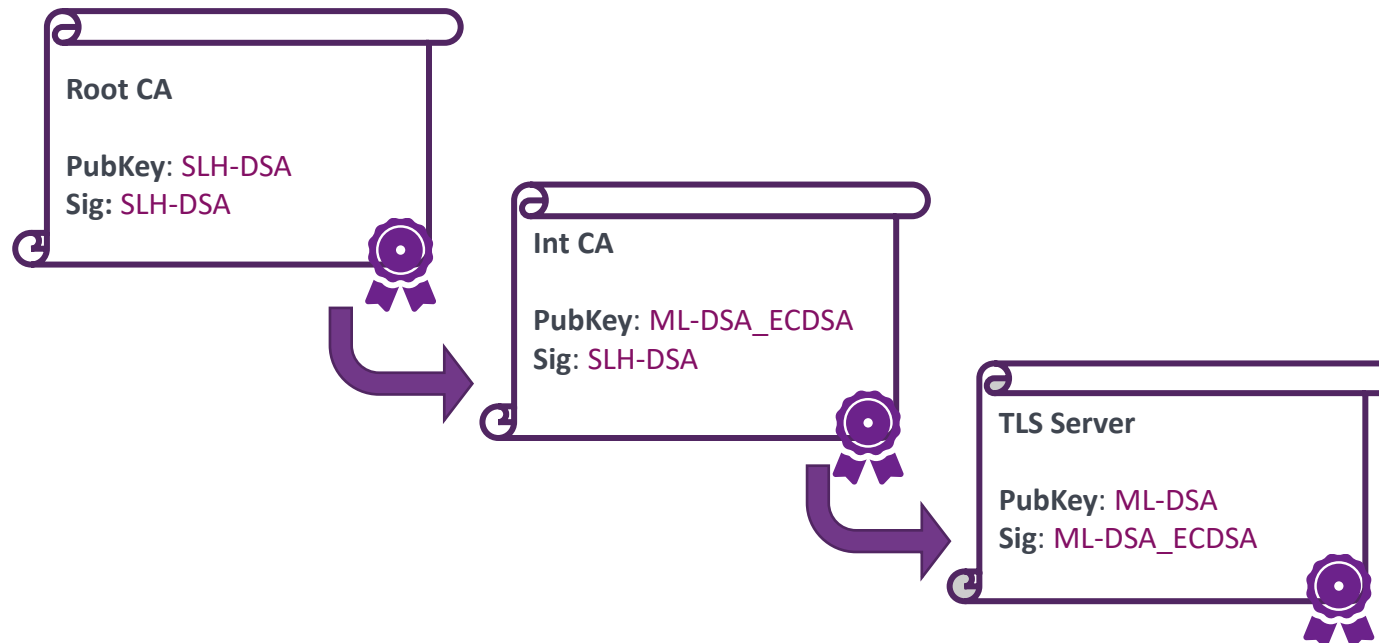


```
-----BEGIN PUBLIC KEY-----  
MIIBigKCAyEAq3DnhgYgLV...  
ar4jRygpzbgHlFn0Luk1mdV...  
...  
jXPqy/ZJ/+...  
-----END PUBLIC KEY-----
```

Driving use-case is Classic McEliece certificates (0.25 – 1.3 megabyte public keys).

Aside: Addressing certificate size with Mixed PKIs

- Consider “mixed” architectures with larger / stronger algorithms on long-lived objects such as root CAs, and smaller / weaker algorithms on short-lived objects such as end entity certificates or TLS handshakes.



Aside: Other options for addressing Performance / Size

Some other potential areas for exploration:

- Clients do more aggressive caching and out-of-band cert distribution.
 - Browsers already do this to some extent. Could be done more aggressively, and by non-browser clients.
 - Need to be careful of privacy leaks; aggressive caching of certificates leaks which sites you have been to before.
- Some wholesale X.509 replacement like Merkle Tree Certificates¹ or Merkle Tree Ladders².
- Or the no-op solution: just deal with the fact that certs are bigger now.
 - It's still unclear which usecases actually **fail** if you violate the TCP packet / frame size limit, and which usecases don't even notice the fragmentation.

1: <https://datatracker.ietf.org/doc/draft-davidben-tls-merkle-tree-certs/>

2: <https://datatracker.ietf.org/doc/draft-harvey-cfrg-mtl-mode/>

CALL FOR “PQ/T HYBRID BRIDGE”

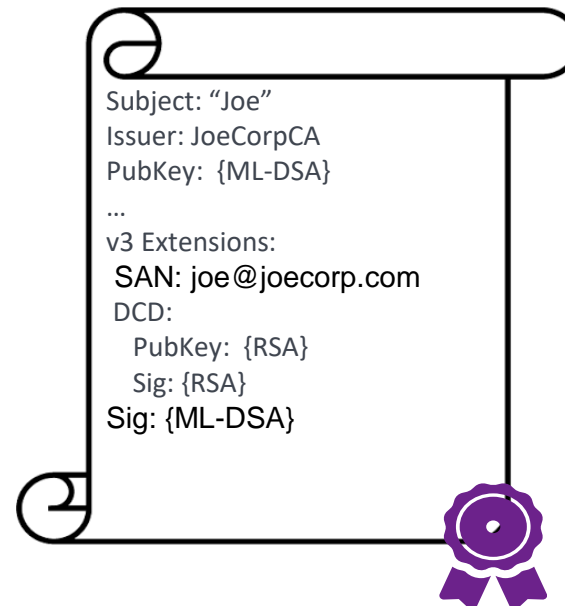
“A dual signature consists of two (or more) signatures on a common message. It may also be known as a hybrid signature or composite signature.”



Federal Office
for Information Security

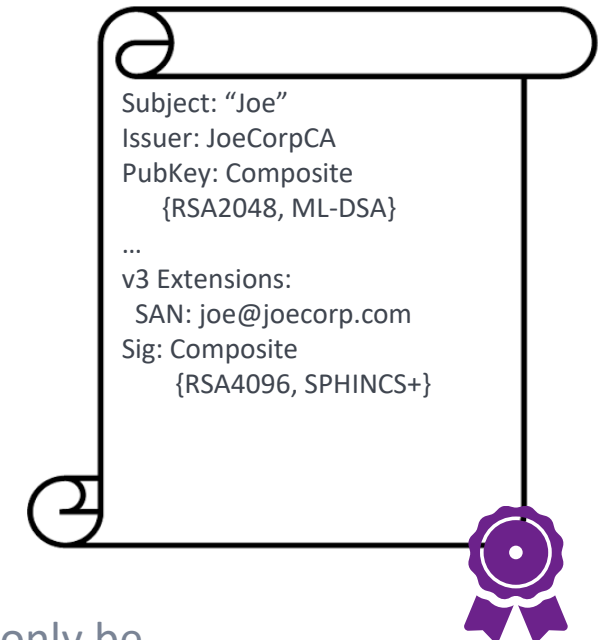
CHAMELEON¹

IETF: draft-bonnell-lamps-chameleon-certs



COMPOSITE²

IETF: draft-ounsworth-pq-composite-sigs



Post-quantum methods should only be used in combination with classic methods whenever possible. ³



ENTRUST

¹ Entrust – DigiCert – Keyfactor; IETF

² Entrust – CableLabs – Cisco collaboration; IETF drafts

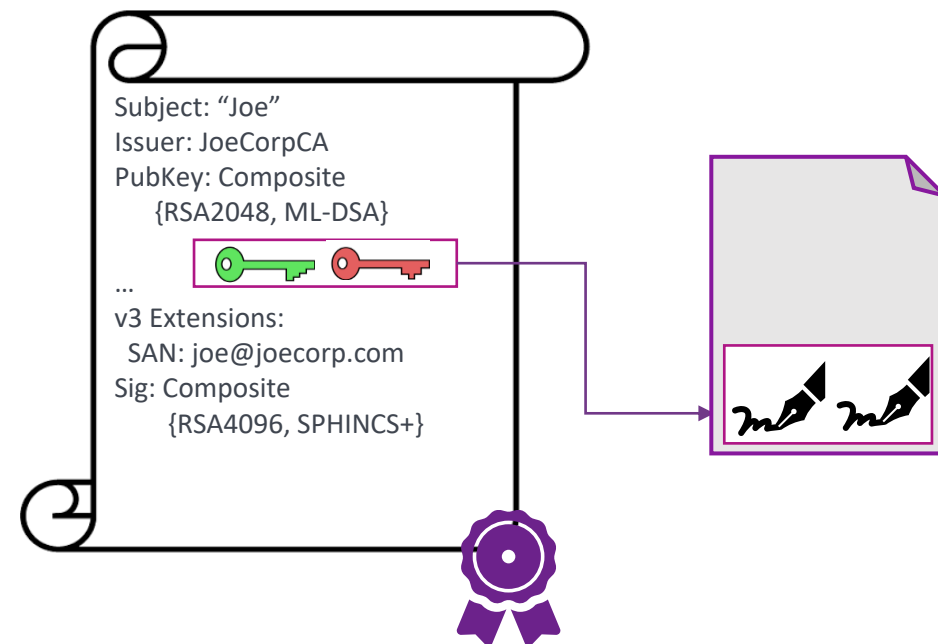
³ BSI Federal Office for Information Security

Hybrids: Composite

5 – 15 yr
data / infra life

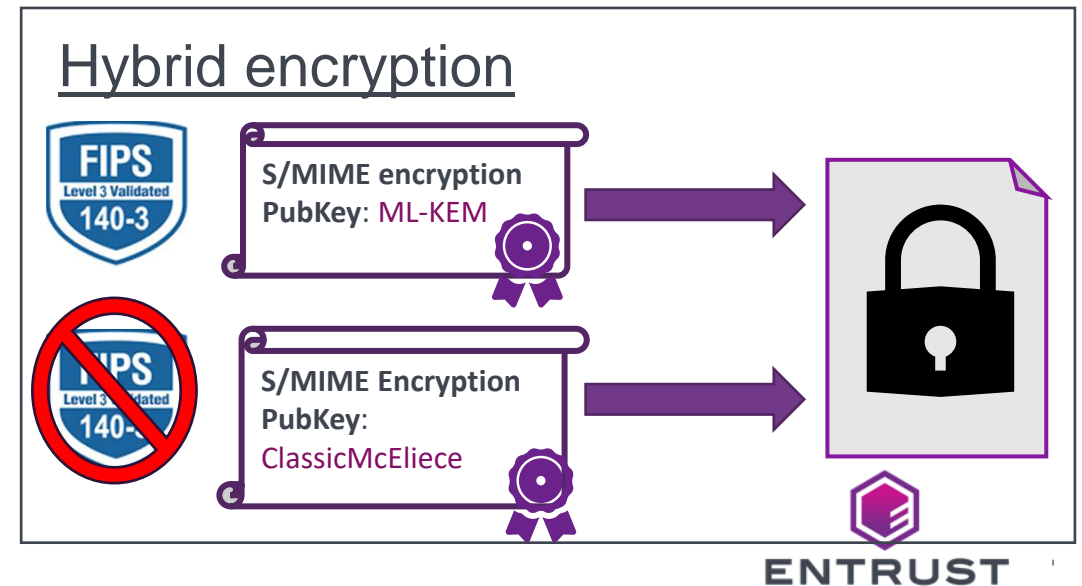


- Designed to be the obvious and straightforward implementation of hybrid dual signatures:
“Just have one key that is actually two keys inside”
- All the complexity of the hybridization is hidden inside the crypto library (written by crypto experts).
- Protocol backwards-compatibility: no need to modify anything at the protocol or application layer because they simply see one key and one signature.



Aside: Composites for satisfying regulatory compliance

- Wait for FIPS-validated SLH-DSA, ML-DSA, ML-KEM implementations (2026?), or deploy sooner with non-validated implementations?
- Differences in national standards; for example BSI (Germany) allows Classic McEliece and FrodoKEM, but NIST (US) does not (yet).
- Composites may address compliance in a few different ways:
 - Combine pre-standards / pre-certified PQC with certified ECC.
 - Combine crypto from two jurisdictions.

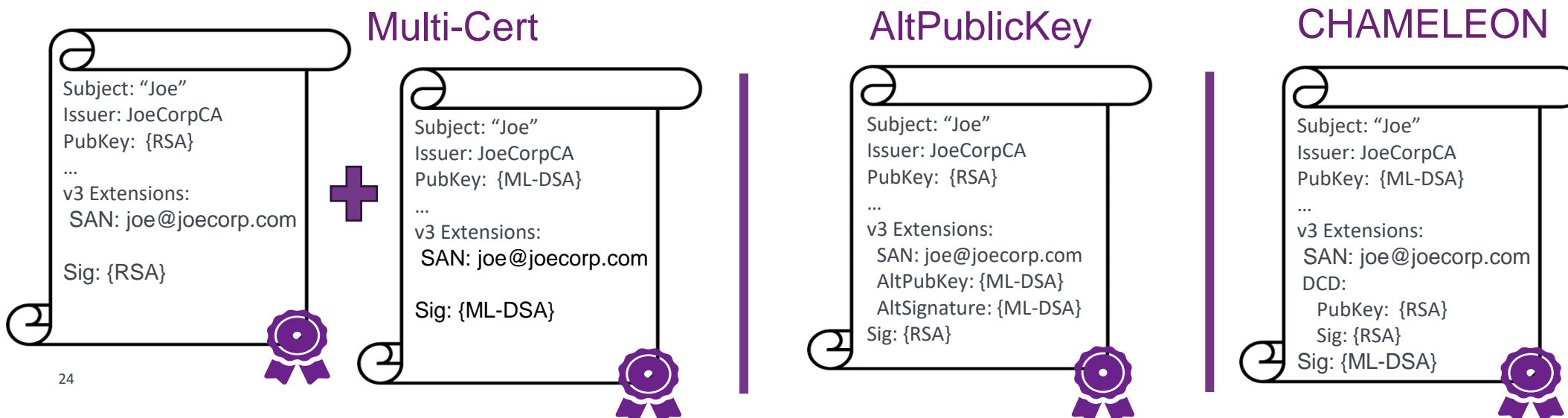


Hybrid Certificates

Heterogeneous env
Backwards compatibility



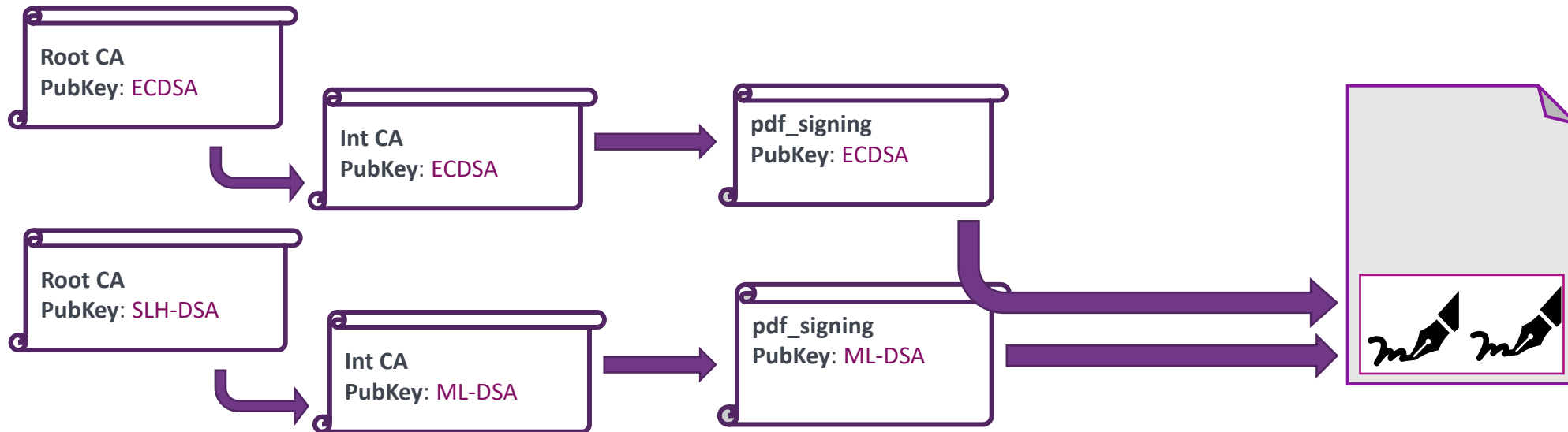
- Any of several “Hybrid certificate” options that each offer slightly different Pros / Cons.
- Consider whether your use-case needs:
 - **OR mode** where client selects one or the other to use (for backwards-compatibility)
 - **AND mode** where both are used together (for forwards-security).
- This only works if the protocols and clients that you use with X.509 know how to carry multiple signatures, or negotiate which one to use (ie it needs feasibility analysis).





Hybrids: Multi-Cert

- Ideal for protocols based on CMS that already allow multiple signatures on a document.
- Allows you to keep your existing PKI and supplement it with a second PQ mirror copy.

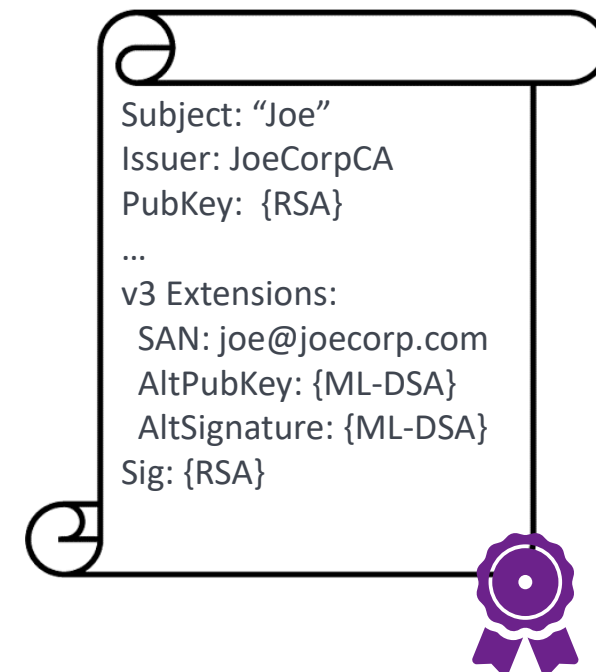


Hybrids: AltPublicKey

Heterogeneous env
Backwards compatibility



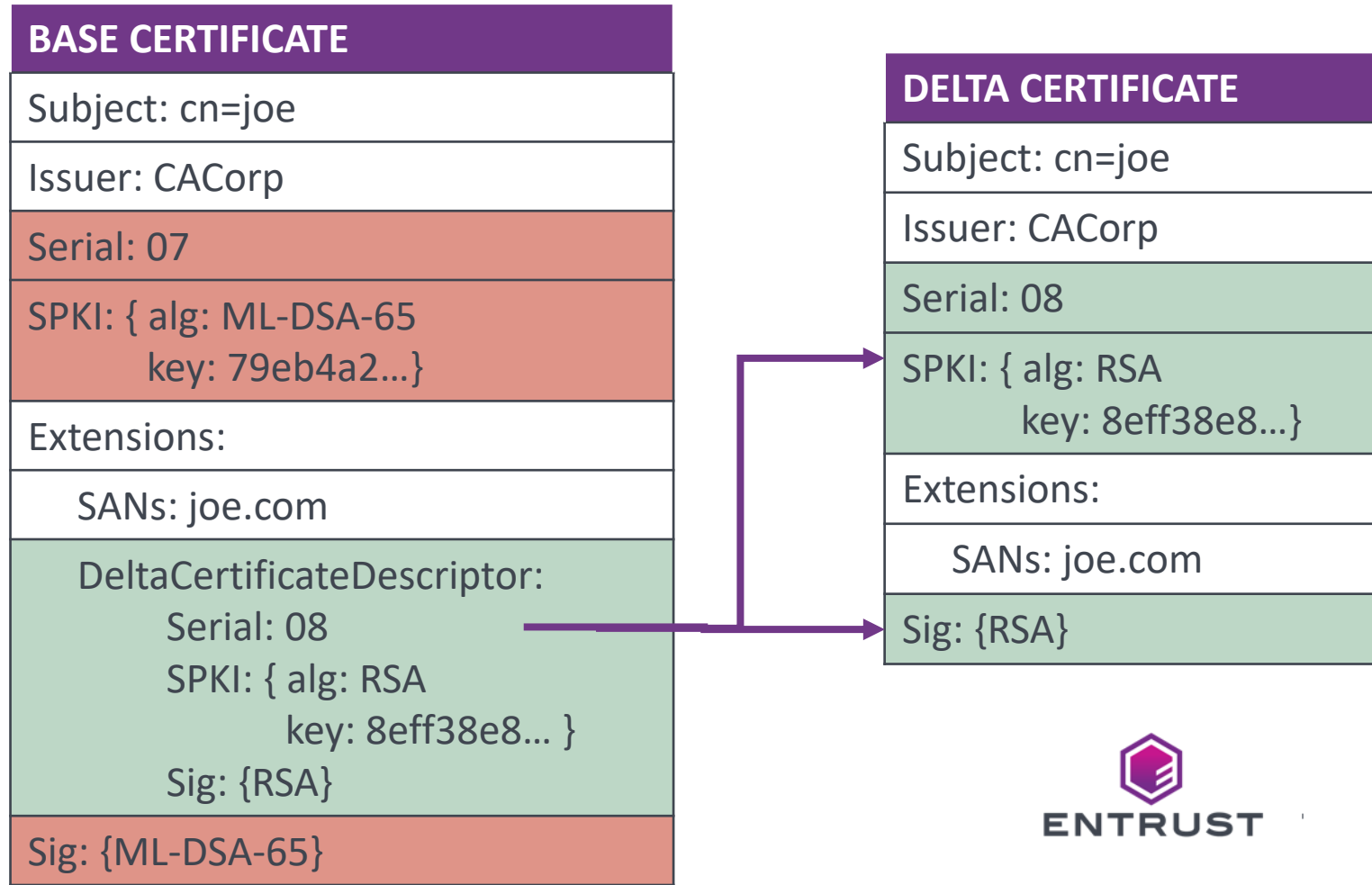
- Standardized in ITU-T / X9 2019 version of X.509.
- Pros:
 - Adopted early.
 - Graceful fallback to legacy
(because unrecognized v3 extensions will be skipped).
- Cons:
 - Large PQ keys need to be transmitted whether or not they are used.
 - Difficult to use with existing protocols that expect a single signature.





Hybrids: Chameleon

- DeltaCertificateDescriptor is a v3 extension that allows you to encode the differences between two parallel certificates.
- Basically, you can “bury” one cert inside another and easily “extract” the inner cert.
- Pros:
 - All the Pros of multi-cert, but with only one cert to manage.
 - Can drop the large PQ keys when not needed.
- Cons:
 - Difficult to use with existing protocols that expect a single signature.



Thank you

greg.wetmore@entrust.com

entrust.com

© Entrust Corporation



ENTRUST

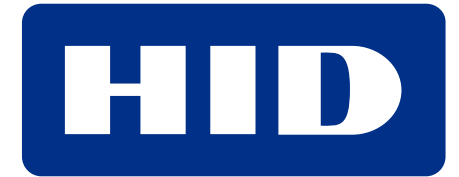
SECURING A WORLD IN MOTION

Post-Quantum

Cryptography Conference



PKI
Consortium



KEYFACTOR



THALES

