The Anatomy of Cryptography Bills of Materials: Standardization and Practice in CycloneDX

—

Basil Hess
IBM Research Europe - Zurich

Migration and Agility in Cryptographic Systems (MAgiCS)

Rome, Italy

May 10, 2026

IBM Research Security

IBM

Agenda & Overview

> *A Cryptography Bill of Materials (CBOM) is an object model to describe cryptographic assets and their dependencies.*

1. Challenges in creating CBOMs, and how a specification addresses this
2. xBOM: the CBOM building blocks
3. Evaluation with two assets of different production stages
4. Future directions

Challenges in creating actionable CBOMs

- A grep for elliptic curve secp256r1 in OpenSSL 3.6.1 source code finds **74** occurrences, and **433** occurrences for md5. What does this tell us?

Naming & Taxonomy
secp256r1 = prime256v1 (164 hits) = P-256 (52 occurrences)

Dependencies
OpenSSL provides cryptography, but: who uses it, what are other external dependencies?

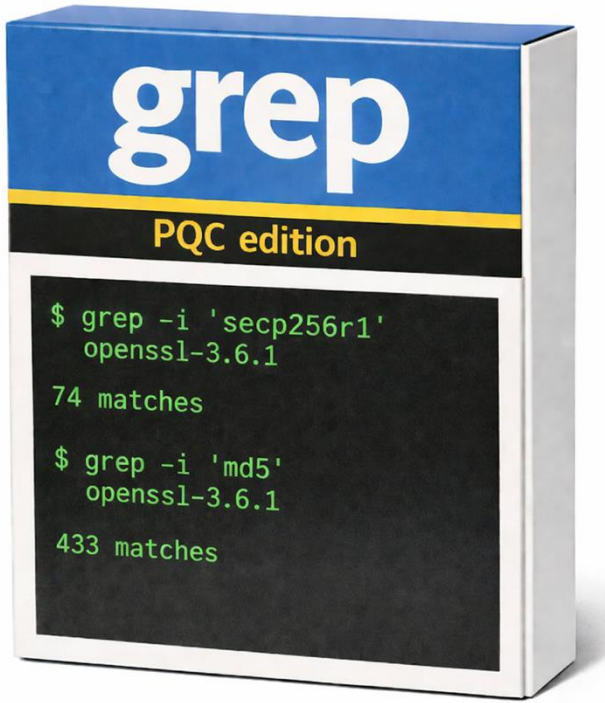
Configuration-driven crypto
Static analysis alone cannot determine actual usage

CBOM

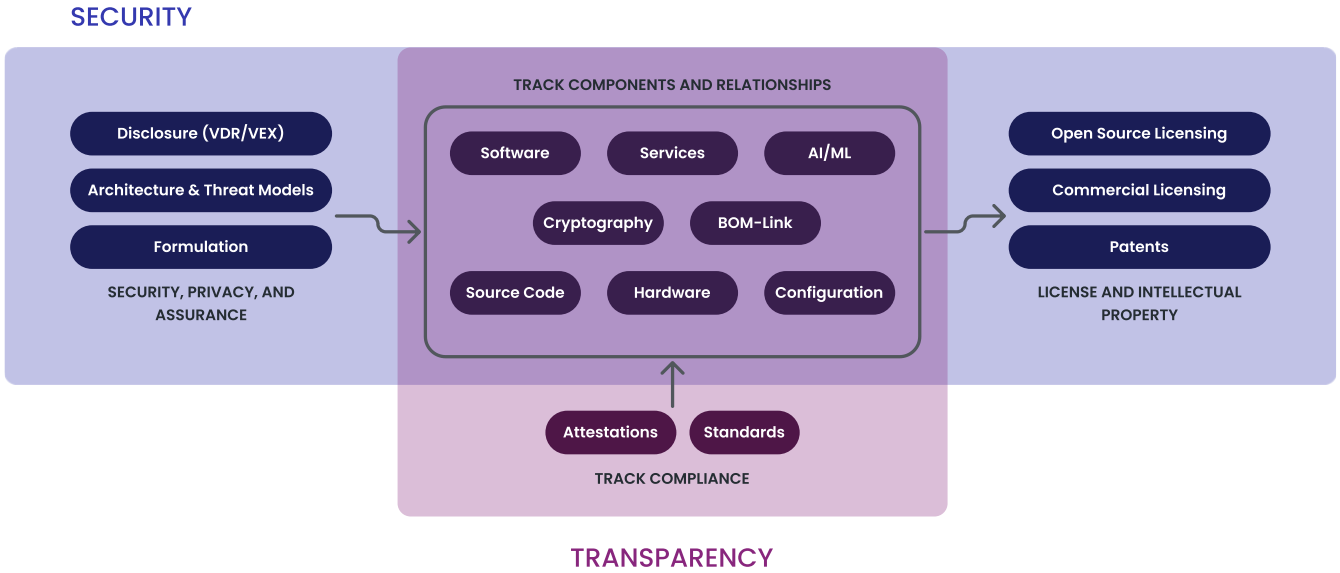
Hybrid & PQC
Is it ECC alone, or a hybrid with ML-KEM?

Protocols, keys and certificates
Are the primary interfaces for cryptography

Beyond source code
Runtime, Hardware, Services



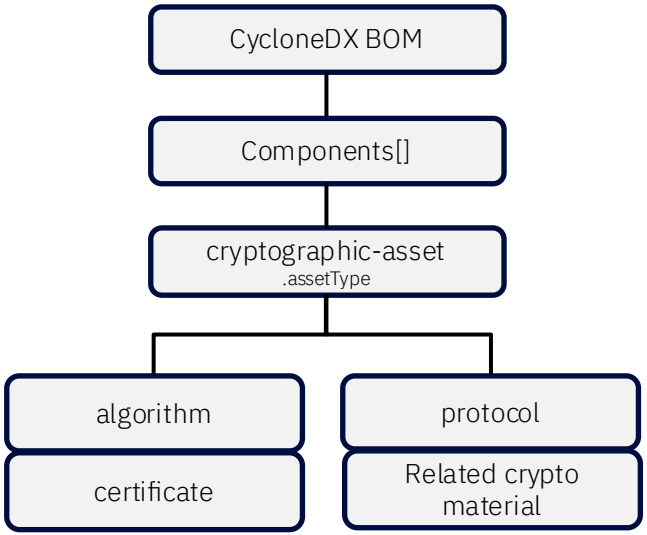
CycloneDX – Data Exchange format and specification for BOM



https://cyclonedx.org

Ecma-424

Anatomy: the cryptographic asset schema



+ evidence

occurrences[]
methods · confidence ∈ [0,1]

Every finding has a source.

algorithmProperties

primitive	block-cipher, hash, KEM, MAC, signature,…
algorithmFamily	AES, RSA, ML-KEM, SLH-DSA, ML-DSA,…
securityProperties	IND-CPA, IND-CCA2, …
cryptoFunctions	Keygen, encapsulate, decapsulate, sign,…
executionEnvironment	software-plain-ram · hardware · TEE
nistQuantumSecurityLevel	0-5

protocolProperties

type	tls, ssh, ipsec, ike, quic,…
version	"1.3"
cipherSuites[]	TLS_AES_256_GCM_SHA384 · 0x13,0x02
ikev2Transforms[]	ENCR · PRF · INTEG · KE
relatedCryptoAssets[]	refs to constituent algorithms

certificateProperties

subjectName / issuerName	
notValidBefore / notValidAfter	
serialNumber · fingerprint	
certificateFormat	X.509, CVC
lifecycleState	pre-activation → active → suspended → deactivated → revoked → destroyed
certificateExtensions[]	KeyUsage · SAN · BasicConstraints
relatedCryptoAssets[]	→ signature algorithm, public key

relatedCryptoMaterialProperties

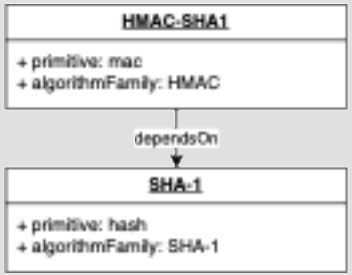
type	private-key, public-key, secret-key, token,…
state	active · suspended · compromised · destroyed
size · format	2048 · PEM · DER · P8
securedBy	{ mechanism: hsm, algorithmRef: aes-256-kw }
activationDate, expirationDate	
usage[]	sign · verify · encrypt · decrypt
fingerprint	

- Not a flat inventory, but an object model with dependencies and evidence

Dependencies: (1) Crypto Constructions

SHA-1 is broken…

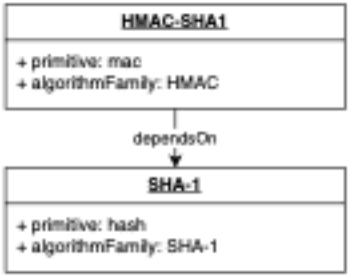
- HMAC-SHA-1
- Self-signed root certificates



Dependencies: (2) Hybrid PQC

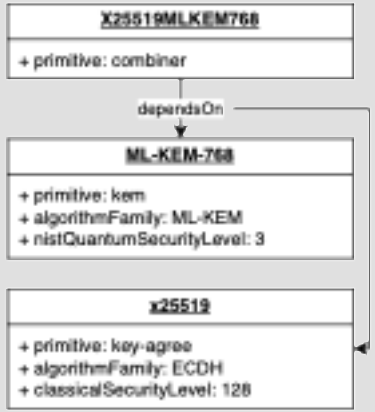
SHA-1 is broken…

- HMAC-SHA-1
- Self-signed root certificates



ECC is not quantum safe…

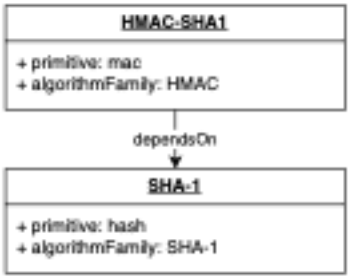
- Hybrids / combiners
- Using ECC + PQC



Dependencies: (3) Applications and Libraries

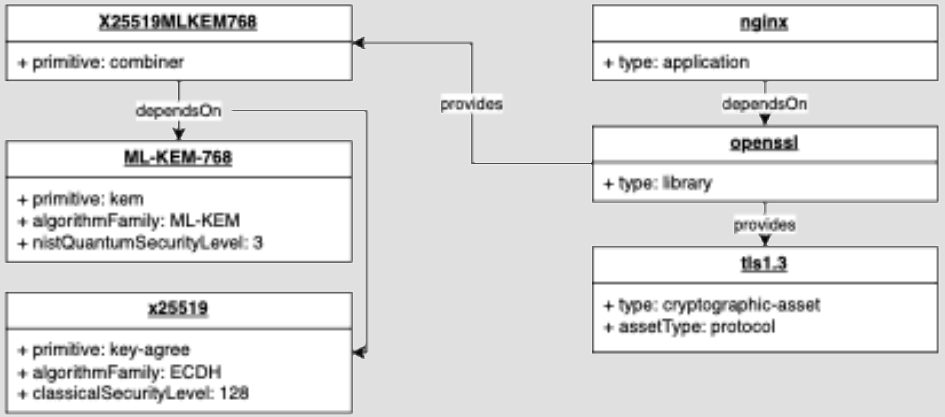
SHA-1 is broken…

- HMAC-SHA-1
- Self-signed root certificates



ECC is not quantum safe…

- Hybrids / combiners
- Using ECC + PQC



Cryptography is provided by

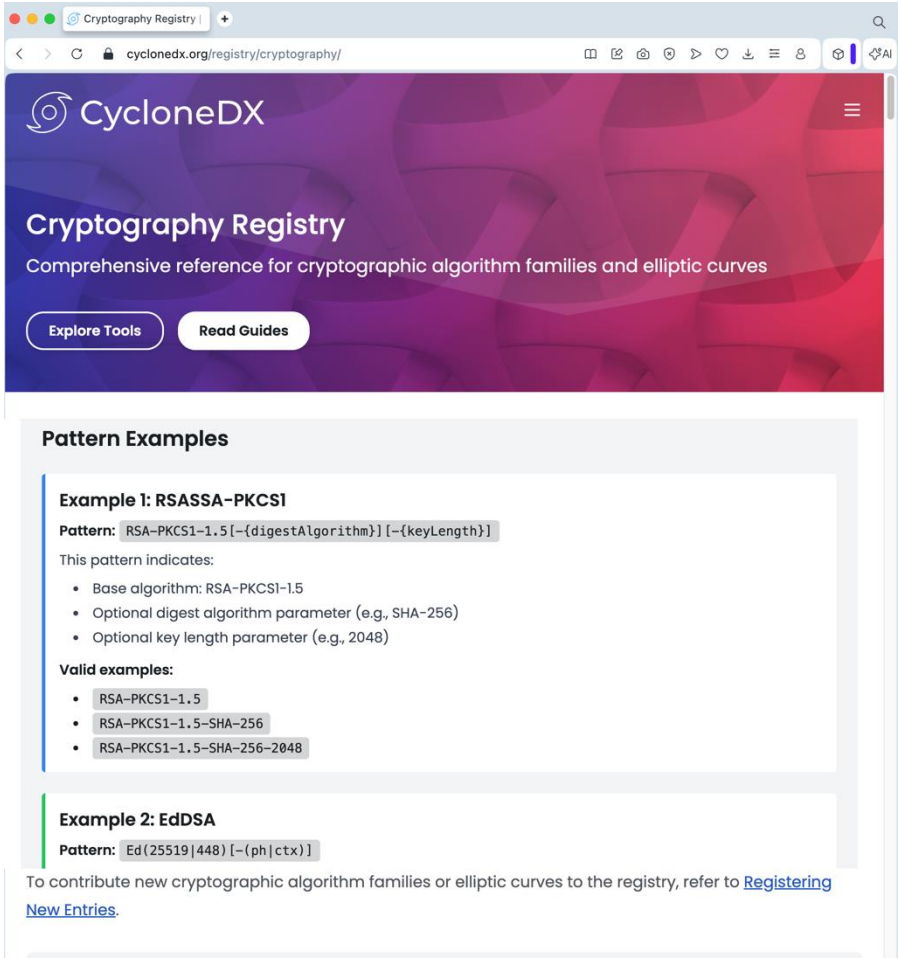
libraries, used by applications or services

Resolving naming ambiguities

Multiple naming conventions for the same algorithms, many different flavours

Found	Caveats
Triple-DES	Also known as: DESede, 3DES
Diffie-Hellman	FFDH or ECDH, which elliptic curve
RSA	Signature, PKE or KEM? RSAES OAEP or PKCS#1.5 Or RSASSA-PSS, but which digest, salt and key length
ML-DSA	Pure, externalMu or HashML-DSA, which parameter set?

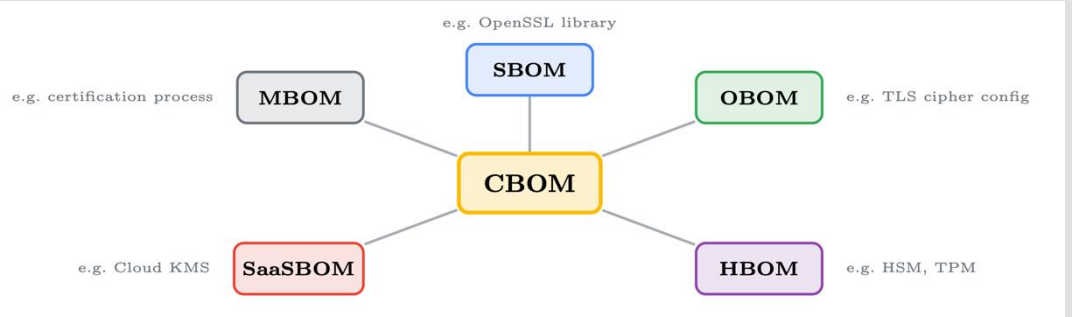
- Unique naming for CBOMs



Building blocks of a CBOM: The xBOM playbook

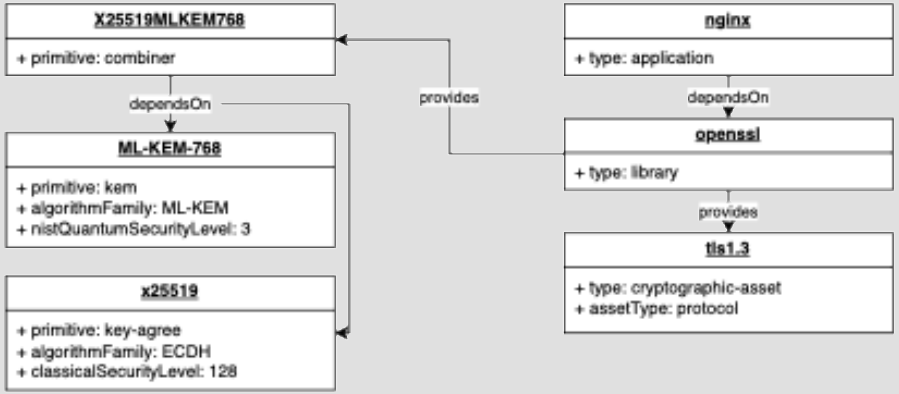
The “x” in xBOM simply defines the context that a CBOM describes

- SBOM: Software
- OBOM: Operations / Configuration
- HBOM: Hardware
- SaaSBOM: Software-as-as-service
- MBOM: Manufacturing



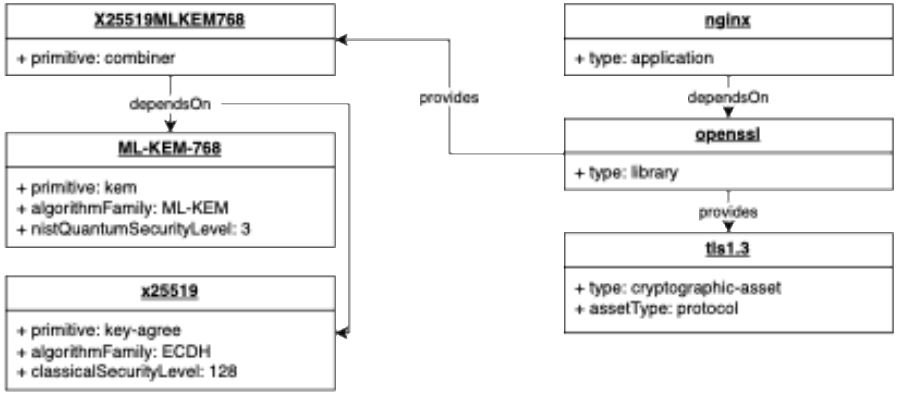
Software BOM (SBOM)

- Captures software as-built
- A CBOM lists the cryptographic capability
- Can be (mostly) derived by analyzing source code



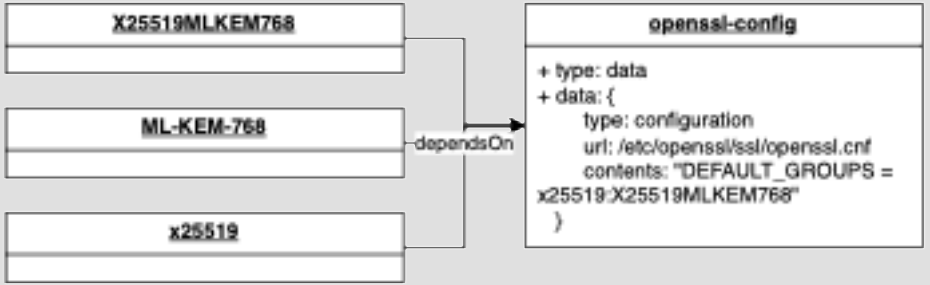
Software BOM (SBOM)

- Captures software as-built
- A CBOM lists the cryptographic capability
- Can be (mostly) derived by analyzing source code

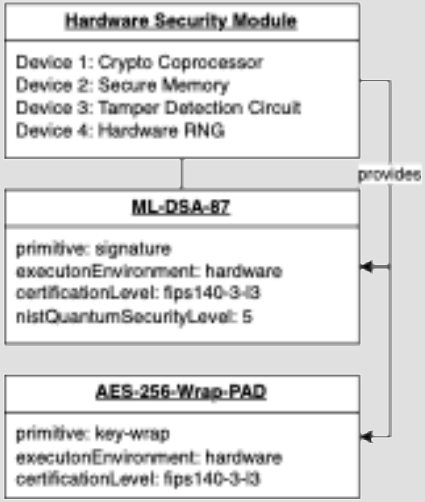


Operations BOM (OBOM)

- Captures deployment and runtime configs
- A CBOM defines the cryptography with an active state
- Examples: OpenSSL config file enabling hybrid (PQC/classical) KEM

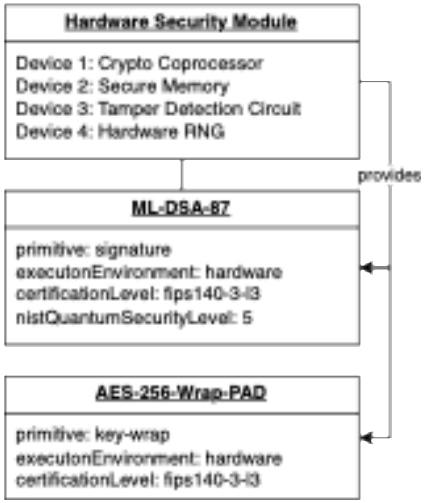


Hardware BOM (HBOM)



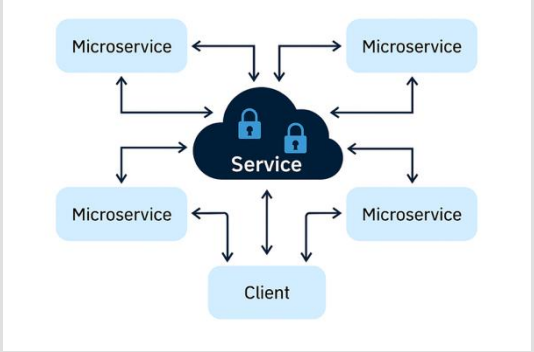
- HBOM models physical components linked to CBOM
- Hardware devices provide cryptography, algorithms, keys

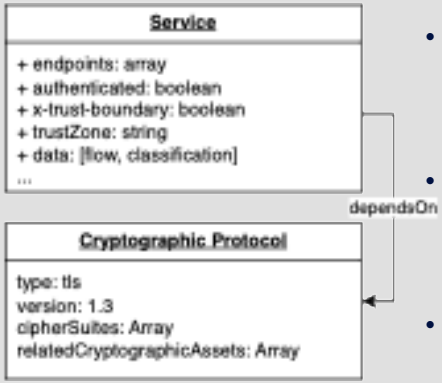
Hardware BOM (HBOM)



- HBOM models physical components linked to CBOM
- Hardware devices provide cryptography, algorithms, keys

and **SaaSBOM**





- Model endpoints, data flows and classifications
- Associate protocols, certificates and key material to services
- Helps with compliance and threat modelling

CBOM: Use Cases at a Glance

Policy Compliance

CEL expressions over CBOM components, evidence and occurrences.

```
// SHA-1 only on self-signed roots
components.all(c,
c.name == "SHA-1" ?
components.exists(cert,
cert.subject == cert.issuer
&& "CA:TRUE" in ext)
: true)
```

Policies: prohibited algorithms, min key size, PQC readiness, scope by evidence (e.g. exclude /test).

Output:
component-level pass/fail with evidence

CI / CD Pipelines

PASS deploy **FAIL: block**

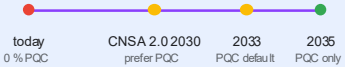
- **Build-time generation**
  CBOM as a build artifact.
- **Gate enforcement**
  Block prohibited crypto regressions.
- **Drift detection**
  Diff CBOMs across builds.
- **Dependency updates**
  Reveal added / removed algorithms.

Certificate Lifecycle

CycloneDX 1.7 + NIST SP 800-57 states

- **Expiration windows**
  e.g. expiring in < 30 days.
- **CA/B Forum 47-day rule**
  phased in 2026 - 2029.
- **Algorithm deprecation**
  find SHA-1 / RSA signers.
- **CA trust changes**
  query affected certificates.
- **Key compromise response**
  trace via dependency graph.

PQC Migration Planning



- **Quantum-vulnerable inventory**
  e.g., nistQuantumSecurityLevel = 0
- **HNDL prioritization**
  Data lifetime and exposure
- **Dependencies**
- **Agility tracking**

Evaluation: Source Code and Container Image

Two different lifecycle phases

Keycloak - source code

github.com/keycloak/keycloak (tag 26.5.1, Java)

Java sources (JCA / BouncyCastle) → **CBOMkit-hyperion** (static analysis) → **CBOM**

Captures

- Algorithms, keys, protocols (e.g., TLS)
- Compound schemes via dependsOn (e.g. SHA512withRSA)
- File + line evidence per occurrence

nginx - container image

docker.io/nginx:1.29.5

filesystem (configs + .pem files) → **CBOMkit-theia** (artifact analysis) → **CBOM**

Captures

- Trust-store certificates, public keys (RSA / EC)
- Signature schemes decomposed into primitives
- 609 evidence locations under /etc/ssl/certs/

CBOM and dependency summary (deduplicated, raw in parentheses)

Target	Analysis	Assets	Algorithms	Protocols	Certificates	Keys	Deps
Keycloak	source code	56	27	1	0	28	37
nginx	container	314 (8 104)	14 (5 400)	0	149 (1 350)	151 (1 354)	2 700

Deduplication keys: algorithms (name, algorithmProperties, OID) - certificates (subject, issuer, validity) - material (name, key value)

Evaluation

Findings

1 Expired certificate in trust store

nginx - one expired root CA

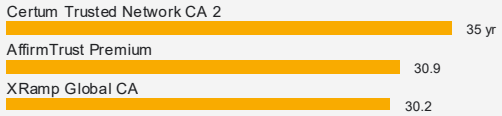
Baltimore CyberTrust Root

notValidAfter: 2025-05-12

/etc/ssl/certs/Baltimore_CyberTrust_Root.pem

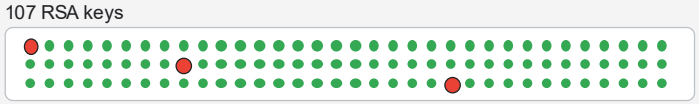
2 Decade-spanning certificates

PQ-relevant if CRQCs arrive before expiry



3 Weak RSA exponents

3 of 107 RSA keys with e < 65 537 (FIPS 186-5)



4 Legacy algorithms with dependency context

Keycloak - flagged via dependsOn graph

CBOM Generation Challenges Today

1. Detection accuracy

False positives (test fixtures) and false negatives
(reflection, dynamic loading, stripped binaries).

Mitigation: evidence-based filtering

```
CEL: exclude evidence.occurrences
where path matches "src/test/.*"
```

2. API abstractions

Cloud KMS / internal services hide server-side
algorithms from static analysis.

Vendor must publish its CBOM (SaaSBOM).

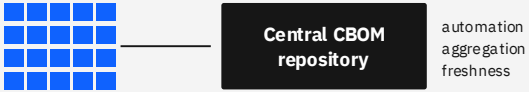
3. Transitive dependencies

Modern apps pull in hundreds of libraries
Cryptography may sit five hops down the dependency tree.

Recursive analysis needed. Depends on availability.

4. Scale & governance

Thousands of apps, hundreds of teams, multi-cloud.
A stale inventory gives false confidence.



Future directions for CBOM

1. Crypto Agility

Capture replaceability per asset, from compile-time fixed to live swap.

hardcoded	recompile
vendor-locked	vendor roadmap
firmware update	scheduled
configurable	restart
hot-swap	no downtime

Combine with risk modeling: currentState → targetState, deadline, effort.

2. Behavioral modeling (BoB)

CycloneDX 2.0 Bill of Behaviors: captures intent

data hashing appropriate — **SHA-3** — **password hashing** use Argon2 / scrypt

Behavioral attributes per asset

`purpose` integrity, availability
`performs` password-authenticated key exchange, derives ephemeral session keys from a password

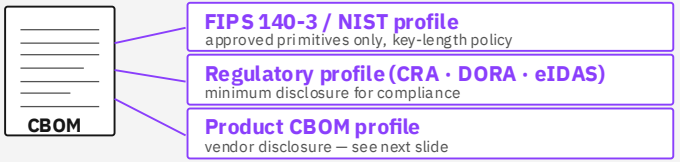
3. Implementation Assurance

Same algorithm, different security. Link evidence.

- **Side-channel resistance**
  Threat model and evaluation
- **Testing**
  Language, constant-time, sanitizers
- **Formal verification**
  EasyCrypt · CBMC

4. Perspectives & Profiles

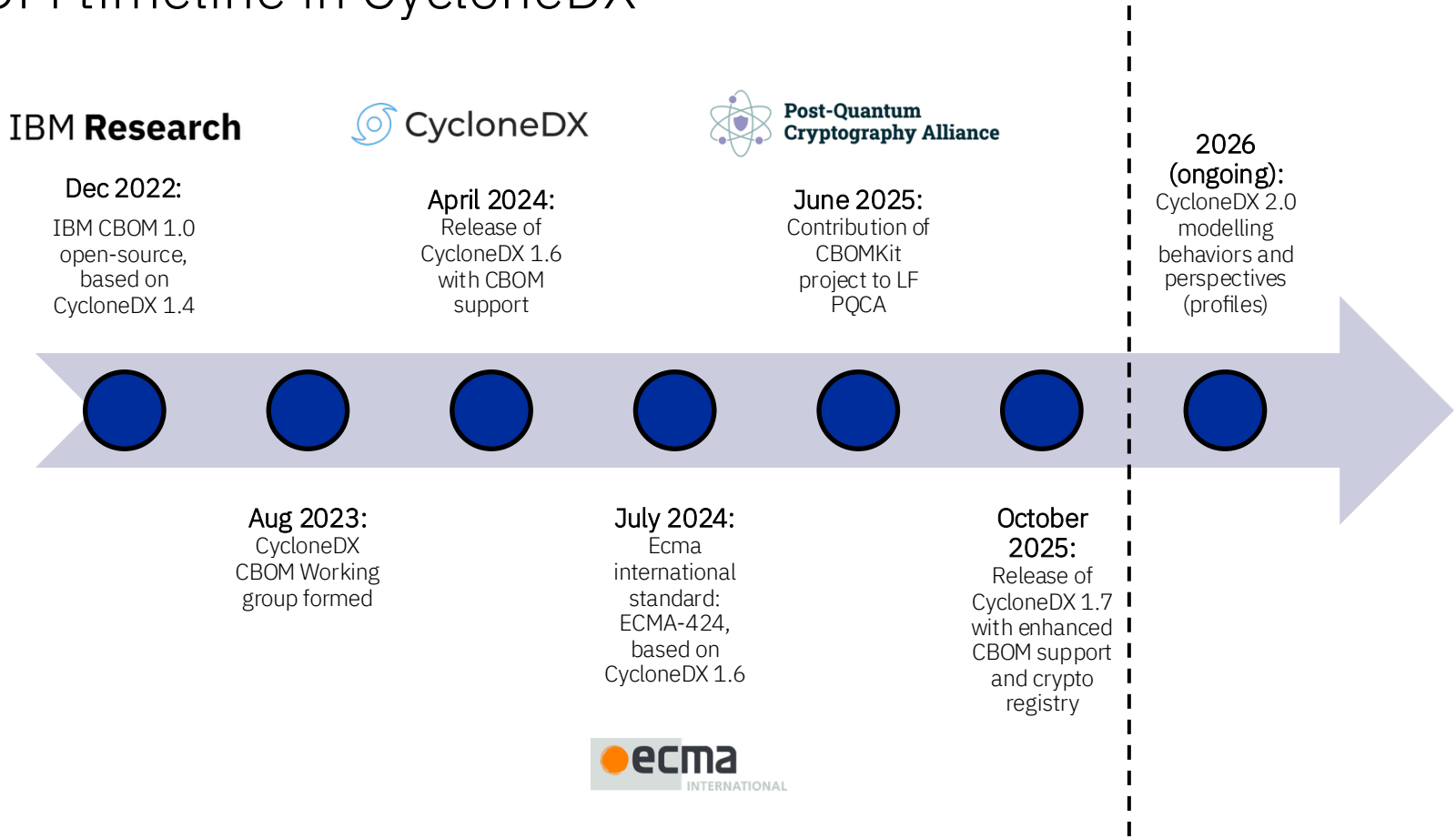
CDX 2.0 Perspectives: formal definitions of a CBOM for a consumer-specific use.



Perspective mappings: filtering and mapping to use-case language

Standards interoperability: SPDX profiles

CBOM timeline in CycloneDX



Why CBOMs, and why now?

- Cryptography is everywhere: code, configuration, certificates, services, hardware
- Calls for transparency
  - OMB M-23-02

    > *… software or hardware implementation of one or more cryptographic algorithms that provide one or more of the following services: (1) creation and exchange of encryption keys; (2) encrypted connections; or (3) creation and validation of digital signatures.*

  - EU Roadmap for the Transition to Post-Quantum Cryptography

    > *Member States should promote and support that useful cryptographic inventories are being created and maintained… Using a* ***standardised format*** *for a cryptographic inventory,* ***like CBOM*** *(Cryptographic Bill of Materials, an extension of the SBOM standard), is recommended.*

  - EU regulations (DORA, CRA, NIS2) mandate technical documentation for secure software and cryptography
- Interoperability matters: need for a standardized CBOM format, enabling interchangeability, automation and trust across the supply chain

THANK YOU

Basil Hess
bhe@zurich.ibm.com

Paper: *The Anatomy of Cryptography Bills of Materials: Standardization and Practice in CycloneDX*
Spec: *ECMA-424, cyclonedx.org*
Tooling: *https://pqca.org/projects/cbomkit*