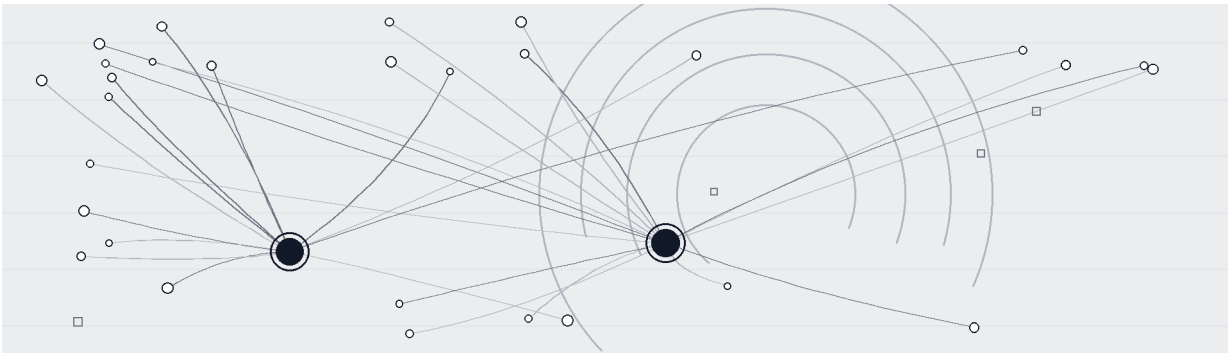
SEPTEMBER 2026

APPLIED QUANTUM

THE APPLIED QUANTUM CBOM PROFILE



A property taxonomy for cryptographic bills of materials, serving the PQC Migration Framework and Cryptographic Concentration Framework

Version 1.0-RC – September 2026

Marin Ivezic and Steve Vaile, Applied Quantum

Disclosure: Applied Quantum provides post-quantum migration advisory services. This work is published openly under CC BY 4.0 and is computable without engaging Applied Quantum.

"Switching vendors moves the contract, not the dependency."

COPYRIGHT AND LICENSE

DISCLAIMER

This document is provided as professional guidance based on the authors' and Applied Quantum's practitioner experience and publicly available standards, regulations and industry research. It does not constitute legal, regulatory, financial or compliance advice. Organisations should consult qualified legal counsel, auditors and regulatory authorities for guidance specific to their jurisdiction, sector and circumstances.

The regulatory instruments, certification records, standards and vendor capabilities referenced in this document reflect the state of the landscape as of 8 September 2026, and that landscape moves. Readers should verify current status against primary sources (regulators and supervisory authorities, NIST, EMVCo and PCI SSC, the CycloneDX specification, public certification listings, vendor documentation) before making decisions.

References to specific vendors, products or tools are for illustrative purposes and do not constitute endorsement.

This is a v1.0 release candidate, published for review and comment; v1.0 final follows the CCF pilot window in November 2026. Corrections are published as dated entries.

ABOUT THIS DOCUMENT

Document type	Property taxonomy specification
Parent document	None – referenced by the Applied Quantum PQC Migration Framework and the Cryptographic Concentration Framework, and owned by neither
Intended audience	CBOM producers and consumers, tooling vendors, and the assessment teams of both frameworks
Assumed knowledge	CycloneDX 1.6/1.7 and its cryptographic bill of materials model
Scope	A documented, openly licensed field set extending CycloneDX for post-quantum migration governance, financial-sector context and cryptographic concentration measurement.

Status	Version 1.0-RC – release candidate, published for review and comment; corrected edition of 9 September 2026; v1.0 final November 2026

VERSION HISTORY

Version	Date	Changes
1.0-RC	August 2026	First public release, as a release candidate.
1.0-RC	9 September 2026	Corrected edition, corrections C-1 to C-20 of the dated correction record: registry status in Section 1; CycloneDX 1.7 certificate references and 1.7-only fields in Section 4; encoding, multiplicity, carrier and tier-numbering rules; compensatoryControls single-valued; NIST IR 8547 milestones; disclosureStatus value deflected; observationWindow format; assets on several paths; worked fragment rebuilt as three records; day-month-year dates. Policy, tiers and field names unchanged.

HOW TO USE THIS DOCUMENT

This document is referenced by the Applied Quantum PQC Migration Framework and the Cryptographic Concentration Framework, and owned by neither. Both cite a minimum version of it: producers implement the fields, consumers state which fields they require and at which conformance level.

ACCOMPANYING RESOURCES

The Profile publishes at CBOMProfile.org, with its vocabulary registry and validation tooling on the Applied Quantum GitHub. The Cryptographic Concentration Framework and its CBOM Conformance Statement – which states the fields CCF consumes – publish at CCFramework.org; the PQC Migration Framework publishes at PQCFramework.org. Practitioner background sits on PostQuantum.com, as further reading rather than a normative source.

TABLE OF CONTENTS

A documented, openly licensed field set extending CycloneDX for post-quantum migration governance, financial-sector context, and cryptographic concentration measurement.

Referenced by the Applied Quantum PQC Migration Framework and the Cryptographic Concentration Framework. Owned by neither. Both cite a minimum version of this document.

---

1. STATUS

This profile is a **property taxonomy layered on CycloneDX**, not a competing specification.

CycloneDX provides the standardised base model this profile is built on. Published as ECMA-424, first edition June 2024 covering v1.6, with v1.7 released October 2025 and adopted as ECMA-424 second edition in December 2025. Its cryptoProperties model provides the core representation for cryptographic assets this profile requires, and the profile references those fields rather than restating them.

What CycloneDX deliberately does not model is domain-specific metadata: migration governance state, financial-sector context, and the inputs to a concentration computation. The specification provides an extension mechanism for exactly this – a vendor property namespace, with an optional public taxonomy registration. Registered namespaces already include those held by IBM, Siemens, NVIDIA, GitLab, Amazon, and Germany's Federal Office for Information Security, whose bsi namespace maps to its own TR-03183 technical guideline.

This profile registers the appliedquantum namespace on the same basis. The public taxonomy is maintained at github.com/appliedquantum/cyclonedx-property-taxonomy, and the top-level registration was filed with the CycloneDX property-taxonomy registry as issue #189 on 14 August 2026. As of 8 September 2026, the issue is open and the published registry table does not yet carry the namespace.

Review. This release candidate is published for comment through 31 October 2026. File comments as issues on the taxonomy repository. Section 12 states what v1.0 final adds beyond this text.

Standing. A registered namespace is a stable extension mechanism for domain-specific fields, governed by the CycloneDX property-taxonomy process. The core specification does not absorb sector-specific fields such as payment system context or HSM firmware family, by design. A subset of this profile is separately proposed for the core specification, and Section 9 records which fields and why. That is a maturity path for part of the taxonomy, not a provisional status for the whole.

2. DEPRECATION AND MIGRATION POLICY

Adoption requires knowing what happens if a field moves. This policy is the answer, and it is binding.

Where a field in this profile is accepted into the CycloneDX core specification:

1. The appliedquantum field is marked deprecated in the release following acceptance.
2. This profile publishes a normative one-to-one mapping from the deprecated field to its core equivalent, including any enumeration translation.
3. The deprecated field remains valid and supported for **no fewer than two subsequent versions** of this profile.
4. Tooling consuming this profile must accept both forms during the deprecation window.
5. No field is removed without a major version increment.

Where the core specification adopts a field with different semantics rather than an equivalent, the appliedquantum field is retained and the difference documented. Silent semantic drift is the failure mode this clause exists to prevent.

3. STRUCTURE

Four tiers. The first three correspond to the model Applied Quantum has been recommending to financial-sector clients. The fourth supports concentration measurement.

Tier	Content	Namespace	Serves
1	CycloneDX core cryptographic fields	native cryptoProperties	Both frameworks
2	PQC migration governance	appliedquantum:pqc	PQC Migration Framework
3	Sector context	appliedquantum:fs, appliedquantum:custody	Both
4	Concentration and dependency-resolution data	appliedquantum:conc, :pki, :path, :entropy, :lineage	CCF

Tiers are additive and independently adoptable. An institution running a PQC migration adopts Tiers 1 to 3. One computing concentration adds Tier 4. Neither requires the other.

One exception is declared. CCF conformance level 1 requires pqc:vulnerabilityStatus from Tier 2, because layer 1 scoping is defined on the Shor and Grover classification rather than on the raw security level. No other Tier 2 field is required by CCF. Tier numbers are not maturity grades, and they are distinct from the CCF conformance levels CCF-1 to CCF-3, which the CCF CBOM Conformance Statement defines in terms of the fields it consumes.

Property names follow the CycloneDX convention namespace:subnamespace:name – lowercase namespaces with lower camelCase leaf names.

Three properties are multi-valued: conc:serviceRef, pqc:regulatoryMapping and pki:crossSignedBy. A multi-valued property is carried as repeated properties, one value per occurrence. Duplicate identical values are invalid, and order carries no meaning. Every other property in this profile is single-valued. Single-valued means one occurrence per properties array; the same property may appear on several distinct components. Every value is a string. A boolean is carried as true or false; a number as a decimal string within the stated range, without a per cent sign; a date as an ISO 8601 calendar date. Where a value is unknown, the property is omitted unless its enumeration defines an unknown value; an empty string, a null or a placeholder is invalid.

4. TIER 1 – CYCLONEDX CORE

Referenced, not restated. See the CycloneDX specification and ECMA-424.

Required for both frameworks: cryptoProperties.assetType; algorithmProperties.primitive, .parameterSetIdentifier, .ellipticCurve, .mode, .cryptoFunctions, .classicalSecurityLevel, .nistQuantumSecurityLevel; cryptoProperties.oid; certificateProperties.subjectName, .issuerName, .serialNumber, .signatureAlgorithmRef, .notValidBefore, .notValidAfter; protocolProperties.type, .version, .cipherSuites; relatedCryptoMaterialProperties.type, .state, .securedBy; the dependencies array using ref, dependsOn and provides; and evidence.occurrences[].location.

Version notes. This profile applies to CycloneDX 1.6 and 1.7 documents; a field introduced in 1.7 is not required in a 1.6 document. ellipticCurve is the CycloneDX 1.7 field, and consumers accept the deprecated curve for 1.6 compatibility. signatureAlgorithmRef is a bom-ref string referencing the algorithm asset, not an algorithm name; CycloneDX 1.7 deprecates it and subjectPublicKeyRef in favour of relatedCryptographicAssets, and consumers accept either form. In relatedCryptographicAssets, the entry with type algorithm carries the certificate-signature algorithm and the entry with type publicKey carries the subject public key, as

in the specification's published 1.7 examples; a producer converting a 1.6 document preserves both roles. certificateProperties.serialNumber, .fingerprint and .certificateState are 1.7 fields. The dependency object carries no type field, and provides expresses that a component supplies a cryptographic capability, which is the relationship this profile relies on to tie implementations to algorithms.

Include `oid` wherever an applicable standardised identifier exists. Where none does, record the canonical identifier defined by the relevant standard or registry. The same curve appears as prime256v1, secp256r1 and P-256 across tools. Any counting exercise that resolves one curve to three dependencies through naming alone produces a wrong answer.

Record the certificate-signature algorithm, via `signatureAlgorithmRef` or its 1.7 relation, as distinct from the subject public key. For certificate authenticity, quantum vulnerability is principally determined by the certificate-signature scheme. The subject public key may carry its own quantum exposure independently, and both are recorded.

CycloneDX 1.7 additionally provides algorithmFamily, a standardised elliptic curve list, and the Cryptography Registry. Use 1.7 where tooling supports it.

5. TIER 2 – PQC MIGRATION GOVERNANCE

appliedquantum:pqc:*

These are populated through risk assessment, stakeholder engagement and ongoing governance. No discovery tool produces them.

Property	Values	Purpose
vulnerabilityStatus	vulnerable-shor · weakened-grover · not-known-quantum-vulnerable · unknown	Classification driving migration priority under this profile's quantum threat model. not-known-quantum-vulnerable asserts only that the algorithm is not classified as vulnerable under that model; it is not a general security claim, and the classification is not a complete taxonomy of cryptanalytic strength.
migrationStatus	not-started · planned · in-progress · hybrid-deployed · pqc-only · not-applicable	Programme state at the endpoint. hybrid-deployed and pqc-only describe

Property	Values	Purpose
		capability enabled; what any path actually negotiates is read from the path record, which is authoritative for layer 5 and for coverage.
disallowedAfter	ISO 8601 date	The date this algorithm becomes disallowed under the governing regime.
horizonSource	nist-ir-8547 · cnsa-2.0 · us-eo-14412 · uk-ncsc · eu-roadmap · sector · internal	Which regime sets that date. Record alongside it the source's status (final, draft, or guidance), jurisdiction and as-of date; an institution-approved horizon takes precedence over any default, and a draft milestone is never presented as a binding requirement. Horizons may differ by primitive under one regime – Executive Order 14412 sets 31 December 2030 for key establishment and 31 December 2031 for digital signatures – and disallowedAfter is recorded per asset, so the split needs no additional structure.
dataClassification	institution-defined	Sensitivity of data protected, driving harvest exposure assessment.
hndlExposure	yes · no · unknown · not-applicable	Whether protected data is subject to harvest-now-decrypt-later risk. A governance determination, not a discovered fact; unknown is a legitimate value and is never defaulted to no.

Property	Values	Purpose
dataRetentionPeriod	ISO 8601 duration	How long the data must remain confidential. With disallowedAfter, an input to the migration urgency calculation.
systemOwner	identifier	Accountable party for migration decisions.
vendorDependency	boolean	Whether migration is self-controlled or waits on a supplier.
vendorReadinessDate	ISO 8601 date	Supplier's stated date for a post-quantum capable release.
vendorReadinessStatus	committed · forecast · announced · unknown	How firm that date is. Roadmap dates are frequently non-binding, and a date alone implies more certainty than exists.
compensatoryControls	free text, single-valued	Interim mitigations such as tunnel overlay or double encryption. Structured control identifiers, where used, are carried inside the text or under the institution's own namespace; this property is not repeated.
regulatoryMapping	requirement identifier, multi-valued per Section 3	Requirements this usage satisfies, e.g. PCI-DSS-12.3.3. One identifier per property, repeated per requirement.
cmdbRef	identifier	Link to the enterprise configuration record.

Anchoring. vulnerabilityStatus is intended to be read alongside the native nistQuantumSecurityLevel rather than replacing it, and disallowedAfter defaults to the

NIST IR 8547 transition milestones where horizonSource is nist-ir-8547. That report remains an initial public draft (November 2024). It deprecates quantum-vulnerable public-key algorithms at 112 bits of security strength after 2030 and disallows all quantum-vulnerable public-key algorithms after 2035. Anchoring the enumeration to a published regime rather than inventing a parallel vocabulary is what makes this mappable if a standards body later publishes its own.

6. TIER 3 – SECTOR AND CUSTODY CONTEXT

6.1 Key custody – `appliedquantum:custody:*`

Applies to any sector operating hardware security modules.

Property	Values	Purpose
platform	product name	The visible unit.
firmwareFamily	supplier-issued canonical family identifier	The dependency unit. Two products sharing the relevant firmware family are treated as one dependency where the failure mode under assessment propagates through that firmware. Use the supplier's or manufacturer's canonical family identifier; never derive family identity from marketing names.
oemOrigin	manufacturer	Captures rebadging. A module sold under three names from one manufacturer is one dependency, and nothing else in the record reveals it.
certificate	FIPS 140-3 or PCI HSM certificate reference	Corroborates the family, and where a security policy exists, links to implementation ancestry.

Firmware-family identifiers are collected in the controlled-vocabulary registry in the taxonomy repository, extended by issue.

Cloud services. Resolve to the underlying module where disclosed. Where it is not, record the cloud provider as the visible dependency boundary and mark the underlying manufacturer as undisclosed. The provider is a boundary, not the manufacturer, and representing it as the manufacturer would corrupt the field's meaning.

6.2 Financial services – `appliedquantum:fs:*`

Property	Values	Purpose
paymentContext	swift · card-network · rtgs · ach · instant · correspondent · securities · none	Determines ecosystem dependencies and coordination requirements.
crossBorder	boolean	Whether the usage involves counterparties in other jurisdictions.
interopConstraint	boolean	Whether changing the algorithm requires coordination with external parties.

7. TIER 4 – CONCENTRATION AND DEPENDENCY-RESOLUTION DATA

7.1 Implementation ancestry – use `pedigree`, not a new field

CycloneDX already has the right structure and it is almost never populated.

The pedigree object carries ancestors, descendants, variants, commits, patches and notes, and the specification describes it as intended for forks where the forked version contains an ancestor node referencing the original. That is precisely the semantic required to record that a deployed ML-KEM implementation descends from mlkem-native, which itself descends from the pq-crystals reference.

This profile requires `pedigree.ancestors` to be populated for cryptographic implementation components where ancestry is known or disclosed. Where it is not, the array is left empty and lineage:disclosureStatus records why, per the absence rule at the end of this section. Each ancestor is a component entry identifying the upstream project and version – and, where obtainable, the commit or content hash, build provenance, and the relevant code path. Project-name-only identification is graded as inferred.

Chains nest. They do not flatten. ancestors is a set, and its order carries no meaning. Within this profile, multiple ancestors at one pedigree level SHALL be interpreted as multiple direct upstream sources – a merge – and a derivation chain SHALL be expressed by nesting, since each ancestor is itself a full component that may carry its own pedigree. The specification's own example uses a single ancestor per node and describes documenting "all forks, and their forks," which is recursion. The SHALL language here makes the interpretation a profile rule rather than a claim about how every CycloneDX consumer behaves.

Both forms occur in cryptographic implementations and the profile requires them to be distinguished:

Chain. A deployed implementation derives from mlkem-native, which derives from the pq-crystals reference. One ancestor at each level, nested.

Merge. AWS-LC is described by its maintainers as based on code from both BoringSSL and OpenSSL. Two ancestors at one level, flat, and correctly so.

Flattening a chain misrepresents a merge, and the two have different propagation properties. In a merge the component inherits from both parents directly. In a chain, depth matters. A defect in an immediate ancestor almost certainly reaches the descendant, while one several hops up may not, because the relevant code may have been rewritten in between. Layer 2 resolution depends on being able to tell these apart.

Record divergence in `commits` and `patches` where available. The specification defines both as describing how a component deviates from an ancestor, and they are the preferred evidence form. Closed-source implementations may be unable to supply commit history. Reproducible source comparison, supplier engineering attestation of the rewritten primitives, or validated source provenance are admissible in its place. Bare assertion is not admissible. A divergence claim must point at some evidence of the deviation.

Use `variants` where direction is unknown. The specification provides variants for components that contain nearly identical code where it is unclear whether one derives from the other or both share a common ancestor. That is a better fit than an ancestry claim for two implementations that look the same and cannot be resolved, and it should be used in preference to guessing a direction.

`pedigree` records derivation only. A library that a product merely uses, bundles or obtains from a provider is expressed through dependencies and provides, never as an ancestor. "Descends from" and "depends on" have different defect-propagation semantics, and conflating them invents ancestry. The assessment layer resolves both relationship types into failure domains, and the record keeps them distinct.

Using the existing mechanism is a stronger position than inventing a field. It demonstrates an intended-but-dormant capability rather than requesting a new one, and any tooling that later populates pedigree becomes immediately useful.

What pedigree does not carry, and this profile supplies:

appliedquantum:lineage:*

Property	Values	Purpose
evidence	E1–E5 per Section 8	How the ancestry was established.
disclosureStatus	disclosed · declined · deflected · pending · no-response · not-requested	Whether the supplier answered. declined records an explicit refusal; deflected records a reply received that does not answer the request; pending becomes no-response at the stated deadline. The states are distinguished because refusal, deflection and silence have different supplier-governance meanings. A partial disclosure is recorded as disclosed; which claims were answered is recorded by the consumer, not in this property.
cmvpCertificate	CMVP certificate number	May provide corroborating evidence of implementation provenance where the module's published security policy identifies the relevant software or libraries. A certificate is not itself lineage evidence; the security policy sometimes is.

Where ancestry is undisclosed, leave pedigree.ancestors empty and set disclosureStatus accordingly. An empty ancestors array with disclosureStatus disclosed records an implementation whose supplier has disclosed that it has no upstream source within the

stated scope. Recording the absence accurately matters more than guessing. Fabricating an ancestor to avoid an awkward record defeats the measurement entirely.

7.2 Trust root – `appliedquantum:pki:*`

Property	Values	Purpose
trustAnchorRef	bom-ref	The terminating root actually relied upon. The dependency unit is the anchor, not the issuing CA, and an issuer name alone cannot resolve a chain.
crossSignedBy	trust-anchor references, multi-valued per Section 3	Records cross-signing relationships. A factual representation only: cross-signing does not by itself establish a common dependency. CCF resolution determines whether compromise of a common anchor would invalidate the certification paths actually accepted, and without this field that determination cannot be made.
trustAnchorOperator	institution-operated · third-party	Distinguishes roots the institution operates from roots it merely trusts, which determines whether remediation is available. The operator's name is carried in native organisation records, not in this property.

7.3 Path and counterparty – `appliedquantum:path:*`

Paths are first-class records, not per-asset annotations. A path is recorded once, as a component keyed by pathId, holding the counterparty fields below. An asset serving one path references it by appliedquantum:path:pathId. An asset serving several paths carries no pathId and references each path record through dependsOn in the native dependencies array; pathId is never repeated on an asset.

Repeating counterparty fields on every asset is non-conformant. Assets on one path can then disagree about the counterparty, and a single counterparty state change would edit thousands of records instead of one.

Property	Carried on	Values	Purpose
pathId	asset (single-path) and path record	identifier	Coverage is a path property; assets must be groupable into paths.
negotiatedOutcome	path record	observed protocol, version and key-establishment group	cipherSuites records what is supported. This records what is negotiated, and it is the layer 5 dependency unit.
counterpartyRef	path record	identifier	Coverage counts counterparty state, not only the institution's own. Use an LEI where one exists; otherwise a scoped identifier with its issuer stated. Aggregation deduplicates by legal group.
counterpartyClass	path record	blocking · contractual · peer · population	Determines the remediation route and permits sector aggregation.
counterpartyTargetState	path record	boolean	The coverage computation input, recorded against the target-state definition set for the service in Phase 1 of the assessment.
populationTargetPct	path record	0–100	Required where class is population; the

Property	Carried on	Values	Purpose
			share of the population at target state.
counterpartyStateReason	path record	optional free text	Why a counterparty is not at target state – unsupported algorithm, not deployed, unverified, policy mismatch. false alone does not distinguish these, and the remediation route differs.
observationWindow	path record	ISO 8601 time interval in start/end form, each end a timestamp with a timezone designator, for example 2026-09-01T00:00:00Z/2026-09-08T00:00:00Z	The period over which the negotiated outcome was observed, so a modal value cannot silently hide a downgrade tail.

The boolean and percentage forms are separate properties. A single polymorphic field would break schema validation and conceal which quantity was measured.

Two further record rules. One path-state record per distinct negotiated outcome or population segment is permitted, each stating its observationWindow. Where a path serves several services with different target states, one record per service is permitted, carrying conc:serviceRef.

7.4 Entropy – `appliedquantum:entropy:*`

CycloneDX 1.7 does not natively provide the entropy-source and key-generation-design fields this profile requires. The native primitive value drbg records that a deterministic generator exists, and nothing records the noise source or generation design behind it.

Property	Purpose
sourceDesign	The entropy dependency unit, e.g. ring-

Property	Purpose
	oscillator-array, free-running-oscillator, thermal-noise. Values are drawn from the controlled-vocabulary registry in the taxonomy repository where an entry exists, and are otherwise implementation-defined; within one assessment, assessors SHALL normalise to a single label per design so that identical designs group as one dependency. Several vendors' secure elements resolve to one design.
generationDesign	The algorithm and library design generating the key material, where distinct from the entropy source: prime-generation, nonce-generation, and derivation designs. Values are drawn from the registry where an entry exists, and are otherwise implementation-defined, normalised within an assessment as for sourceDesign. Captures the ROCA failure class.
drbgSpecification	e.g. SP800-90A-CTR_DRBG, AIS20-DRG.4. Separates the deterministic layer from the noise source.
esvCertificate	NIST Entropy Source Validation certificate number. ESV certificates are standalone, publicly listed, and referenceable by number across multiple modules; the public-use documents describe the noise source design. A certificate validates a source in stated operating environments: record whether the deployment environment matches, and treat the certificate as evidence in the chain, never as deployment proof.
evidence	E1–E5. Expect E4 or E5 across most estates.

Expect sparse population and record that honestly. No public dataset maps deployed devices to entropy designs. The evidence chain runs device to module certificate to referenced ESV certificate to public-use document, manually. A registry entry names a

design family. Two products sharing an entry share a design family; whether they share one design is a separate claim, graded through entropy:evidence.

7.5 Assessment scaffolding – `appliedquantum:conc:*`

Property	Level	Purpose
serviceRef	asset	Important business service identifier, **multi-valued** per Section 3. An asset supporting several services carries one property per service; duplicate identical values are invalid, and order carries no meaning. Matches the framework's rule that shared infrastructure is counted in full in every service it supports. Bills of materials are organised by system; concentration is computed per service. Without this mapping there is no unit to compute over.
evidenceTier	asset	E1–E5.
criticalityDerivesKeys	asset	Certifies or derives other keys. *Determination-carried: supplied by the service or key owner, not by a discovery tool.*
criticalityAuthorizesIrreversible	asset	Authorises an irreversible action or underwrites a finality guarantee.
criticalityNotRevocable	asset	Cannot be revoked within the service's impact tolerance.
criticalityPastHorizon	asset	Validity, including verification validity, extends past the migration completion date.
criticalityNoEquivalent	asset	No operational equivalent;

Property	Level	Purpose
		failure halts the service.
discoveryCoverage	service record	Decimal fraction, 0–1. The fraction of that service's cryptographic estate this bill of materials represents. The denominator is the independently enumerated estate for the service – built from the configuration record, service map and application portfolio – not the set of assets the discovery tooling happened to observe. Asset classes the discovery method cannot observe are excluded from the fraction and named in discoveryMethod as unobserved; their coverage is unknown, not zero. Recorded per service, not per document: a blended figure conceals the service most likely to hold the finding. A document-level summary may be derived, and never replaces the per-service values.
discoveryMethod	metadata	How the estate was enumerated, so a reviewer can judge the coverage claim.
assessmentDate	metadata	Concentration moves; figures need a date.

Carriers. An asset-level property is carried in the properties array of the component it describes. Implementation ancestry and the lineage properties are carried on the implementation component (Section 7.1). Path properties other than pathId are carried on the path record (Section 7.3). discoveryCoverage is carried on the native services entry for the service. discoveryMethod and assessmentDate are carried in

metadata.properties. A criticality property whose determination has not been made is omitted; omission is never read as false.

On `discoveryCoverage`. An institution that has mapped its estate poorly scores better than one that has mapped it well, because undiscovered dependencies do not appear. Publishing coverage alongside every figure is what stops improved discovery from reading as deterioration.

8. EVIDENCE TIERS

Applies wherever a property carries evidence or evidenceTier.

Tier	Basis	Example
E1	Runtime observed	Traffic capture of the negotiated key exchange
E2	Binary confirmed	Library identified in the deployed binary
E3	Inventory declared	SBOM component entry, HSM console export
E4	Vendor attested	Supplier response to a disclosure request
E5	Inferred	Reasoning from documentation or certification records

Evidence tier never changes a computation. It changes the confidence stated alongside it.

Where a namespace carries its own evidence property (lineage:evidence, entropy:evidence), it grades that namespace's values, and the asset-level conc:evidenceTier is the default only, and it MUST NOT be read as overriding a more specific property-level evidence value. All other values are graded by conc:evidenceTier. This is how the framework's per-value tier requirement is expressed in the schema.

Tiers are ordered within a claim type rather than across them. See Universal Framework A.6 for the admissible-evidence table.

9. MAPPING TO STANDARDS, AND UPSTREAMING

Every field maps to its nearest published analogue, or is marked as a gap. Gaps are the justification for this profile existing.

Field	Nearest published analogue	Status
pqc:vulnerabilityStatus	CycloneDX nistQuantumSecurityLevel; NIST IR 8547	Proposed upstream
pqc:migrationStatus	None	Gap. Proposed upstream
pqc:hndlExposure	FS-ISAC PQC risk model; Mosca inequality	Gap in schema. Proposed upstream
pqc:dataRetentionPeriod	FS-ISAC shelf-life guidance	Gap in schema. Proposed upstream with hndlExposure
pqc:disallowedAfter	NIST IR 8547 transition milestones	Anchored, retained
Implementation ancestry	CycloneDX pedigree	Uses existing mechanism
lineage:cmvpCertificate	CMVP validated module list	Proposed upstream
entropy:*	NIST SP 800-90B/C; ESV certificate list; AIS-31	Gap. Proposed upstream
custody:firmwareFamily, oemOrigin	FIPS 140-3 CMVP; PCI HSM	Gap. Retained
pki:trustAnchorRef, crossSignedBy	None	Gap. Retained
path:*	None	Gap. Retained
conc:serviceRef	PRA SS1/21, DORA important business services	Gap in schema. Retained
fs:*	None	Sector-specific. Retained indefinitely

Proposed upstream means a proposal will be filed against the CycloneDX specification. Those fields have cross-industry value: quantum vulnerability classification, migration state, harvest exposure, certificate references and entropy modelling are not financial-sector concerns. **Retained** means the field is sector-specific or computation-specific and belongs in a vendor namespace permanently, which is what vendor namespaces are for.

Proposals for the fields marked proposed upstream will be filed against the CycloneDX specification following this release candidate's publication, and filing status is tracked in the taxonomy repository.

10. WORKED FRAGMENT

A deployed ML-KEM implementation and the path it serves, as three records: the cryptographic asset, the implementation component that provides it and carries its ancestry, and the first-class path component the asset references. The document envelope shows where the document-level and service-level properties are carried. Deployment identifiers, certificates and vendors below are synthetic, with certificate references as explicit placeholders. Standardised identifiers and named upstream projects are real: the OID is ML-KEM-768's, and the ancestry shown reflects the publicly documented mlkem-native derivation from the pq-crystals reference. The fragment validates against the CycloneDX 1.7 schema and against this profile's JSON Schema, and is the kit's fixture-valid.json.


```
{
  "bomFormat": "CycloneDX",
  "specVersion": "1.7",
  "version": 1,
  "metadata": {
    "properties": [
      {
        "name": "appliedquantum:conc:discoveryMethod",
        "value": "Configuration export and TLS observation; payment-terminal embedded keys unobserved"
      },
      {
        "name": "appliedquantum:conc:assessmentDate",
        "value": "2026-09-08"
      }
    ]
  },
  "services": [
    {
      "bom-ref": "service/IBS-014-card-authorization",
      "name": "Card authorisation",
      "properties": [
        {
          "name": "appliedquantum:conc:discoveryCoverage",
          "value": "0.82"
        }
      ]
    }
  ],
  "components": [
    {
```

```
      "type": "cryptographic-asset",
      "bom-ref": "crypto/tls/acquirer-gw-01/mlkem768",
      "name": "ML-KEM-768",
      "cryptoProperties": {
        "assetType": "algorithm",
        "oid": "2.16.840.1.101.3.4.4.2",
        "algorithmProperties": {
          "primitive": "kem",
          "parameterSetIdentifier": "ML-KEM-768",
          "cryptoFunctions": [
            "keygen",
            "encapsulate",
            "decapsulate"
          ],
          "nistQuantumSecurityLevel": 3
        }
      },
      "properties": [
        {
          "name": "appliedquantum:conc:serviceRef",
          "value": "IBS-014-card-authorization"
        },
        {
          "name": "appliedquantum:conc:evidenceTier",
          "value": "E2"
        },
        {
          "name": "appliedquantum:pqc:vulnerabilityStatus",
          "value": "not-known-quantum-vulnerable"
        },
        {
          "name": "appliedquantum:pqc:migrationStatus",
          "value": "hybrid-deployed"
        },
        {
          "name": "appliedquantum:custody:platform",
          "value": "Vendor B payment HSM (synthetic)"
        },
        {
          "name": "appliedquantum:custody:firmwareFamily",
          "value": "VB-PS-7"
        },
        {
          "name": "appliedquantum:custody:oemOrigin",
          "value": "Vendor B"
        },
        {
          "name": "appliedquantum:custody:certificate",
          "value": "FIPS-140-3-EXAMPLE-001"
        },
        {
          "name": "appliedquantum:entropy:sourceDesign",
          "value": "ring-oscillator-array"
        },
        {
          "name": "appliedquantum:entropy:drbgSpecification",
          "value": "SP800-90A-CTR_DRBG"
        },
        {
          "name": "appliedquantum:entropy:esvCertificate",
          "value": "ESV-EXAMPLE-001"
        },
        {
          "name": "appliedquantum:entropy:evidence",
          "value": "E4"
```

```
        },
        {
          "name": "appliedquantum:fs:paymentContext",
          "value": "card-network"
        },
        {
          "name": "appliedquantum:fs:crossBorder",
          "value": "true"
        },
        {
          "name": "appliedquantum:fs:interopConstraint",
          "value": "true"
        },
        {
          "name": "appliedquantum:path:pathId",
          "value": "PATH-ACQ-SCHEME-02"
        }
      ]
    },
    {
      "type": "library",
      "bom-ref": "impl/acquirer-gw-01/vendor-b-kem-lib",
      "name": "Vendor B KEM library (synthetic)",
      "version": "3.2.0",
      "supplier": {
        "name": "Vendor B"
      },
      "pedigree": {
        "ancestors": [
          {
            "type": "library",
            "name": "mlkem-native",
            "version": "1.0.0",
            "supplier": {
              "name": "PQ Code Package"
            },
            "pedigree": {
              "ancestors": [
                {
                  "type": "library",
                  "name": "pq-crystals ML-KEM reference",
                  "version": "standard"
                }
              ]
            }
          }
        ],
        "notes": "Chain, not merge: the deployed implementation derives from mlkem-native, which derives from the pq-crystals reference. Ancestry per supplier attestation, corroborated by the module security policy."
      },
      "properties": [
        {
          "name": "appliedquantum:lineage:evidence",
          "value": "E4"
        },
        {
          "name": "appliedquantum:lineage:disclosureStatus",
          "value": "disclosed"
        },
        {
          "name": "appliedquantum:lineage:cmvpCertificate",
          "value": "CMVP-EXAMPLE-001"
        }
      ]
```

```
    },
    {
      "type": "data",
      "bom-ref": "path/PATH-ACQ-SCHEME-02",
      "name": "Acquirer to scheme A authorisation path",
      "properties": [
        {
          "name": "appliedquantum:path:pathId",
          "value": "PATH-ACQ-SCHEME-02"
        },
        {
          "name": "appliedquantum:path:negotiatedOutcome",
          "value": "TLS1.3/X25519"
        },
        {
          "name": "appliedquantum:path:observationWindow",
          "value": "2026-09-01T00:00:00Z/2026-09-08T00:00:00Z"
        },
        {
          "name": "appliedquantum:path:counterpartyRef",
          "value": "CP-SCHEME-A"
        },
        {
          "name": "appliedquantum:path:counterpartyClass",
          "value": "blocking"
        },
        {
          "name": "appliedquantum:path:counterpartyTargetState",
          "value": "false"
        },
        {
          "name": "appliedquantum:path:counterpartyStateReason",
          "value": "hybrid key establishment not supported at scheme endpoint"
        }
      ]
    }
  ],
  "dependencies": [
    {
      "ref": "impl/acquirer-gw-01/vendor-b-kem-lib",
      "provides": [
        "crypto/tls/acquirer-gw-01/mlkem768"
      ]
    }
  ]
}
```


The path record uses component type data as a stated profile convention. CycloneDX has no native path or topology object, data is schema-valid for a record-like collection, and consumers of this profile can rely on the convention. A native service or data-flow representation is under evaluation for a later version.

Read the three records together. The asset record states what is deployed and where it counts: migrationStatus is hybrid-deployed, the service is IBS-014, the custody and entropy context is stated. The implementation record carries the ancestry and its evidence, and the provides entry ties it to the asset. The path record states what the path negotiates: a classical exchange over the stated observation window, because the scheme endpoint does not support the hybrid group. The negotiated outcome on the path record, not the capability on the asset, is what layer 5 counts and what Effective

Coverage tests. The path is a first-class record the asset references, so one counterparty state change updates one record.

11. LICENCE AND VERSIONING

Licence. Creative Commons Attribution 4.0. Free to use, adapt and share, including commercially, with attribution to Steve Vaile and Marin Ivezic, Applied Quantum.

Canonical source. Published at cbomprofile.org. Cite as: *The Applied Quantum CBOM Profile, v1.0-RC, Steve Vaile and Marin Ivezic / Applied Quantum, cbomprofile.org.*

Machine-readable companions. A JSON Schema constraining every property name and value, a validator, a schema generator, and positive and negative fixtures accompany this release in the taxonomy repository. They are licensed Apache-2.0, matching CycloneDX's own schema licensing, while this document remains CC BY 4.0. The schema's canonical identifier is https://cbomprofile.org/schema/appliedquantum-cbom-profile-v1.0-rc.schema.json. The controlled vocabularies for entropy source designs, entropy generation designs and firmware families live in the same repository under vocab and are extended by issue. Validation has three layers: the CycloneDX schema for structure, this profile's schema and validator for property names, values, multiplicity and record rules, and the consuming framework's own sufficiency check. A pass at one layer establishes nothing about the others. The taxonomy repository is tagged at each Profile version; an assessment cites the tag or the schema's canonical identifier, not the moving main branch.

Versioning. Semantic, versioned independently of both frameworks. Each framework cites a minimum version of this profile rather than aligning baselines, because this is a shared dependency rather than an extension of either. A consumer's minimum-version citation, for example v1.0-RC or later, binds within a major version.

Stability. Enumeration values may be added within a minor version. Removals, renames and semantic changes require a major increment, because they invalidate stored assessments.

Extensibility. Institutions may add properties under their own namespace. Consumers ignore what they do not read. Omitting a property reduces what can be computed rather than invalidating the document.

12. SCOPE OF V1.0 FINAL

This release candidate is the intended content of v1.0 final, with one named addition. **Canonical asset identity and deduplication** is ratified for v1.0 and lands in a second release candidate in Q4 2026. Its counting rules cover keys, certificates and endpoints

under rotation, replicas, aliases, ephemeral keys and hybrid constructions, drafted against what the first consuming assessments surface.

Three questions this document previously listed as open are resolved here. The namespace string stays appliedquantum, and the registration is filed under it. The factual inventory and the assessment overlay remain one document for v1.0, with determinations marked in place, and the split is revisited at v2.0 once a machine-readable field registry can generate both views from one source. The controlled vocabularies publish as a registry with this profile, in the taxonomy repository, rather than through a standards venue first.

Carried to a later minor version rather than v1.0: per-claim evidence, so that an asset can hold strong evidence for its algorithm and weak evidence for its ancestry. A native path and hop graph model with ordered hops and per-hop state, which also decides whether populations need a distinct record structure. Evaluation of CycloneDX declarations, evidence and citation structures as carriers before further flat properties are added. And the machine-readable field registry that generates these tables, the schema and the consumer annexes from one source.