Keyfactor
Tech Days
2023

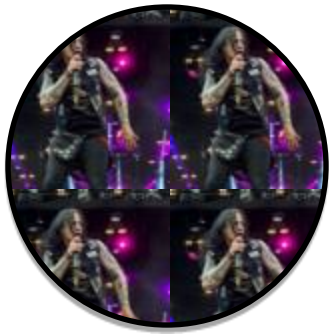
Hardcore PKI

Tomas Gustavsson

Chief PKI Officer

Keyfactor

2015 2016 2017 2018 2019 2020 2021 2023



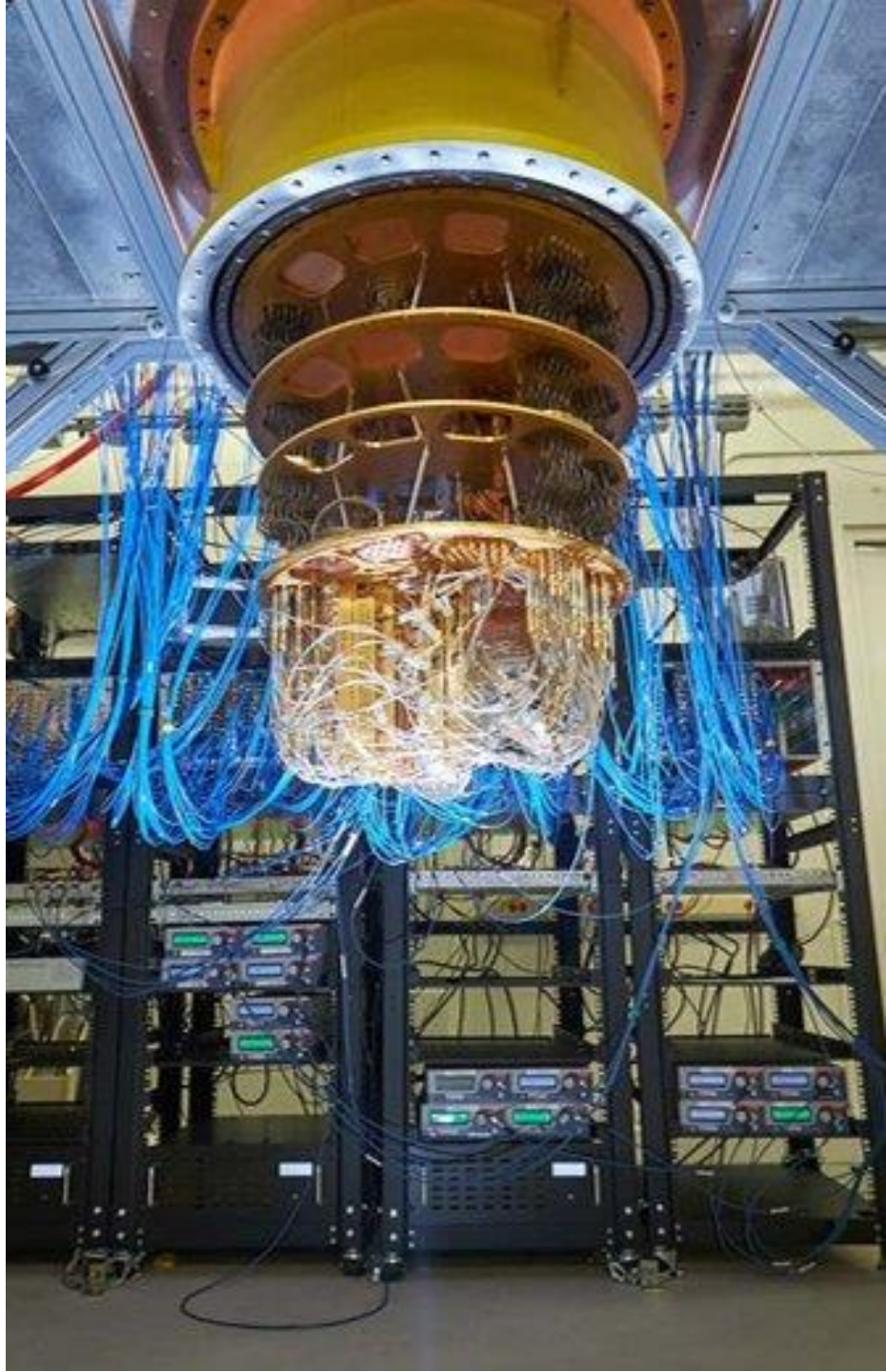
~~Hard Core~~
Hardcore

Hardcore PKI – 2023

- Post Quantum
  - More than algorithms
- System operations
  - Features is the easy thing
- IoT
  - Emerging immaturity – yet more options

Post-quantum is here!

- Migration – threat or not
  - SHBS Algorithms
  - NIST and other Standardization
  - HSMs
  - POP with KEM keys
- Filter through the noise



Operations

Security is more than features

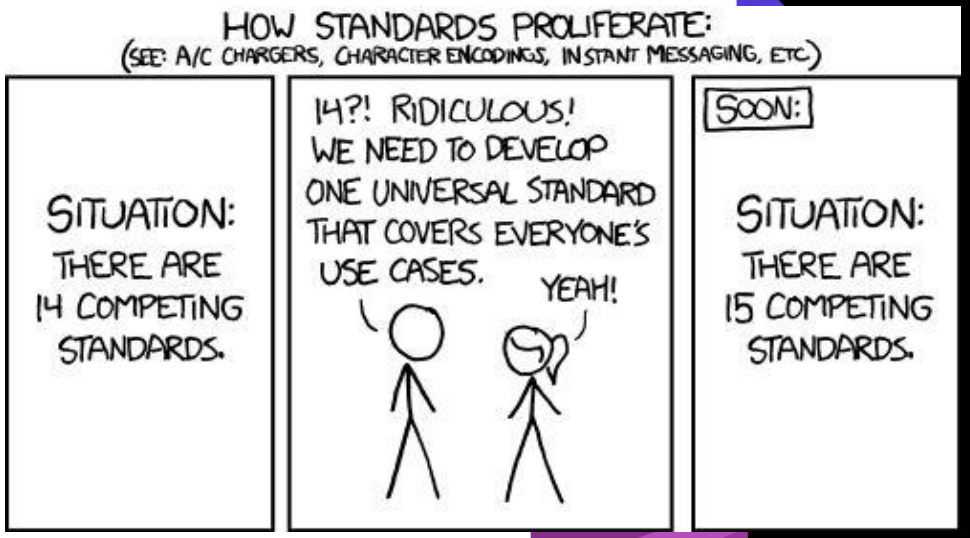
- Put in production
  - Service, without outages, 24/7
  - Maintainability
  - Availability
  - Ease of use
  - Security

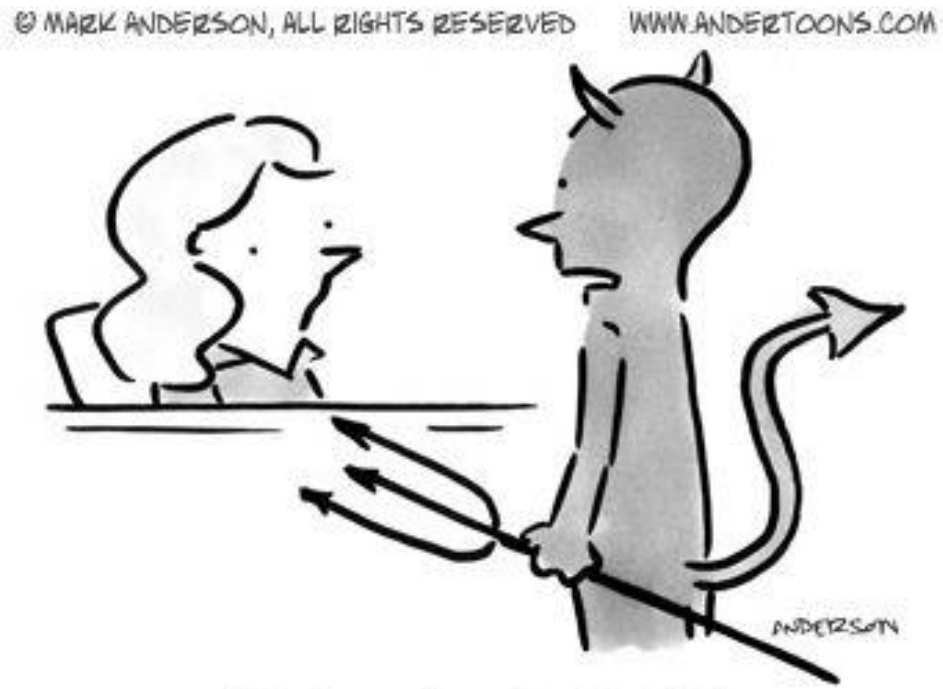
IoT Use Cases

Adapting to new (constrained) use cases

- Limiting further than standards
- Using standards-allowed features not used before
- How many options can you handle?

Revisit *IoT Standards*, *Making PKI Hard* and *Ease of Use*





"I'm here about the details."

Post-Quantum

- Migration to new algorithms will happen
  - Don't migrate from RSA to EC?
- What happens to PKCS#10 CSRs?
  - PQ Signature algorithms – POP OK
  - PQ Encryption algorithms – POP NOK
  - RFC4211 (CRMF) explicit POP: POPOPrivKey->subsequentMessage->encrCert
- Sift through the noice
  - *breaking-rsa-with-a-quantum-computer…*
  - Among others… Shor and Mosca says no…

Post-Quantum

- Standardization
  - SHBS – FIPS 800-208
    - Private key can not leave the HSM
    - No Backup(?)
  - Sign and Encrypt

- PKCS#11
  - LMS: soon-ish
  - Dilithium et al: Not until standards are out
  - Vendor defined attributes
    - if and but
    - sending large data (pre-hash or not)

- REST based HSMs…

OASIS

PKCS #11 Cryptographic Token Interface
Base Specification Version 3.0

```
C_SignInit->C_SignUpdate->C_SignFinal
vs
C_SignInit->C_Sign
```

```
if (mechanism == CKM.EDDSA) {
    if (myKey.getSlot().isThalesLunaHsm()) {
        // From cryptoki_v2.h in lunaclient
        final long LUNA_CKM_EDDSA = (0x80000000L + 0xC03L);
        mechanism = LUNA_CKM_EDDSA;
    } else if (myKey.getSlot().isUtimacoHsm()) {
        /// utimaco detected. CKM.EDDSA=>CKM.ECDSA
        mechanism = CKM.ECDSA;
    }
}
```

Post-Quantum

- Wait, there is more…

- Key sizes
  - Not obvious with PQC
  - "security strength"; FIPS 800-57
- Benchmarks
  - Key sizes, signature performance (HSM/No HSM), …

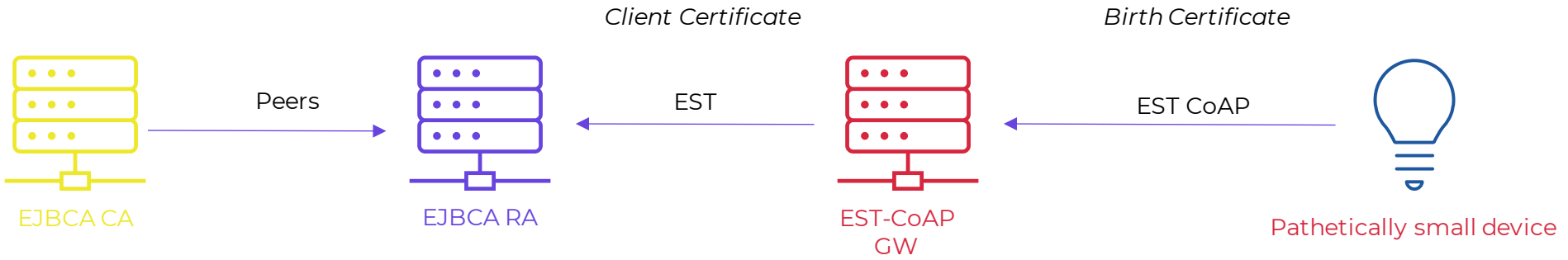
Security Strength	Symmetric Key Algorithms	FFC (DSA, DH, MQV)	IFC* (RSA)	ECC* (ECDSA, EdDSA, DH, MQV)
128	AES-128	$L = 3072$ $N = 256$	$k = 3072$	$f = 256\text{-}383$
192	AES-192	$L = 7680$ $N = 384$	$k = 7680$	$f = 384\text{-}511$
256	AES-256	$L = 15360$ $N = 512$	$k = 15360$	$f = 512+$

CA Name	**PQCDilithium3** (817931096)
Certificate Type/Version	X.509 v.3
Certificate Serial Number	268F30DC7841B72E808ABBAA9B1EI
Issuer DN	CN=PQCDilithium3
Valid from	2023-01-26 09:13:48+01:00
Valid to	2028-01-25 09:13:47+01:00
Subject DN	**CN=PQCDilithium3**
Subject Alternative Name	
Subject Directory Attributes	None
Public key	DILITHIUM3 (192 bits)

Operations

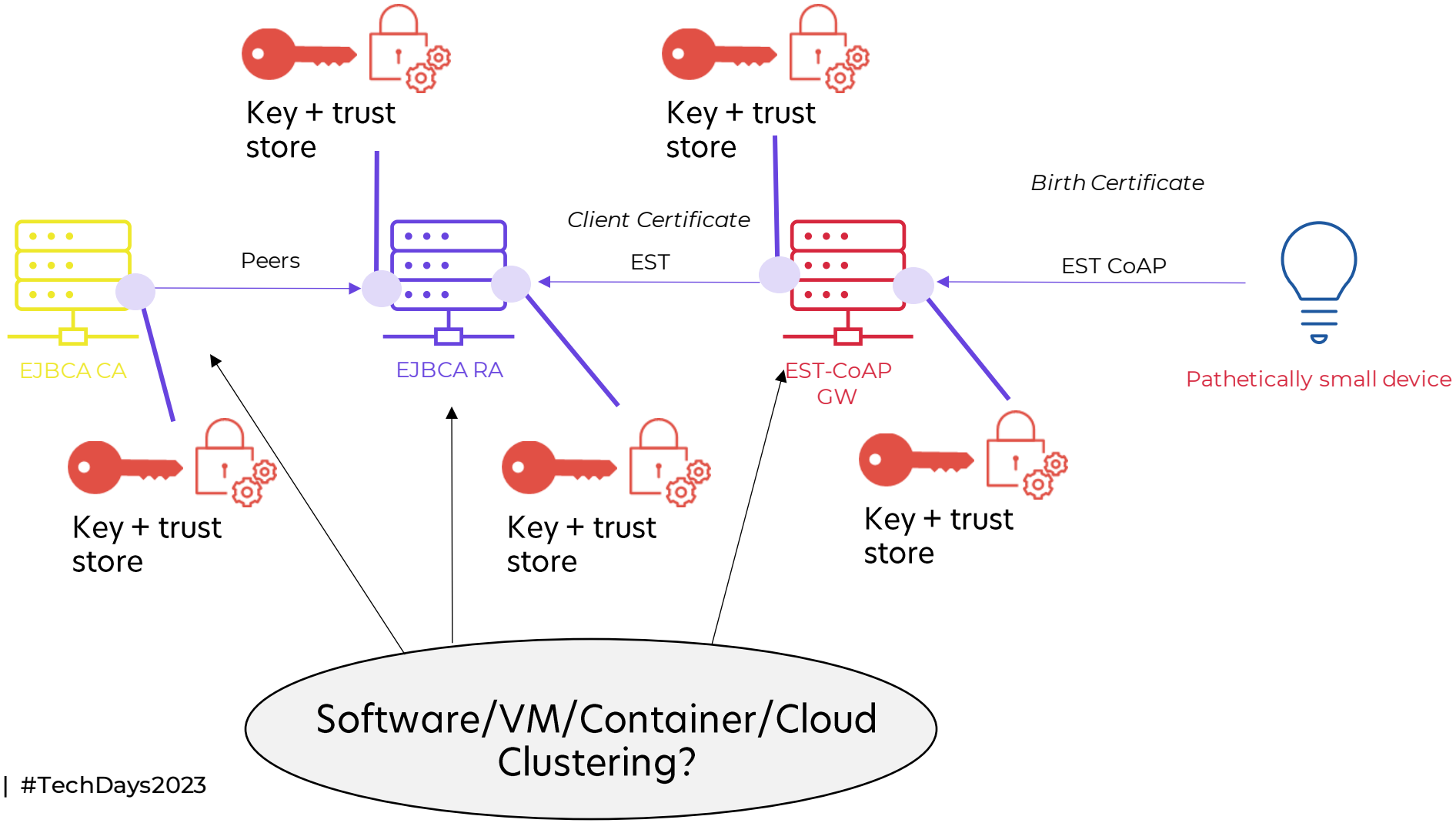
- Where most of the work is spent
- Going from PoC to production

Example: EST-CoAPs



- PoC functionality quickly
- Operationalize/Management/Ease-of-use/Security/Agility
- HA model

Example: EST-CoAPs



IoT Use Cases

- Even in 2023 there are new options
  - Extension order
  - EC compressed points
  - ...

Add, remove, edit or view available Custom Certificate Extensions[?]

ID	Object Identifier (OID)	Label	Critical	Require
1	2.5.29.19	Basic Constraints	Yes	Yes
2	2.5.29.35	X Authority Key Identifier	No	Yes
4	2.5.29.14	Subject Key Identifier	No	Yes

Uncompressed and compressed format

EC Public Keys, with named curves, can be encoded in two different ways in the SubjectPublic uncompressed form. While they are equivalent when using the public key, the byte encoding c with both formats from EJBCA 8.0. CAs will be created using the uncompressed format, while

- If the CSR contains a SubjectPublicKeyInfo using named curve in uncompressed forma
- If the CSR contains a SubjectPublicKeyInfo using named curve in compressed format,
- If the CSR contains a SubjectPublicKeyInfo using explicit parameters in uncompressed

ECC named curves vs explicit parameters

Normally, you want to generate requests and certificates using r explicit parameters instead, this is for instance mandated by ICA

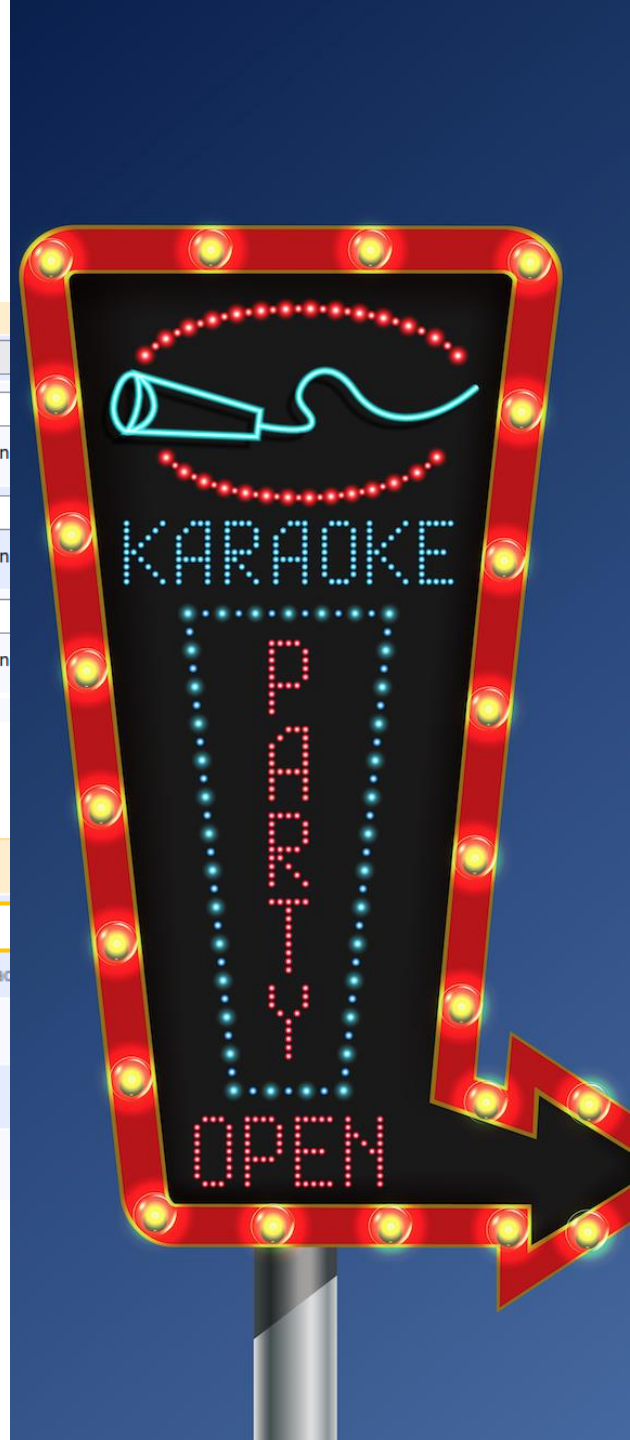
- When generating requests with *clientToolBox PKCS11HSM*
- When creating CAs with *ejbca.sh ca init* you can specify a
- When EJBCA issues certificate with public keys from cert

To create a CA with explicit EC parameters in the certificate you

```
if ((publicKey instanceof BCECPublicKey)) {
    log.debug("CSR has compressed EC point format, setting COMPRESSED as certificate SubjectPublicKeyInfo encoding");
    ((BCECPublicKey)publicKey).setPointFormat("COMPRESSED");
    pkinfo = SubjectPublicKeyInfo.getInstance(publicKey.getEncoded());
}
```

IoT Use Cases

- "Even in 2023" there are new options
  - Matter Subject DN fields
  - Ed25519 vs ECDSA vs LMS vs XMSS vs Dilithium vs Falcon



Subject DN Attributes [?]	
Subject DN Attributes	O, Organization
CN, Common name	Required Modifiable Validation
Matter VID	Required Modifiable Validation
Matter PID	Required Modifiable Validation

- ECDSA wapip192v1
- Ed25519
- Ed448
- FALCON-512
- FALCON-1024
- DILITHIUM2
- DILITHIUM3
- DILITHIUM5
- DSTU4145 163

CA Certificate Data	
Subject DN	CN=Keyfactor Matter PAA,VID=FFF1
	DN in string form, e.g. 'CN=My CA,O=MyOrg,C=SE', elements will be ordered ac
Signed By	Self Signed
Certificate Profile	MatterPAA
end date of the certificate [?]	10y

“

I learn something new about PKI every week

Tomas Gustavsson

Chief PKI Officer

Keyfactor