Internet Engineering Task Force
Internet-Draft
Intended status: Informational
Expires: 22 October 2026

J. Dev, Ed.
Comcast

20 April 2026

Output Schema Based on Cryptographic Bill of Materials (CBoM) for Post-Quantum Cryptography Usage
draft-dev-xipher-cbom-extension-00

Abstract

This document defines an output schema for inventorying cryptographic assets based on Cryptographic Bill of Materials (CBoM). It highlights key differences between an inventoring schema and CBoM, identifies gaps, and proposes a unified schema that can leverage existing CycloneDX parameters to create a usable cryptographic asset inventory.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on 22 October 2026.

Copyright Notice

Table of Contents

1. Introduction

A CBoM or Cryptographic Bill of Materials (CBoM) is an extension of the Software Bill of Materials (SBoM). SBoM are typically used to describe various components of a software, which helps assess vulnerabilities and provenance of such components. A CBoM is an added representation on top of SBoM that further describes cryptographic components in the software. For example, a CBoM can describe the cryptographic algorithms, key sizes, modes of operation, and other relevant details of the cryptographic components used in the software. This additional information helps organizations better understand what cryptographic algorithms exist, where they are used, and make informed decisions about their deployment (e.g., whether to deprecate algorithms or not, etc.). For the purpose of this document, we follow IBM's well known CBoM schema [cbom]. As of 2025, the latest version of CBoM schema that is part of the well known SBoM standard CycloneDX v1.7 is the following:

```
<CODE BEGINS> file "cbom.json"
          component:
            type: library
            name: example-component
            version: 1.0.0
            cryptographicProperties:
              - algorithm
                - additional algorithm properties
              - certificate
                - additional certificate properties
              - protocol
                - additional protocol properties
              - related cryptomaterial
                - additional related cryptomaterial properties
            other properties...
<CODE ENDS>
```

Figure 1: Pseudo representation of CBoM Available in CycloneDX v1.7

Such CBoM are currently generated at a software component level, either through static or dynamic analysis. In practice, this can look like a mix of automated tools and manual processes to generate CBoM for software components.

2. Cryptographic Asset Inventory (CAI)

A Cryptographic Asset Inventory (CAI) is an organizational level inventory of all cryptographic assets used within an organization. This includes hardware, software, and services that utilize cryptographic algorithms for security purposes. The difference between a CBoM and CAI is that the former is an extension of SBoM while the latter is centered around cryptographic assets that goes beyond software components. The purpose of a CAI is to provide a comprehensive view of the cryptographic assets in use, their configurations, and their compliance with organizational policies and industry standards. A CAI helps organizations manage their cryptographic assets effectively, ensuring that they are up-to-date, secure, and compliant with relevant regulations. Think of it this way - CBoMs are part of SBoMs, HBoMs, and other BoMs. CAI would be one schema that encompasses all instances of CBoMs in other BoMs.

3. Components of a CAI

An important feature of CBoM is its ability to provide detailed cryptographic properties at the component level. this essentially makes it use case independent. By capturing the maximum information available, it can be used for a variety of use cases. For CAI, however, we typically scope information that is relevant to cryptographic compliance. For this document, the use case we focus on is the ability to provide enough information that would help an organization be compliant with existing cryptographic standards and future PQC standards. Hence, a CAI should fulfil the following functions:

1. Criteria 1: At some level of grouping, provide a list of cryptographic assets that are present in the overall system. This system should encompass all assets, including software, hardware, and underlying infrastructure. The grouping can be done at any feasible level, i.e., application, ownership, remediation team, etc.

2. Criteria 2: The inventory should provide compliance information according to current cryptographic standards. These standards can be a combination of organizational encryption policy, industry standards for secure encryption, and regulatory requirements.

3. Criteria 3: The inventory should provide indicators of future quantum-resistant compliance. This should include any newly standardized algorithms that are present in code, network, or infrastructure.

4. Criteria 4: The inventory should contain minimum information that would enable required parties to assess and manage cryptographic risks effectively. It should not contain fields unnecessary for the purpose of inventorying.

5. Criteria 5: The inventory should to the extent possible, be automated for accurate updates to cryptographic information.

4. Differences between CBoM and a Cryptographic Asset Inventory (CAI)

We looked at the CryptoProperties schema in CycloneDX which is a representation of the original CBoM format [cyclone-cbom]. Based on the necessary properties of an organization level CAI, we identified the following areas in existing CBoM standards which can be augmented with additional information for a PQC inventory:

1. CBoM has a larger set of infomation that is use case independent

2. Complex inventories to inform PQC are hard to capture in the current structure

3. Clarity on mandatory versus optional fields needed

4. Hierarchy and relationships between cryptographic components are needed for understanding downstream effect

5. Additional information on criticality scoring would be helpful for asset prioritization

6. Chance of asset duplication

7. Need PQC-ready-compliance information in addition to CBoM information

Enhancements	Orignal CBoM Schema	CBoM Extension
Huge amount of information present.	CBoMs capture the maximum set of information that can be collected about cryptographic operations. This approach makes automation difficult and leaves a lot of blank values in the resulting JSON schema.	The extended format scopes down the fields required to make a migration decision to quantum-resistant cryptography.
Fundamental structural issues when cryptographic assets (application, framework, library, container, operating-system, device, firmware, or a file) are mapped at “organization” level.	Traditional BoMs designed at an organizational level is a list of information that is hard to action unless mapped to specific entities. This approach works well for single products, but is hard to scale. It also only works for code-based inventory where some contributor information might be available.	The extended format maps assets at both organization, application, and owner level. This helps in mapping assets to teams which have responsibilities to make changes and works for both code-based and non-code-based cryptographic assets.
No hierarchical nature of crypto-	The CycloneDX schema can accept both related and	Hierarchical system for assets

elements is enforced, so algorithms can exist in a CBoM without any source attribution or supporting metadata.	unrelated asset type. For example, cryptoProperties.assetTypes can be an algorithm, certificate, protocol, or relatedCryptoMaterial. However, a certificate can be part of a protocol, an algorithm can be part of a certificate, and the relatedCryptoMaterial will not work when assetType is an algorithm.	and supporting metadata added.
There is a chance of asset information duplication.	If asset types for both protocol and algorithm are recorded, their information can duplicate. For e.g., ikev2TransformTypes and tlsCipherSuites can be captured under both protocol and algorithm schema.	When assets are mapped at application level, the extended format ensures unique identification and tracking of each asset, reducing the chance of information duplication, especially because the algorithm information itself is not the main asset but a part of it.
The utility of the produced information in a CBoM is quite broad.	Without connection with internal policies, it will be hard to determine the relevance of the information and make it actionable. For e.g., algorithm information without highlight if they are deprecated or not is hard to action.	Introduce fields that can track compliance with current standards and new quantum-resistant standards.
Missing criticality score prevents determination of	Criticality scoring is missing.	Criticality scoring is integrated into the schema, which

migration priority for high-risk applications.		can be updated based on risk assessment frameworks like CARAF [caraf].

Table 1: Comparison of Existing CBoM and CBoM Extension Against CAI Criteria.

5. Data Structure for CAI

The standard CBoM structure as shown in Figure 1 (Section 5) has a total of 55 fields grouped by component type. Each algorithm is a separate component "linked" by dependency. The new structure for CAI as shown in Figure 2 (Section 5) has a total of 28 fields grouped by application. Each application contains multiple crypto-elements with their respective properties. The new structure is more compact and hierarchical, making it easier to understand and manage cryptographic assets at an application level for PQC needs.

```
<CODE BEGINS> file "og-str"
              Component (app/file/library/etc) {
                Algo properties
                Cert properties
                Proto properties
                Related cryptomaterial properties
              }
<CODE ENDS>
```

Figure 2: Original Structure

The new structure Figure 2 (Section 5) also makes it easier for cryptographic assets to inherit the risk properties of its parent application. The algorithm field is a sub-component wrapped inside a crypto-element, which in turn is wrapped in an application element. A noticeable change between the two is the inclusion of "Compliance_info", which specifies if certain algorithms are compliant or not.

```
<CODE BEGINS> file "new-str"
            Application {
              Application metadata {
                    Owner
                    Criticality
                    Other metadata
                }
              Crypto-elements {
```

```
                    Source: code {
                          Certificates {
                            Algorithm and compliance info
                            Crypto-element metadata
                          }
                          Crypto Libraries {
                            Algorithm and compliance info
                            Related Crypto Materials (keys, IVs, nonces)
                          }
                          Cert/library/proto {
                            Algorithm and compliance info
                            Crypto-element metadata
                          }
                      }
                    Source: network {
                          Protocols {
                            Algorithm and compliance info
                            Crypto-element metadata
                          }
                          Certificates {
                            Algorithm and compliance info
                            Crypto-element metadata
                          }
                        }
                      Source: infrastructure {
                            Internal {
                              Infra Type
                              (Container, OS, Hardware, Cloud services, Devices, Files)
                              {
                                Algorithm and compliance info
                                Crypto-element metadata
                              }
                            }
                            External {
                              Infra Type
                              (Container, Hardware, Cloud services, Files)
                              {
                                Algorithm and compliance info
                                Crypto-element metadata
                              }
                            }
                        }
                    }
                }
   <CODE ENDS>
```

Figure 3: New Structure

Using the CBoM schema as the standard template and adding the additional elements, we observe the following overall structure.

PQC Use Case Schema	Existing CBoM Schema Equivalent
application_ID	
application_ID.owner	component.authors
application_ID.criticality	
application_ID.asset_ID	component.name
application_ID.asset_ID .responsible_entity	
application_ID.asset_ID .purpose	
application_ID.asset_ID .compliance.is_compliant	
application_ID.asset_ID .compliance.is_PQC_vulnerable	
application_ID.asset_ID .compliance.upgradeable	
application_ID.asset_ID .compliance.compliance_message	
application_ID.asset_ID	cryptoProperties.scanner


|.source[y]
|(0=code.source_name,
|1=network.source_name,
|2=infra.source_name)
|application_ID.asset_ID
|.source[0].repo_name
|application_ID.asset_ID
|.source[x].discovery_date
|application_ID.asset_ID
|.type[x]
|(0=library, 1=protocol,
|2=certificate)
|cryptoProperties.assetTypes

application_ID.asset_ID .type[x].algorithm_in_use	cryptoProperties.algorithm.variant
	cryptoProperties.protocol.tlsCipherSuites
application_ID.asset_ID .type[x].device_info	cryptoProperties.algorithm.implementationPlatform
application_ID.asset_ID .type[x].signature_algorithm_name	cryptoProperties.certificate.certificateAlgorithm
application_ID.asset_ID .type[x].key_size	cryptoProperties.relatedCryptoMaterial.size
application_ID.asset_ID .type[x].signature_algorithm_oid	cryptoProperties.oid
application_ID.asset_ID .type[x].asset_path	cryptoProperties.detectionContext.filepath
application_ID.asset_ID .type[x].curve_name	
application_ID.asset_ID .type[0].line	cryptoProperties.detectionContext.lineNumbers
application_ID.asset_ID .type[0].code_block	cryptoProperties.detectionContext.additionalContext
application_ID.asset_ID	cryptoProperties.confidenceLevels.variant

```
   |.type[0].confidence_level        |
|
   +---------------------------------+-------------------------------------------------
+
   |application_ID.asset_ID          |
|
   |.type[1].protocol_name           |
|
   +---------------------------------+-------------------------------------------------
+
   |application_ID.asset_ID          |cryptoProperties.certificate.notValidAfter
|
   |.type[2].expiry_date             |
|
   +---------------------------------+-------------------------------------------------
+
   |application_ID.asset_ID          |
|
   |.type[2].parent_cert_references  |
|
   +---------------------------------+-------------------------------------------------
+
   |application_ID.asset_ID          |cryptoProperties.certificate.issuerName
|
   |.type[2].issuer                  |
|
   +---------------------------------+-------------------------------------------------
+
   |                                 |+39 rows not added
|
   +---------------------------------+-------------------------------------------------
+
```

Table 2: Overall Mapping of Proposed Extended CBoM based on CAI Schema to CBoM Schema. The fields on the left is the extended schema for PQC use case.

The above table maps the proposed CAI schema fields to their equivalent CBoM schema fields. Not all fields have a direct equivalent, indicating areas where the CAI schema introduces new concepts or structures not present in the original CBoM schema.

6. Conclusion

In this document, we proposed an extension to the existing CBoM schema to better suit the needs of an organizational level Cryptographic Asset Inventory (CAI). By identifying gaps in the current CBoM standards and introducing a more hierarchical and use-case focused structure, we aim to provide a more effective schema for organizations to manage their cryptographic assets, ensuring compliance with current and future standards, including post-quantum cryptography requirements. Future work includes developing tools for automated generation and maintenance of CAI based on this proposed schema.

7. IANA Considerations

This memo includes no request to IANA.

8. Security Considerations

This document should not affect the security of the Internet.

9. References

9.1. Normative References

[cbom] IBM, "IBM is donating its CBOM toolset to the Linux Foundation", 2025, <https://research.ibm.com/blog/cryptographic-cbom-linux-foundation>.

[cyclone-cbom] OWASP, "CycloneDX: An Authoritative Guide to CBOM", 2025, <https://cyclonedx.org/guides/OWASP_CycloneDX-Authoritative-Guide-to-CBOM-en.pdf>.

[caraf] Ma, CM., "CARAF: Crypto Agility Risk Assessment Framework", 2021, <https://academic.oup.com/cybersecurity/article/7/1/tyab013/6289827>.

Acknowledgements

This template uses extracts from templates written by Pekka Savola, Elwyn Davies and Henrik Levkowetz.

Contributors

Thanks to all of the reviewers of this initial draft.

Author’s Address

Jayati Dev (editor)
Comcast
1800 Arch St
Philadelphia, Pennsylvania 19130
United States of America
Email: Jayati_Dev@comcast.com