内容来源与真实性联盟

内容凭证

C2PA技术规范

2.2版，2025年5月1日：

目录

1. 引言 2

1.1. 概述 2

1.2. 范围 2

1.3. 技术概述 3

2. 术语表 6

2.1. 入门术语 6

2.2. 资产与内容 6

2.3. C2PA的核心要素 8

2.4. 附加条款 9

2.5. 概述 10

3. 规范性引用 12

3.1. 核心格式 12

3.2. 模式 12

3.3. 数字与电子签名 12

3.4. 可嵌入格式 13

3.5. 其他 13

4. 标准条款 15

5. 版本控制 16

5.1. 兼容性 16

5.2. 版本历史 16

6. 断言 23

6.1. 通用 23

6.2. 标签 23

6.3. 版本控制 24

6.4. 多个实例 24

6.5. 模式验证 25

6.6. 断言存储 25

6.7. 嵌入式数据与外部存储数据 25

6.8. 断言的编辑 25

6.9. 断言中的时间规格 26

7. 数据框 27
7.1. 通用 27
7.2. 架构与示例 27
8. 唯一标识符 28
8.1. 唯一标识C2PA清单和资产 28
8.2. 因冲突导致的清单版本控制 29
8.3. 识别非C2PA资产 29
8.4. URI 引用 30
9. 内容绑定 34
9.1. 概述 34
9.2. 硬性绑定 34
9.3. 软式固定器 35
10. 索赔 36
10.1. 概述 36
10.2. 语法 36
10.3. 创建声明 39
10.4. 多步处理 44
11. 清单 47
11.1. JUMBF的使用 47
11.2. 清单类型 53
11.3. 将清单嵌入各种文件格式 55
11.4. 外部清单 55
11.5. 嵌入外部清单的引用 56
12. 实体图 57
13. 加密技术 58
13.1. 哈希 58
13.2. 数字签名 59
14. 信任模型 63
14.1. 概述 63
14.2. 签署者身份 63
14.3. 验证状态 64
14.4. 信任列表 65

14.5. X.509证书 66
15. 验证 73
15.1. 验证过程 73
15.2. 返回验证结果 74
15.3. 显示清单信息 83
15.4. 确定哈希算法 83
15.5. 定位活动清单 84
15.6. 定位并验证声明 86
15.7. 验证签名 86
15.8. 验证时间戳 87
15.9. 验证凭证撤销信息 90
15.10. 验证断言 92
15.11. 验证成分 99
15.12. 验证资产内容 103
16. 用户体验 111
16.1. 方法 111
16.2. 原则 111
16.3. 披露级别 111
16.4. 公开审查、反馈与演进 112
17. 信息安全 113
17.1. 威胁与安全考量 113
17.2. 危害、误用与滥用 114
18. C2PA标准声明 116
18.1. 引言 116
18.2. 感兴趣区域 116
18.3. 关于断言的元数据 124
18.4. 标准C2PA断言摘要 129
18.5. 数据哈希 130
18.6. 基于BMFF的哈希 133
18.7. 通用框哈希 146
18.8. 集合数据哈希 155
18.9. 多资产哈希 158
18.10. 软封面装订 162

18.11. 云数据 167
18.12. 嵌入式数据 169
18.13. 缩略图 169
18.14. 操作 170
18.15. 成分 186
18.16. 元数据 198
18.17. 时间戳 200
18.18. 证书状态 201
18.19. 资产参考 202
18.20. 资产类型 203
18.21. 深度图 207
18.22. 字体信息 209
19. 专利政策 213
附录A：嵌入清单 214
A.1. 支持的格式 214
A.2. 在多部分资源中嵌入清单文件 216
A.3. 将清单嵌入非BMFF资产 216
A.4. 将清单嵌入PDF文件 220
A.5. 将清单嵌入基于BMFF的资源 222
A.6. 将清单嵌入基于ZIP的格式 229
附录B：c2pa.metadata的实现细节 232
B.1. 完全支持的架构 232
B.2. 部分支持的架构 232
附录C：弃用注意事项 240
C.1. 构造的状态 240

第一章 导言

1.1. 概述

随着数字内容传播速度的加快以及强大创作与编辑技术的普及，确立媒体来源至关重要，这能保障透明度、促进理解，并最终赢得信任。

当前媒体信任正面临前所未有的挑战。社交平台通过日益复杂且不透明的算法放大特定内容的传播范围与影响力，导致来源错误、语境错乱的内容迅速蔓延。无论是无意的误导信息还是蓄意的虚假宣传，不真实的内容正呈激增之势。

目前，希望为作品添加元数据的人士无法在跨平台环境中以安全、防篡改且标准化的方式实现这一目标。若缺乏来自权威来源的元数据，出版商和消费者将无法获取判断媒体真实性所需的关键背景信息。

来源追溯技术赋能内容创作者与编辑——无论其地理位置或技术接触程度如何——都能披露资产的创建方式、变更过程及具体修改内容。每次资产变更时，其既有溯源记录均被完整保留，新变更信息将逐层叠加至溯源链中。由此，具备溯源机制的内容可提供真实性标识，使消费者能够识别被篡改的内容。此类溯源信息可包含变更内容及变更来源。为创作者、发布者和消费者提供溯源能力，是构建网络信任体系的关键基础。

为大规模解决出版商、创作者和消费者面临的这一问题，内容来源与真实性联盟（C2PA）制定了本技术规范，旨在提供内容来源与真实性保障。该规范通过构建丰富的数字来源应用生态系统，支持全球范围内的自愿采用数字来源技术，服务于各类个人与组织，同时满足相应的安全要求。

本规范始终基于行业专家及合作伙伴组织（包括Project Origin联盟与内容真实性倡议组织）收集的场景、工作流程及需求进行制定。监管机构与政府部门亦可采用本规范建立数字溯源标准。

1.2. 适用范围

本规范阐述了C2PA架构的技术要点，该架构是一种存储和访问可通过加密手段验证信息的模型，其可信度可依据预定义的信任模型进行评估。本文档包含创建和处理C2PA清单及其组件的相关信息，涵盖运用数字签名技术实现防篡改特性及建立信任机制的具体方法。

在制定本规范之前，C2PA制定了指导原则，使我们能够始终专注于确保规范的使用方式尊重隐私和个人对数据的控制权，同时以批判的眼光审视潜在的滥用和误用风险。例如，强烈建议本规范的实施者为媒体资产的创作者和发布者提供控制是否包含特定来源数据的能力。

重要提示

摘自指导原则的总体目标章节：

> C2PA规范不应就特定来源数据集的"优劣"作出价值判断，而仅需验证其中包含的声明是否与基础资产相关联、格式正确且未遭篡改。

该规范必须避免对消费者内容可访问性产生负面影响。

C2PA的其他文件将处理具体实施考量，例如预期用户体验以及威胁与危害建模的细节。

1.3. 技术概述

C2PA信息包含一系列声明，涵盖资产创建、编辑操作、采集设备详情、内容绑定等诸多领域。这些被称为断言的声明构成了特定资产的来源信息，代表着一系列信任信号，可供人类使用者提升对该资产可信度的认知。断言与附加信息共同封装为数字签名实体——声明。该声明由声明生成器代表签名者使用签名者的签名凭证进行数字签名，从而生成声明签名。

这些断言、声明及声明签名均由名为声明生成器的硬件或软件组件整合为可验证单元，称为C2PA清单（参见图1“C2PA清单及其组成部分”）。存储于资产内容凭证中的C2PA清单集合，即构成其来源数据。

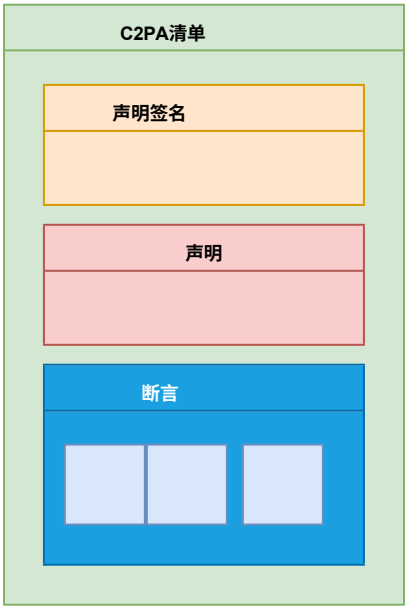

图1. C2PA清单及其组成部分

1.3.1. 建立信任

在C2PA信任模型中，信任决策的基础是签名者身份——该身份与用于在C2PA清单中签署声明的加密签名密钥相关联。当C2PA清单的声明签名与可信时间戳结合时，可无限期地进行验证流程，以确定声明是否在签名凭证有效且未被撤销期间签署。

1.3.2. 示例

一种非常常见的场景是用户使用支持C2PA的相机（或手机）拍摄照片。此时相机将生成一份清单，其中包含若干声明：包括相机自身信息、图像缩略图以及若干加密哈希值，这些哈希值将照片与清单绑定。这些声明将被列于声明中，经数字签名后，整个C2PA清单（参见图2"照片示例C2PA清单"）将被嵌入输出JPEG文件。该C2PA清单将永久保持有效。

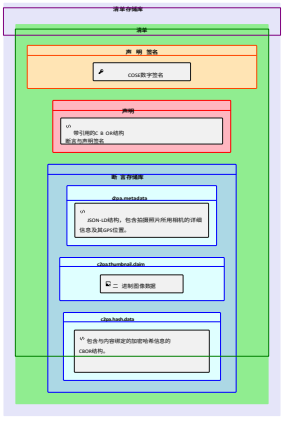

图2. 照片的C2PA清单示例

清单消费者（如C2PA验证器）通过首先验证数字签名及其关联凭证，帮助用户建立资产可信度。它还会检查每个声明的有效性，并将声明信息与签名呈现给用户，以便用户能基于充分信息判断数字内容的可信度。

1.3.3. 设计目标

在构建C2PA架构时，确立明确的设计目标至关重要，以确保该技术能广泛适配全球各类软硬件实现，并实现全民可及。相关目标详见表1"C2PA设计目标"。

表1. C2PA设计目标

目标	描述
隐私	使用户能够控制其信息的隐私权限，包括消费数据及溯源记录中的信息
责任	确保消费者能够确定资产的来源
可扩展性	支持以与网络媒体创作/消费同等规模创建/消费/验证媒体来源信息
可扩展性	确保未来元数据和凭证提供商能够添加其信息，而无需C2PA的输入或批准
互操作性	确保不同实现方案能无歧义地协同运作
全流程适用性	在资产从创建到后续所有修改及发布/分发的全流程中，跨多种工具保持资产溯源信息完整性
技术极简主义	仅在规范中创建最低限度的新技术，依托先前成熟的技术体系
安全性	设计确保消费者能够信任溯源信息的完整性与来源，并确保设计方案经专家评审
内容无处不在	支持为所有常见媒体类型（包括文档）添加来源信息
灵活区域性	支持在线与离线（仅资产存储）两种模式，实现溯源信息的存储、使用及验证
全球通用性	为全球感兴趣用户的需求而设计
无障碍性	确保技术使用方式符合公认的无障碍标准，例如《网络内容无障碍指南》（WCAG）
危害与滥用	设计需防范并减轻潜在危害，包括对人权的威胁及弱势群体面临的不成比例风险
持续演进	持续对照这些目标审查技术规范，确保其始终作为我们的优先事项

第二章 术语表

2.1. 导论术语

2.1.1. 参与者

参与C2PA生态系统的人类或非人类（硬件或软件）实体。例如：相机（采集设备）、图像编辑软件、云服务或使用此类工具的人员。

> **注** 在C2PA生态系统中，组织或行为者群体也可视为行为者。

2.1.2. 声明生成器

生成资产声明及声明签名的非人类（硬件或软件）行为体，由此形成该资产关联的C2PA清单。

2.1.3. 签名者

持有用于签署声明的私钥凭证的持有者。签名者通过凭证主体进行身份识别。

2.1.4. 清单消费者

一个消耗具有关联C2PA清单的资产的操作者，其目的是从C2PA清单中获取来源数据。

2.1.5. 验证者

指执行验证过程中所述操作的清单消费者。

2.1.6. 操作

行为体对资产执行的操作。例如"创建"、"嵌入"或"应用滤镜"。

2.2. 资产与内容

2.2.1. 数字内容

资产中代表实际内容的部分，例如图像的像素，以及理解内容所需的任何附加技术元数据（例如色彩配置文件或编码参数）。

2.2.2. 资产元数据

关于资产及其数字内容的非技术性信息。

2.2.3. 资产

包含数字内容、资产元数据及可选C2PA清单的文件或数据流。

注	为符合本定义目的，我们将扩展"文件"的典型定义，涵盖云原生及动态生成数据。

2.2.4. 衍生资产

衍生资产是指基于现有资产进行操作以修改其数字内容而创建的资产。

示例：经过剪短的音频流，或添加了页面的文档。

2.2.5. 资产呈现

指对资产进行"非编辑性转换"操作（如重新编码或缩放）后生成的资产呈现形式（可作为资产的组成部分或全新资产存在）。

示例：为降低屏幕分辨率或网络带宽而重新编码的视频文件。

2.2.6. 组合资产

组合资产是指通过整合多个数字内容片段（称为组件）构建而成，这些片段源自一个或多个其他资产。当基于现有资产创建时，它属于衍生资产的特殊情况——但组合资产也可从"空白状态"开始创建。

示例：

- 通过将现有视频片段和音频片段导入"空白画布"创建的视频。
- 将另一张图像导入并叠加在原始图像上的图像。

2.2.7. 编辑性转换

一种改变数字内容意图或含义（或两者兼有）的转换类型。

2.3. C2PA的核心要素

2.3.1. 声明

一种数据结构，用于表示签名者所作声明（或"创建"声明），或在声明生成时直接收集的、关于该资产的信息。此数据构成C2PA清单的一部分。

2.3.2. 声明

一种经过数字签名且防篡改的数据结构，该结构引用了一组关于资产的断言，以及表示内容绑定所需的信息。若存在任何被删除的断言，则包含相应的声明。此数据是C2PA清单的一部分。

2.3.3. 声明签名

该声明上的数字签名使用签名者拥有的私钥生成。声明签名是C2PA清单的一部分。

2.3.4. C2PA清单

基于一个或多个断言（包括内容绑定）、单个声明及声明签名的组合，所形成的资产来源信息集合。C2PA清单属于C2PA清单存储库的组成部分。

注	C2PA清单可引用其他C2PA清单。

2.3.5. C2PA清单存储库

一组C2PA清单，可嵌入资产内部或置于资产外部。

2.3.6. 内容凭证

这是C2PA清单的首选非技术术语。因此，C2PA清单存储库代表资产的内容凭证。

内容凭证亦泛指整体C2PA技术体系，故本质上作为复数名词使用。若单个C2PA清单即为内容凭证，则多个C2PA清单或更广泛的通用概念即为内容凭证。

2.3.7. 活动清单

在C2PA清单存储库内的C2PA清单列表中，最后一个清单即为可验证内容绑定集合所在的清单。

2.3.8. 溯源

通过来源数据所呈现的逻辑概念，用于理解资产的历史及其与参与者及其他资产的交互过程。

2.3.9. 来源数据

资产的C2PA清单集合，若为组合资产则包含其组成部分的清单。

> **注** C2PA清单可引用其他C2PA清单。

2.3.10. 真实性

数字内容的属性，包含一组可通过密码学验证确保未被篡改的事实（如来源数据和硬绑定）。

2.3.11. 内容绑定

将数字内容与特定资产关联的C2PA清单的信息，该关联可通过硬绑定或软绑定实现。

2.3.12. 硬绑定

通过一个或多个加密哈希值唯一标识整个资产或其部分内容。

2.3.13. 软绑定

一种内容标识符，其特征为：(a) 统计上不具备唯一性（如指纹），或 (b) 以不可见水印形式嵌入被标识的数字内容中。

2.3.14. 信任信号

用于辅助清单消费者判断资产可信度的信息集合。这些信息补充了基础信任模型所依赖的签名方。

2.3.15. C2PA信任列表

由C2PA管理的X.509证书信任锚列表，该列表为硬件及软件签名者签发证书，供其用于声明签名。

2.4. 附加条款

2.4.1. 持久内容凭证

持久内容凭证是一种内容凭证，其存在一个或多个软绑定，可实现其在清单存储库中的发现。

2.4.2. 指纹

一组可从数字内容计算得出的固有属性，用于识别该内容或其近似副本。

> **示例：** 资产可能因元数据被删除或损坏而与C2PA清单分离。可通过资产数字内容的指纹在数据库中检索，从而恢复带有完整C2PA清单的资产。

2.4.3. 隐形水印

以人类基本无法察觉的方式嵌入资产数字内容中的信息，可用于唯一标识该资产或存储指向C2PA清单的引用。

2.4.4. 可见水印

数字内容中可感知组件，承载有关资产来源的人类可读信息。

2.4.5. 清单存储库

用于存放C2PA清单及C2PA清单存储库的仓库，可通过内容绑定进行检索。

2.5. 概述

该图展示了所有这些不同元素如何共同构成C2PA架构。

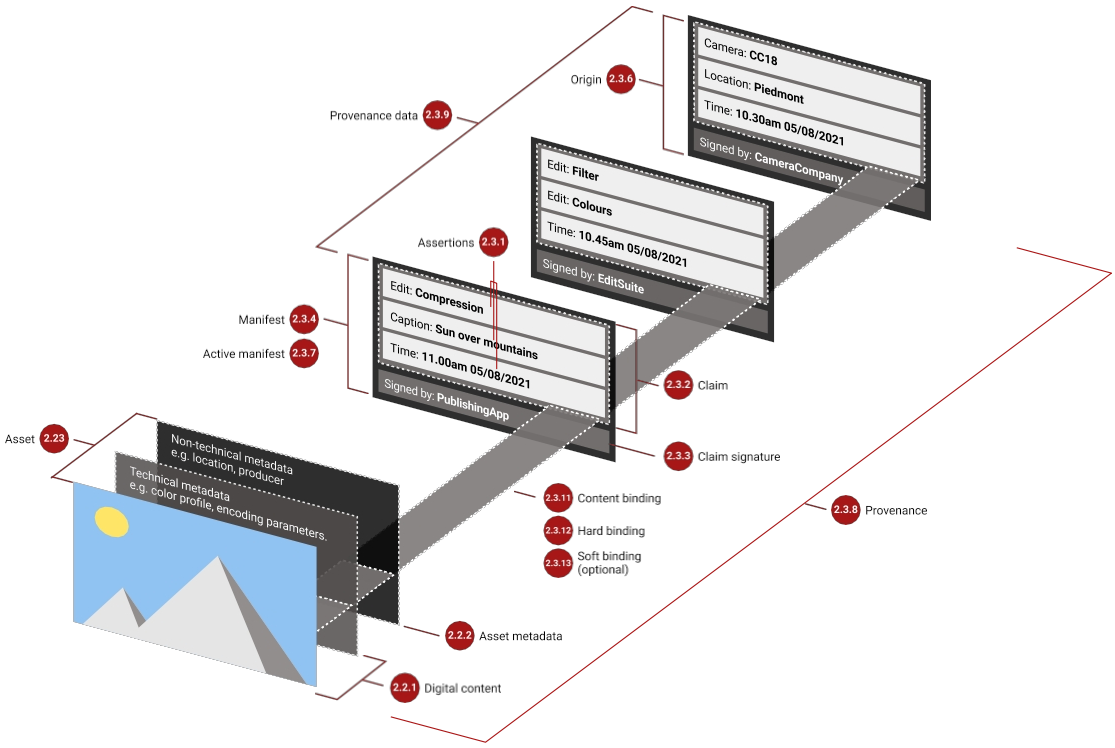

图3. C2PA的组成要素

第3章. 规范性参考

3.1. 核心格式

- CBOR
- JSON
- JSON-LD
- JPEG通用元数据框格式（JUMBF）

3.2. Schemas

- CDDL
- JSON Schema
- Dublin Core 元数据倡议

3.3. 数字与电子签名

- 加密消息语法 (CMS)
- 互联网 X.509 PKI 时间戳协议
- 互联网 X.509 公钥基础设施证书和证书吊销列表 (CRL) 配置文件的算法和标识符
- 互联网X.509公钥基础设施：DSA与ECDSA的附加算法及标识符
- 美国安全哈希算法
- 在线证书状态协议 (OCSP)
- JSON网络算法（JWA）
- PKCS #1：RSA 加密规范 2.2 版
- Edwards曲线数字签名算法（EdDSA）
- CBOR对象签名与加密（COSE）
- 在COSE消息中使用RSA算法
- 用于互联网X.509公钥基础设施的Ed25519、Ed448、X25519和X448算法标识符
- X.509证书通用扩展密钥用途（EKU）用于文档签名
- CBOR对象签名与加密（COSE）：用于承载和引用X.509证书的头部参数
- 互联网X.509公钥基础设施：X.509证书中的徽标

- JSON高级电子签名（JAdES）

3.4. 可嵌入格式

- ISO 基础媒体文件格式 (BMFF)
- PDF 1.7
- PDF 2.0
- JPEG 1
- JPEG XT，ISO/IEC 18477-3
- JPEG XL，ISO/IEC 18181-2:2024
- PNG
- SVG
- GIF
- ID3
- 数字负片或DNG
- TIFF/EP
- TIFF v6)
- RIFF
- 多图片格式（MPF）
- 开放字体格式
- OpenType

3.5. 其他

- 可扩展元数据平台 (XMP)
- XMP 的 JSON-LD 序列化
- IPTC 照片元数据标准
- Exif
- 全局唯一标识符
- 统一资源名称（URN）
- 全局唯一标识符（UUID）
- ISO 8601
- RFC 3339
- RFC 2326

- 媒体片段

- Web 注释数据模型
- Brotli压缩数据格式
- RFC 5646、BCP 47

第4章 标准术语

关键词"必须"、"不得"、"要求"、"应当"、"不应当"、"应该"、"不应该"

本文件中出现的"RECOMMENDED"、"NOT RECOMMENDED"、"MAY"和"OPTIONAL"等术语，无论采用何种字母大小写形式（大写、小写或混合），均应按BCP 14、RFC 2119及RFC 8174中的定义进行解释。

第5章. 版本控制

5.1. 兼容性

随着内容凭证规范的演进，诸如框标签、断言（及其字段）、声明和时间戳等构造也随之发展。新增了断言类型，部分现有断言及声明已推出包含附加字段的新版本。此外，某些构造已被废弃。本规范中标记为"废弃"的构造，表示声明生成器不得写入该构造（或值），但验证器仍应读取。

为促进声明生成器与验证器之间的互操作性，声明生成器需声明其生成声明所采用的规范版本。当声明生成器声明其采用特定版本规范时，即表明该资产的活动清单是依据该版本规范生成，因此不包含附录C *《弃用考量》* 中表19"构造状态"所列该版本规范下的任何弃用构造。

注 本规范未规定此声明的具体技术方式，但预期将通过其他途径提供指导。

验证器应至少兼容规范的一个版本，但可兼容更多版本。兼容特定版本规范的验证器应支持该版本中所有未废弃的构造。若验证器遇到使用其不支持版本规范构造的清单（因构造已弃用或未知），可忽略该弃用构造并按该构造不存在处理清单其余部分。或可将整个清单视为来源未知，通过返回 `ingredient.unknownProvenance` 或 `manifest.unknownProvenance` 状态码进行处理。

5.2. 版本历史

5.2.1. 2.2 - 2025年5月

本版本重点对规范进行了技术性与编辑性修订，旨在阐明2.1版的部分新特性，同时回应了实施者的需求。规范已更新以反映该领域的最新最佳实践。

- 新增软绑定解析API的补充规范
- 在组件断言中新增字段以标识软绑定清单恢复
- 新增对多部分资产的支持，例如Android动态照片
- 新增更新清单中添加时间戳和撤销信息的支持，替代时间戳清单
- 新增对"声明签名生成时间"的支持

- 新增对c2pa-kp-claimSigning扩展密钥使用（EKU）的支持
- 限制C2PA信任列表仅用于具有c2pa-kp-claimSigning执行关键单元(EKU)的证书
- 引入 digitalSourceType 值 http://c2pa.org/digitalsourcetype/trainedAlgorithmicData （替代c2pa.trainedAlgorithmicData）及http://c2pa.org/digitalsourcetype/empty
- 用嵌入式数据断言替换数据框
- 针对清零编辑后的断言提供了额外指导
- 澄清了创建断言和收集断言在信任模型中的使用
- 明确了"签名者"与"声明生成者"的术语定义及其角色定位
- 对各类硬性绑定断言进行变更与优化
    - 允许 c2pa.hash.data 排除资产的经典元数据部分
    - 在 c2pa.hash.boxes 断言中添加排除支持
    - 在更新清单中支持使用 c2pa.hash.bmff 断言
- 明确声明存储库中允许的JUMBF盒子类型
- 明确证书吊销处理机制
- 明确时间戳验证规则
- 改进并澄清了操作断言
- 改进软绑定断言
- 重构BMFF哈希图以提升清晰度与正确性
- 取消了要求清单存储库中所有清单必须被引用的规定

5.2.2. 2.1 - 2024年9月

本版本针对规范进行了技术与编辑层面的双重修订，旨在提升内容凭证的安全性与可靠性。所有已公开的安全漏洞均已修复，规范内容亦更新为该领域最新最佳实践标准。

- 明确定义清单与资产状态
    - 规范化清单格式
    - 有效清单
    - 可信清单
    - 有效资产
- 明确的废弃处理与版本管理定义及流程

- 用于标记清单的新 `c2pa` URN 命名空间！

        - 包含完整定义的ABNF规范

            - 新版成分v3声明

    - 支持更丰富的基于成分的工作流模型。
    - 新增对dataTypes和claimSignature的支持。
    - 字段重命名以增强与其他声明的一致性。
    - 新增验证状态字段以配合新状态信息
    - dc:title 和 dc:format 现为可选字段
- 新增c2pa.hash.bmff.v3断言
    - 支持基于 BMFF 的资产进行固定和可变块大小的哈希处理
- 新增时间戳清单
    - 为特定资产建立"存在时间"。
    - 类似于更新清单，但签名者为TSA
- 改进的RFC 3161标准时间戳处理模型。
    - sigTst2与CTT时间戳技术
    - 引入新的C2PA TSA信任列表
- 增强验证机制
    - 为所有标准断言提供详细验证说明
    - 使用成分断言时，现要求对成分进行验证
    - 扩展成分验证功能以提供更详细的状态信息
    - 支持对配料中编辑后的声明进行验证
    - 新增时间戳验证的详细要求
    - 数据框及任何自定义框的哈希URI现已纳入验证范围
    - 定义了处理唯一ID匹配清单的流程
    - 在验证流程中处理"孤立清单"
    - 新增大量验证状态码，包括新型"信息性"码类型
- 改进文档及哈希算法安全性
    - 基于BMFF的资产
    - "通用箱"
    - ZIP

- 格式嵌入部分已移至独立附录
    - 新增对JPEG-XL的支持

- 软绑定机制优化
- 改进操作断言
    - 标准清单中必须包含 `c2pa.created` 或 `c2pa.opened` 之一
    - 新增若干标准动作类型
    - 单个清单中现可包含多个动作断言
    - 操作模板现通过更多示例进行更清晰说明
    - 基于RFC 3339的感兴趣区域
- 已明确各类资产唯一标识符的类型/形式。
- 为JPEG Trust添加了部分缺失的兼容性支持
- 清理所有CDDL许可条款，包括移除规范性表述
- 并对多个编辑领域进行了改进
    - 将自定义标签重新定义为自定义命名方案。
    - 嵌入PDF文件
    - 为ISO标准化准备文档进行了大量编辑改进

5.2.3. 2.0版 - 2024年1月

此版本与先前版本存在重大差异。它减少了"参与者"一词的使用，该术语不再代表人类和组织。除验证器配置的信任列表外，还引入了新的默认信任列表"C2PA信任列表"，旨在覆盖颁发给硬件和软件的证书。这一理念转变导致规范中出现以下功能变更：

- 仅允许使用X.509证书进行签名。
- 完善验证与信任模型章节
    - 引入"结构良好"与"有效"的C2PA清单概念
    - 厘清验证流程的各个环节
- 优化元数据处理机制
    - 移除了已弃用的Exif、IPTC和Schema.org元数据断言
    - 定义了新的通用"元数据断言"概念
    - `c2pa.metadata`仅允许使用固定的模式集与值集
    - 创建 `c2pa.metadata` 的流程现已详细记录
    - XMP处理部分已重构以反映相关变更
    - 改进关于清单外标准元数据位置哈希处理的建议
- 移除了"W3C可验证凭证"章节

    - 删除了所有相关引用及VC存储库内容
    - 从动作断言中移除了参与者字段
    - 从断言元数据中移除了已识别人类
- 移除了"培训与数据挖掘"断言
- 移除了"推荐"断言

此外，为完善规范的各个方面，还进行了以下其他变更：

- 声明升级至v2版本
    - 移除已弃用及未使用的字段
    - 将断言拆分为创建断言与收集断言
    - 仅允许单一声明生成器，且必须是签名者
    - `claim-generator-info` 字段新增专属的 `operating_system` 字段
- 强烈建议所有支持盒式哈希的格式均采用此方案
- 移除了已弃用的 `c2pa.hash.bmff` 断言
- 新增 `c2pa.watermarked` 操作
- `c2pa.font` 操作现已更名为 `font` 操作
    - 同时 `c2pa.font.info` 现更名为 `font.info`
- 清理了CDDL模式的呈现
- 更新了部分规范性引用并移除了关于未来版本的注释
- 进行了大量编辑改进，包括修复链接

5.2.4. 1.4 - 2023年11月

- 新增对将C2PA清单嵌入ZIP格式文件（如EPUB、OOXML、ODF、OpenXPS）的支持
- 清单文件现可压缩至特殊数据块容器中。
- 新增对多文件（即集合）哈希的支持
- 新增文本类格式（如PDF、Office、EPUB等）的兴趣区域支持
- 新增`c2pa.`metadata断言以支持Exif、IPTC、Schema.org和XMP
- 对TIFF嵌入支持进行了重大修订
- 新增对在OpenType和TrueType字体中嵌入C2PA清单的支持
- 新增对PDF中对象级清单的支持

- 扩展了嵌入式清单的链接头支持
- 澄清了框哈希相关问题

- 澄清了签名相关问题，包括时间戳、PKIStatus和文档签名EKU
- 与 Exif 3.0 保持一致
- 改进了CDDL模式
- 多项编辑改进

5.2.5. 1.3 - 2023年4月

- 动作断言新增v2版本，支持多项新选项
- 新版本v2的成分声明，支持嵌入式数据
- 新增资产引用与资产类型声明
- 新增数据框，用于在清单内存储任意数据
- 新增通用框哈希方法，实现更包容的字节范围哈希
- 新增"感兴趣区域"数据结构，可应用于各类声明
- 新增文档签名EKU作为C2PA签名方的替代默认EKU（当验证方未配置EKU列表时）
- 新增数字源类型字段供C2PA使用
- 新增对多种新格式的支持：MPF、WebP、AIFF、AVI、GIF
- 更新实体图以反映自1.0版本以来的新增内容
- 更新X.509证书的COSE头部定义至RFC 9360标准
- 更新PDF嵌入指南及其与PDF签名的关联说明
- 更新了关于JUMBF哈希和JUMBF框切换的信息
- 弃用BMFF哈希v1版本
- 明确了在C2PA清单中使用JUMBF保护框的规范
- 明确了C2PA的特定要求：所有中间X.509证书均需包含在COSE签名中
- 明确时间戳具有无限期有效性
- 大量编辑改进！！

5.2.6. 1.2 - 2022年10月

- 新增C2PA清单嵌入DNG或TIFF格式的详细说明
- 在Actions中新增digitalSourceType字段
- 将 `stds.iptc.photometadata` 改为 `stds.iptc` 以支持 IPTC 视频元数据
- 明确添加可选字段时的断言版本控制

5.2.7. 1.1 - 2022年9月

- 定义支持加盐盒式哈希的机制
- 新增 `c2pa.hash.bmff.v2` 断言，通过修改哈希模型提升安全性
- 启用声明元数据功能
- 用 `claim_generator_info` 替换 `claim_generator_hints`
- 新增断言以支持背书概念
- 改进 `c2pa.actions` 断言
- 所有错误与状态码现均以 `c2pa` 为前缀
- 定义W3C虚拟证书的编辑机制
- 明确证书中EKU的验证规则
- 修订验证算法以反映技术变更
- 对CDDL和JSON模式进行修正以符合规范文本
- 修订图表以反映变更
- 各类编辑与排版修正
- 更新规范性引用（含JUMBF及W3C VC数据模型）

5.2.8. 1.0 - 2021年12月

- 初始版本

第6章 断言

6.1. 概述

预计系统中用于创建或处理资产的每个声明生成器，都将生成或组合一个或多个关于资产何时、何地以及如何产生或转换的断言。断言是带标签的数据，通常（但非强制要求）采用基于CBOR的结构，用于声明资产相关信息。部分断言包含人工生成的信息（如辅助功能的替代文本），其余则来自机器（软硬件）提供的生成数据（如相机型号）。

断言示例包括：

- 元数据（如相机制造商或镜头等设备信息）；
- 对资产执行的操作（例如剪辑、色彩校正）；
- 资产或其组成部分的缩略图；
- 内容绑定（例如加密哈希值）。

某些断言可能被后续声明覆盖（参见第6.8节“断言的覆盖”），但一旦作为声明的一部分提出，便不可修改。

6.2. 标签

6.2.1. 命名空间

C2PA数据结构中的字符串值可使用句点(.)作为分隔符组织为命名空间。本规范定义的任何字符串值均须以C2PA命名空间`c2pa`开头。实体专属命名空间应以该实体的互联网域名开头，类似于Java包的定义方式（例如：`com.litware`, `net.fineartschool`）。

实体特定命名空间的点分隔组件应遵循POSIX或C语言环境中规定的变量命名规范（`[a-zA-Z0-9][a-zA-Z0-9_-]*`），具体定义如下文命名空间ABNF所示。

命名空间的ABNF

```
限定命名空间 = "c2pa" / 实体
实体 = 实体组件 *( "." 实体组件 )
实体组件 = 1( 数字 / 字母 ) *( 数字 / 字母 / "-" / "_" )
```

6.2.2. 标签命名

每个断言都具有由C2PA规范或外部实体定义的标签。这些标签是带命名空间的字符串，其命名空间遵循前文所述规则或由实体定义。最常见的标签将定义在c2pa命名空间中，但标签也可使用任何遵循规范的命名空间。标签采用简单的递增整数方案进行版本控制（例如`c2pa.actions.v2`）。若未提供版本号，则默认为`v1`。公开已知的标签列表详见第18章《*C2PA标准断言*》。

注	本文件早期版本亦为成熟标准预留命名空间， 但现已改为通过实体特定命名空间（如`org.iso`、`org.w3`）实现。

断言标签的ABNF表示法

```
namespaced-label = qualified-namespace labelqualified-
namespace = "c2pa" / entity
实体 = 实体-组件 *( "." 实体-组件 )
实体组件 = 1( 数字 / 字母 ) *( 数字 / 字母 / "-" / "_" )标签 = 1*( "." 标签组件 )
标签组件 = 1( 数字 / 字母 ) *( 数字 / 字母 / "-" / "_" )
```

标签中以句点分隔的组件遵循POSIX或C语言环境中规定的变量命名规范（`[a-zA-Z][a-zA-Z0-9_-]*`），但需遵守以下限制：重复的下划线字符（`___`）专用于标记相同类型的多重断言。

6.3. 版本控制

当断言的架构发生变更时，应采用向后兼容的方式。这意味着可添加新字段，现有字段可标记为弃用（即允许读取但禁止写入）。现有字段不得删除。此时标签应包含递增的版本号，例如从 `c2pa.action`（已弃用）迁移至 `c2pa.action.v2`。

由于可选字段的添加可在保持向后兼容性的前提下进行，因此此类字段可添加至现有断言的架构中，且无需更改版本号。

C2PA标准断言中已弃用的字段应在第18章《*C2PA标准断言*》中标注。声明生成器在创建断言时不得向已弃用的断言字段插入数据。

当需要进行非向后兼容变更时，应为断言赋予新标签，而非提升标签版本号。

注	例如，`c2pa.ingredient` 可更改为虚构的 `c2pa.component`。

6.4. 多重实例

同一清单中可包含多个相同类型的断言，但由于断言通过声明标签被引用

其标签进行引用，因此断言标签必须唯一。通过在标签后添加双下划线和单调递增的索引来实现这一要求。例如：若清单中仅包含一个 c2pa.metadata 类型的断言，则断言标签为 c2pa.metadata；若包含三个此类断言，标签则为 c2pa.metadata、c2pa.metadata__1 和 c2pa.metadata__2 。

当标签包含版本号时，该版本号即为标签组成部分。因此当存在多个实例时，实例编号仍紧跟标签之后——例如 c2pa.ingredient.v2__2。

6.5. 模式验证

本文档提供的模式以及可从C2PA网站下载的机器可读模式，仅用于辅助理解读写语法。验证器无需也不建议执行任何形式的模式验证。

6.6. 断言存储库

清单中某项声明所引用的断言集合，被整合为一个称为*断言存储库的*逻辑结构。断言及断言存储库应按第11.1节"JUMBF的使用"所述方式存储；特别要求：声明的created_assertions或gathered_assertions（但不包括redacted_assertions）中引用的每个断言，均须存在于与该声明位于同一C2PA清单的断言存储库中。

每个清单仅包含一个断言存储库。但由于资产可能关联多个清单（每个清单代表特定断言序列），因此单个资产可能关联多个断言存储库。

6.7. 嵌入式数据与外部存储数据

某些断言数据因体积庞大或使用频率较低，可能采用外部托管方式。此类数据不会嵌入断言存储库，而是通过统一资源标识符（URI）进行引用。该机制通过云数据断言实现（详见第18.11节"云数据"）。与嵌入式断言数据不同，云数据不会在清单验证过程中被检索或验证，仅当应用程序根据第15.10节"验证断言"所述的不同验证规则明确需要时，才会进行检索和验证。

6.8. 断言的编辑

当资产作为组件使用时，其清单中存在的断言可被移除。此过程称为断言编辑。

编辑操作包括：从清单的断言存储中移除整个断言，或保留标记的断言容器，但将该断言内的JUMBF内容框替换为单个UUID内容框，其ID字段值为CAA98EEE-9D4D-F80E-86AD-4DFFCA263973（称为C2PA编辑UUID）

且其DATA字段仅包含零值（二进制0x00）。

此外，声明中应通过URI引用被编辑断言的形式，在声明的edited_assertions字段中添加移除记录。强烈建议声明生成器同时添加c2pa.redacted操作断言，其编辑字段应遵循第18.14.4.7节"参数"中的描述。

当编辑引用C2PA清单的组件断言时，若编辑后该清单在C2PA清单存储库中不存在其他引用，则应将其移除。

注释

由于每个断言的URI引用包含断言标签，因此也知道断言的类型。

信息（如缩略图、元数据等）被移除。这使得人类和机器都能应用规则来判断移除是否合理。

除非断言的编辑同时需要修改数字内容，否则应使用更新清单来记录编辑操作，因为该清单声明了内容未发生变更。

声明生成器不得对标签为 `c2pa.actions` 或 `c2pa.actions.v2` 的断言进行编辑，因该断言类型承载着理解资产历史的关键信息。同样不得对任何与内容强关联的断言进行编辑——包括`c2pa.hash.data`、`c2pa.hash.boxes`、`c2pa.hash.collection.data`、`c2pa.hash.bmff.v2`（已弃用）或`c2pa.hash.bmff.v3`，因这些断言对确定资产完整性至关重要。

该声明将验证失败（详见第15.11.3节“成分声明验证”），

当通过已弃用的成分声明（`c2pa.ingredient` 或 `c2pa.ingredient.v2`）引用的成分清单中存在被编辑的声明时，对该声明将失败（详见第15.11.3节"成分声明验证"），因为仅`c2pa.ingredient.`v3声明支持声明签名哈希验证方法（详见第15.11.3.3.1节"声明签名哈希验证方法"）。

6.9. 声明中的时间规范

断言中日期和/或时间值的默认规范采用日期/时间格式，在CBOR中序列化为标签编号0（参见RFC 8949第3.4.1节），在CDDL中以`tdate`类型表示。

存在一种情况，即在添加声明的签名时间时，该时间以特殊类型的CBOR日期/时间表示。

此外还存在时间戳声明，其采用签名流程中所述的标准时间戳格式。

日期和时间采用不同表示形式的原因在于，根据现有标准的使用情况，为每个具体用例选择最合适的表示方式。

第7章 数据框

重要提示	本节保留以供历史参考。数据框的概念已被已弃用，转而采用标准断言，该断言使用标准JUMBF嵌入文件内容类型框来包含数据。更多信息请参阅[_data_box]。

7.1. 通用

数据框提供了一种将任意数据纳入C2PA清单的方法，这些数据可被断言引用，而非直接作为二进制字符串嵌入断言字段。这些数据框存储于数据框存储库中，每个数据框均为单个CBOR内容类型框（`cbor`）。

数据框的数据直接作为数据字段的值提供，该字段为字符串类型（`bstr`），因此可承载任意二进制数据。数据类型需通过`dc:format`字段使用标准IANA媒体类型进行标识。

注	IANA 结构化 后缀 (https://www.iana.org/assignments/media-type-structured-suffix/media-type-structured-suffix.xhtml)，例如 `+json` 和 +zip，也可作为`dc:format`字段的取值。

有时，为更清晰地说明数据的格式和用途，可能需要将一种或多种资产类型作为`data_types`字段的值提供。

数据框应标注为`c2pa.data`，并遵循断言标签关于多重实例的规则。

7.2. 模式与示例

此类型的模式由CDDL中数据框的CDDL定义中的`data-box`-map规则定义：

数据框的CDDL

```
; 用于存储任意数据的框

data-box-map = {
   "dc:format": 格式字符串, ; 数据的IANA媒体类型 "data" : 字符串, ; 任意文本/二进制数据
   ? "data_types": [1* $asset-type-map], ; 数据类型的附加信息
}
```

第8章. 唯一标识符

8.1. 唯一标识C2PA清单与资产

每个C2PA清单均通过统一资源名称RFC 8141（来自c2pa URN命名空间的URN）进行唯一标识与引用，而C2PA资产则通过其有效清单的c2pa URN值实现唯一标识。C2PA URN的ABNF规范详见《C2PA URN的ABNF描述》。

c2pa URN应包含两个必填组件和两个可选组件，按下列顺序排列，各部分间以冒号(:)分隔。

- URN标识符（urn:c2pa）：必需。
- UUID v4（按RFC 9562第4节的字符串表示形式）：必填。
- 声明生成器标识符字符串：可选。
- 版本与原因字符串（如下所述）：可选。

当存在时，"声明生成器标识符"字符串应包含不超过32个ASCII字符（符合RFC 20规范），且不包含控制字符（RFC 20, 5.2）或图形字符（RFC 20, 5.3）。

当存在时，"版本与原因"字符串应由一个正整数组成，后跟一个下划线（_），再接另一个正整数。这些值的具体含义及其使用方式详见《冲突导致的版本声明》。此外，当存在"版本与原因"字符串时，"声明生成器标识符"字符串也必须存在，但该字符串可以为空。

C2PA URN的ABNF语法

```
c2pa_urn = c2pa-namespace UUID [claim-generator [version-reason]]c2pa-namespace =
"urn:c2pa:"

; 此定义取自RFC 9562 UUID      = 4个十六进制字节 "-"
                2十六进制字节 "-"
                2十六进制字节 "-"
                2十六进制字节 "-"
                6十六进制字节
十六进制字节 = 十六进制数字 十六进制
数字 数字     = %x30-39
HEXDIG       = 数字 / "A" / "B" / "C" / "D" / "E" / "F"

; ASCII 字符，但不包括控制字符或图形字符可见字符（空格除外） = %x21-7E / %x80-FF

; 声明生成器标识符是由0到32个可见字符（除空格外）组成的字符串
; 这意味着空字符串也是有效的
索引生成器 = ":" 索引生成器标识符
声明生成器标识符 = 0*32可见字符（除空格外）

; version-reason 是一个由正整数组成的字符串
; 随后是一个下划线和一个正整数
```

```
版本-原因 = ":" 版本号 "_" 原因 版本号 = 1*数字
reason = 1*DIGIT
```

示例：

- urn:c2pa:F9168C5E-CEB2-4FAA-B6BF-329BF39FA1E4
- urn:c2pa:F9168C5E-CEB2-4FAA-B6BF-329BF39FA1E4:acme
- urn:c2pa:F9168C5E-CEB2-4FAA-B6BF-329BF39FA1E4:acme:2_1
- urn:c2pa:F9168C5E-CEB2-4FAA-B6BF-329BF39FA1E4::2_1

注释	本规范的先前版本使用 RFC 9562、UUID URN，且声明的标识符为 在URN开头添加生成器。然而，这种做法被发现既不符合RFC 9562（UUID规范），也不符合RFC 8141（统一资源名称规范）。

该c2pa URN标识符应用于C2PA工作流的多个环节，例如在衍生或组合资产中识别构成该资产的原始资产时。

8.2. 因冲突导致的版本声明

在某些情况下，可能需要因标识符冲突而重新标记C2PA清单。例如，当声明生成器已将某成分清单添加至资产的C2PA清单存储库后，又添加了另一种成分——该成分清单在存储库中具有相同标签，但因某项断言值被修改等原因导致清单内容发生变更。此时需将修改后的成分清单副本复制至资产的C2PA清单存储库，并进行重新标记。

重新标记清单：

- 若当前统一资源名称（URN）未包含"声明生成器标识符字符串"，则声明生成器应在末尾追加
  :.
- 在所有情况下，声明生成器应在 URN 后附加冒号 (:)，接着是一个从 1 开始的单调递增整数，随后是一个下划线 (_)，最后跟随下方列表中代表重新标记原因的整数。
  - 1: 与其他C2PA清单冲突

例如，若声明生成器因冲突需第二次重新标记C2PA清单，则附加字符串为: :2_1。

8.3. 识别非C2PA资产

对于不含C2PA清单但包含嵌入式XMP的资产（该XMP遵循XMP规范第2部分2.2节定义的xmpMM:DocumentID和/或xmpMM:InstanceID字段），应使用这些值作为资产标识符。

当处理不包含C2PA清单且未嵌入XMP的资产时，声明生成器可采用任意方法为其提供唯一标识符。

8.4. URI 引用

8.4.1. 标准URI

清单中所有信息引用（无论存储于资产内部（即嵌入式）或外部（如云端）），均应通过ISO 19566-5:2023标准C.2节定义的JUMBF URI引用实现。此类URI通常作为hashed_uri或hashed_ext_uri数据结构的组成部分使用。

当引用的是压缩清单时，JUMBF URI不应包含任何关于brob盒的信息，但清单的URI将被视为未压缩清单进行处理。这意味着该URI将包含c2ma或c2um盒的标签，但不包含c2cm盒的标签。此外，指向压缩清单的URI引用不应包含brob盒的标签——仅包含压缩清单本身的标签。

在解析内部JUMBF URI引用时，若路径中存在因多个子框具有相同标签而导致的歧义标签，验证器应将该引用视为未解析。

8.4.2. 哈希URI

8.4.2.1. 嵌入式

当 URI 用于同一 C2PA 清单存储中的嵌入项时，应使用 hashed_uri。

本规范为采用CDDL定义的架构提供了等效的哈希URI映射数据结构（哈希URI采用CDDL授权）：

哈希URI的CDDL

```
; 该数据结构用于存储同一JUMBF内URL的引用及其哈希值。此处采用插件机制，使哈希URI映射可在未定义映射的独立文件中使用
$hashed-uri-map /= {
   "url": jumbf-uri-type, ; JUMBF URI引用
   ? "alg": tstr .size (1..max-tstr-length), ; 用于计算本声明中所有哈希值的密码哈希算法标识符字符串，取自C2PA哈希算法标识符列表。若该字段缺失，则从外围结构中采用该结构定义的哈希算法。若两者均存在，则使用本结构中的字段。若所有位置均无值，则该结构无效；无默认值。
   "hash": bstr, ; 包含哈希值的字节字符串
}

; CBOR头部(#)和尾部($)已在正则表达式中引入，故无需显式指定 jumbf-uri-type /= tstr .regexp
"self#jumbf=[\\w\\d\/][\\w\\d\\.\/:-]+[\\w\\d]"
```

由于断言存储必须与引用它们的声明位于同一C2PA清单框中，因此仅允许

self#jumbf URI 才被允许。这些 self#jumbf URI 可相对于整个 C2PA 声明存储，

此时必须以斜杠 (U+002F) 开头，或相对于当前 C2PA 清单。URI 不得包含 `..`（两个 U+002E 字符）序列。

示例1. `self#jumbf` URI示例

以下为有效的 `self#jumbf` URI 示例：

- `self#jumbf=/c2pa/urn:c2pa:F095F30E-6CD5-4BF7-8C44-CE8420CA9FB7/c2pa.assertions/c2pa.thumbnail.claim` 相对于整个存储库（因其以 `/` 开头）；
- `self#jumbf=c2pa.assertions/c2pa.thumbnail.claim` 则相对包含该 URI 的容器清单而言。

8.4.2.2. 外部

当引用C2PA清单存储库外部的资源时，采用哈希外部URI映射数据结构。该结构是哈希URI的变体，其引用对象为外部URI而非`self#jumbf`。哈希外部URI数据结构由以下CDDL中的规则定义：

哈希外部URI的CDDL

```
; 用于存储外部URL及其哈希值的引用数据结构。
; 我们在此使用插件机制，以便在独立文件中使用哈希扩展URI映射
; 无需在同一文件中定义映射
$hashed-ext-uri-map /= {
   "url": ext-url-type, ; http/https URI 引用
   "alg": tstr .size (1..max-tstr-length), ; 用于计算此URI数据哈希值的密码哈希算法标识符字符串，取自C2PA哈希算法标识符列表。与其他类型的alg字段不同，此字段在此处为必填项。
   "hash": bstr, ; 包含哈希值的字节字符串
   ? "dc:format": 格式字符串, ; 数据的IANA媒体类型
   ? "size": 数据类型, ; 数据的字节数
   ? "data_types": [1* $asset-type-map], ; 数据类型的附加信息
}

; CBOR头部(#)和尾部($)已在正则表达式中引入，故无需显式标注
ext-url-type /= tstr .regexp "https?:\/\/[-a-zA-Z0-9@:%._\\+~#=]{2,256}\\.[a-z]{2,6}\\b[-a-zA-Z0-9@:%_\\+.~#?&//=]*"
```

重要提示

遵循通用规范，建议使用 `https` 协议获取断言数据以保护传输过程中的数据隐私，但 `http` 协议亦被允许——因数据完整性由哈希字段保障，且隐私保护并非所有场景的必要要求。包含外部 URI 的清单作者应根据实际需求选择协议。

可选的`dc:format`字段若存在，可作为http(s)头部Content-Type字段的替代方案。若存在该字段，则必须在任何内容协商/请求过程中将其作为所需格式进行检索。

有时，为更清晰说明数据格式及用途，可能需要在`data_types`字段中提供一种或多种资源类型作为其值。

还提供了一个可选的大小字段，用于指定要检索的数据大小。这可作为验证程序的提示信息，在哈希值之外提供额外参考。

注 该字段可用于指示是否执行下载操作、验证操作或两者兼施。

8.4.2.3. 哈希JUMBF框

创建指向任何JUMBF框（例如断言框和数据框）的URI引用时，应针对结构的JUMBF超级框内容执行哈希运算。该超级框包含JUMBF描述框及其内部的所有内容框（但不包含结构的JUMBF超级框头部）。

注 有关哈希运算的更多细节，请参阅第13.1节“哈希运算”。

如最新版本的JUMBF（ISO 19566-5:2023）所述，并如图4"示例`c2pa.actions`断言"所示，任何JUMBF描述框中均可包含新的私有字段。本C2PA规范将C2PA盐定义为私有字段，其值为包含以下内容的标准框：

- 框长度（LBox，4字节大端无符号整数）；
- 框类型（TBox，4字节大端无符号整数，取值为`c2sh`（代表C2PA盐哈希））；
- 有效载荷数据（由随机生成的二进制数据组成，长度为16或32字节）。

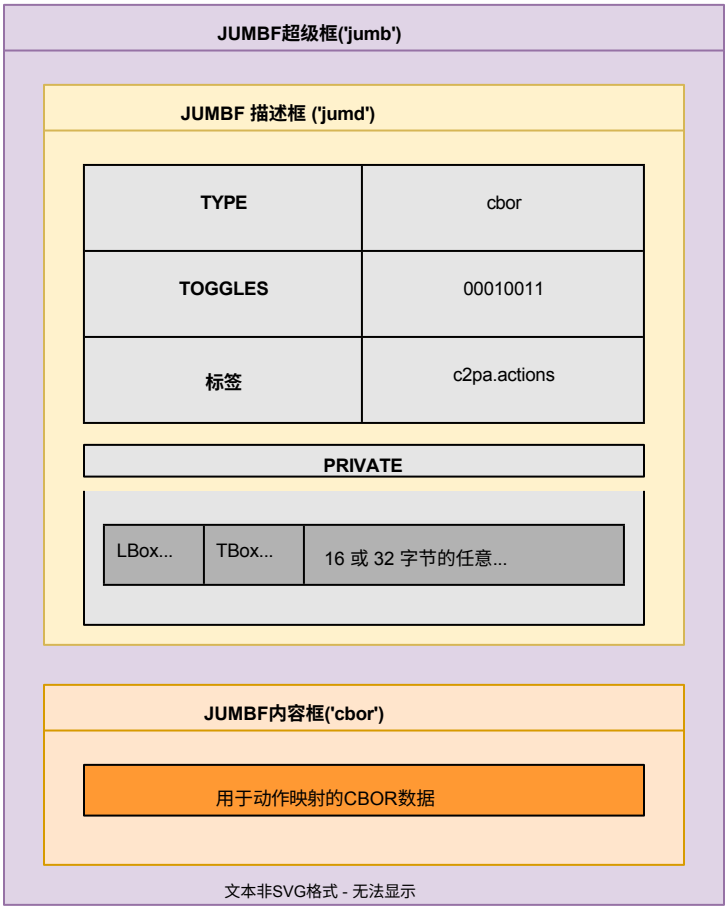

图4. c2pa.actions断言示例

第九章 内容绑定

9.1. 概述

标准C2PA清单的关键特性在于包含一个或多个称为内容绑定（content bindings）的数据结构，这些结构能唯一标识资产的特定部分。C2PA支持两种绑定类型：硬绑定（hard bindings）和软绑定（soft bindings）。硬绑定（也称为加密绑定）通过确定仅与该资产匹配的值（甚至不包括由此资产衍生出的其他资产或由此资产生成的呈现版本），使验证者能够确保：(a) 此清单属于该资产，以及 (b) 该资产未被修改。软绑定是从资产的数字内容（而非原始位）计算得出的。软绑定适用于识别衍生资产及资产呈现形式。

单个清单中不得包含超过一个定义硬绑定的断言，但可包含零个或多个定义软绑定的断言。

9.2. 硬绑定

9.2.1. 基于字节范围的哈希算法

用于检测篡改的最简单硬性绑定方式，是如第13.1节"哈希处理"所述，对资产全部或部分字节应用加密哈希算法。此方法适用于任何类型资产，但仅应考虑用于不支持基于框式哈希的格式。

采用这种硬绑定形式时，需通过数据哈希断言来定义哈希处理的字节范围（以及未处理的字节范围）。由于数据哈希断言定义的是字节范围，因此具备足够的灵活性，无论资产是单个二进制文件还是由多个块或片段组成，均可适用。

9.2.2. 使用通用框哈希进行哈希计算

当资产格式为非BMFF基箱格式（如JPEG、PNG、GIF或其他此处列出的格式）时，应使用通用箱哈希断言。该断言由结构体数组构成，每个结构体包含一个或多个盒子（通过名称/标识符标识），以及覆盖这些盒子数据（及文件中可能存在的间隔数据）的哈希值，同时注明所用哈希算法。

9.2.3. 对BMFF格式资产进行哈希处理

若资源基于ISO BMFF格式，则可改用针对盒式结构优化的硬绑定（称为BMFF基哈希断言）。

对于整体验证的单体MP4文件资产（即mdat框作为整体进行验证），其断言验证方式与数据哈希断言几乎完全相同。区别仅在于：它使用框排除列表而非字节范围来定义哈希计算的字节范围（以及不进行哈希计算的范围）。

对于碎片化MP4（fMP4）文件，断言本身需结合特定于数据块的哈希信息，该信息的位置遵循A.5节"将清单嵌入基于BMFF的资产"中的规定。

9.2.4. 集合哈希处理

在工作流中，当C2PA清单引用的是资产集合而非单个资产时，应使用集合数据哈希断言作为指定集合内资产硬绑定的方法。

注 例如，集合数据哈希断言可用于描述AI/ML模型训练数据集中的每个文件夹。

9.2.5. 资产元数据绑定

声明生成器可将资产元数据（即C2PA清单外的元数据，如EXIF或XMP）排除在内容绑定之外。为此，应采用适用于数据哈希声明、通用框哈希声明或基于BMFF的哈希声明的相应排除机制。

注 被排除的资产元数据不归属于签名者。

任何符合附录B《*c2pa.metadata实现细节*》所述通用元数据断言规范、且可由签名者断言的资产元数据值，均应复制至该断言中并纳入C2PA清单。

9.3. 软绑定

9.3.1. 通用

软绑定通过软绑定断言来描述，例如从数字内容计算出的指纹或嵌入数字内容中的不可见水印。这些软绑定即使底层位元不同，也能实现数字内容的匹配。

注 例如不同分辨率或编码格式的资产版本。

此外，若资产的C2PA清单被移除，但该清单副本仍存于其他来源存储库中，则可通过现有软绑定机制实现清单与资产的匹配。

由于它们具有不同的用途，软绑定不得用作硬绑定。

9.3.2. 允许的软绑定算法列表

所有软绑定均应采用本规范支持的软绑定算法列表中列出的算法之一生成。

第10章 声明

10.1. 概述

声明汇集了特定时间点关于某项资产的所有断言，包括用于绑定内容的断言集。随后，该声明将按照第10.3.2.4节“声明签名”所述进行加密哈希处理并签名。声明具备与断言完全相同的属性（包括被分配标签`c2pa.claim.v2`），但不支持使用断言元数据。声明以CBOR数据形式编码，因此必须符合CBOR核心确定性编码要求（参见RFC 8949第4.2.1节）。

注意 先前版本支持在声明中使用断言元数据，但该功能现已弃用。

本规范的早期版本曾使用标签 `c2pa.claim` 及关联的声明映射来表示该声明，但现已弃用。验证器仍应接受该标签（及关联声明映射），但声明生成器不得再生成此类声明。

10.2. 语法

10.2.1. 模式

此类型的模式由以下CDDL定义中的`claim`-map-v2和claim-map规则定义，分别对应标签为`c2pa.claim.v2`和`c2pa.claim`的声明：

```
; C2PA声明映射的CDDL模式claim-map = {
   "声明生成器": tstr, ; 用户代理字符串，格式遵循 http://tools.ietf.org/html/rfc7231#section-5.5.3 规范，用于包含创建声明的声明生成器的名称和版本
   "claim_generator_info": [1* generator-info-map],
   "签名": jumbf-uri-type, ; 指向本声明签名的JUMBF URI引用"断言": [1* $hashed-uri-map],
   "dc:format": tstr, ; 资产的媒体类型
   "instanceID": tstr .size (1..max-tstr-length), ; 唯一标识资产特定版本
   ? "dc:title": tstr .size (1..max-tstr-length), ; 资产名称
   ? "redacted_assertions": [1* jumbf-uri-type], ; 被编辑的组件清单声明的JUMBF URI引用列表
   ? "alg": tstr .size (1..max-tstr-length), ; 用于标识计算本声明中所有数据哈希断言所采用的密码哈希算法的字符串（除非另有覆盖），取自C2PA数据哈希算法标识符注册表。该字符串为本声明中data-hash和hashed-uri结构的'alg'字段提供值
   ? "alg_soft": tstr .size (1..max-tstr-length), ; 除非另有覆盖，否则用于标识计算本声明中所有软绑定断言所用算法的字符串，取自C2PA软绑定算法标识符注册表。
   ? "元数据": $断言元数据映射, ; 断言的附加信息
}

; C2PA声明映射的CDDL模式 claim-map-v2 = {
```

```
  "instanceID": tstr .size (1..max-tstr-length), ; 唯一标识资产特定版本
  "声明生成器信息": $生成器信息映射, ; 本声明的生成器 "签名": jumbf-uri-类型, ; 指向本声明签名的JUMBF URI引用 "创建的声明": [1* $哈
  希URI映射],
  ? "gather_assertions": [1* $hashed-uri-map],
  ? "dc:title": tstr .size (1..max-tstr-length), ; 资产名称,
  ? "redacted_assertions": [1* jumbf-uri-type], ; 被编辑的组件清单断言所引用的JUMBF URI列表
  ? "alg": tstr .size (1..max-tstr-length), ; 用于标识计算本声明中所有数据哈希断言所用密码哈希算法的字符串（除非另有覆盖），取自C2PA数
据哈希算法标识符注册表。该值将填入本声明中data-hash和hashed-uri结构的'alg'字段
  ? "alg_soft": tstr .size (1..max-tstr-length), ; 用于标识计算本声明中所有软绑定断言所用算法的字符串（除非另有覆盖），取自C2PA
软绑定算法标识符注册表。
  ? "元数据": $断言元数据映射, ; （已弃用）断言的附加信息
}

生成器信息映射 = {
  "name": tstr.size(1..max-tstr-length), ; 可读字符串，用于命名声明生成器
  ? "version": tstr, ; 产品版本的人类可读字符串
  ? "图标": $哈希URI映射 / $哈希扩展URI映射, ; 图标的哈希URI（可为内嵌或远程）
  ? "操作系统": tstr, ; 声明生成器运行操作系统的可读字符串
  * tstr => any
}
```

CBOR诊断表示法（RFC 8949第8节）中的声明映射v2结构示例：

```
{
  "alg" : "sha256", "claim_generator_info"
  : {
    "名称": "乔的图片编辑器", "版本": "2.0",
    "操作系统": "Windows 10"
  },
  "签名" : "self#jumbf=c2pa.signature","创建断言" : [
    {
      "url": "self#jumbf=c2pa.assertions/c2pa.hash.data", "hash":
      b64'U9Gyz05tmpftkoEYP6XYNsMnUbnS/KcktAg2vv7n1n8='
    },
    {
      "url": "self#jumbf=c2pa.assertions/c2pa.thumbnail.claim", "hash":
      b64'G5hfJwYeWTlflxOhmfCO9xDAK52aKQ+YbKNhRZeq92c='
    },
    {
      "url": "self#jumbf=c2pa.assertions/c2pa.ingredient.v3", "hash":
      b64'Yzag4o5jO4xPyfANVtw7ETlbFSWZNfeM78qbSi8Abkk='
    }
  ],
  "redacted_assertions" : [ "self#jumbf=/c2pa/urn:c2pa:5E7B01FC-4932-
    4BAB-AB32-
D4F12A8AA322/c2pa.assertions/c2pa.metadata"
  ]
}
```

10.2.2. 字段

若存在，dc:title的值应为资产的人类可读名称。

注 c2pa.claim 包含的 dc:format 字段在 c2pa.claim.v2 中已不再存在。

若资产包含XMP，则应使用该资产的xmpMM:InstanceID作为实例ID。当无XMP可用时，应采用资产的其他唯一标识符作为实例ID的值。

注意 某些字段名称（如dc:title）采用命名空间前缀，其名称与定义直接源自XMP标准。但在C2PA中使用这些字段时，无需依赖XMP。

签名字段必须存在，且应包含指向声明签名的URI引用。

created_assertions 字段必须存在，且应包含一个或多个指向本声明所作断言的 URI 引用。在标准清单中，该字段至少应包含一个代表硬性约束的断言引用和一个操作断言引用。

注 所有创建的断言均归属于签名者，因信任模型根植于对签名者的信任。

当存在时，gather_assertions字段应包含一个或多个URI引用，指向工作流中其他组件提供给声明生成器的断言。

注 通过将断言放入此列表，声明生成器声明该断言属于声明的一部分，但其来源并非声明生成器且不归属于签名者。例如，包含人工操作者输入信息的断言将被列入gather_assertions。

当存在时，redacted_assertions 字段应包含指向一个或多个已编辑断言的 URI 引用。

10.2.3. 声明生成器信息

10.2.3.1. 通用信息

关于声明生成器的详细信息应作为claim_generator_info的值呈现。清单消费者应使用claim_generator_info的值来确定声明生成器的相关信息，供自身使用或在用户界面中展示。

注 c2pa.claim 包含一个 claim_generator 字段，其值为简单字符串，该字段在 c2pa.claim.v2 中已不再存在。

10.2.3.2. 生成器信息映射

添加 claim_generator_info 字段时，其值应为 generator-info-map 对象，该对象必须包含name字段。还可包含version字段或icon字段（或两者兼有）。此外，允许添加其他字段

允许包含其他字段，需遵循第6.2.1节"命名空间"所述的标准实体特定命名空间规则。该对象的数据应代表实际生成声明的非人类（硬件或软件）主体（即声明生成器本身）。

声明生成器可能希望向呈现用户体验的清单消费者提供其自身的图形化表示，此处称为图标。若存在图标字段，其值应为哈希URI。该哈希URI应指向嵌入式数据断言，其标签为`c2pa.icon`，并遵循断言标签关于多重实例的规则。清单消费者还应支持本规范早期版本推荐的数据框方案。

注 与断言数组类似，哈希URI所用的哈希算法由哈希URI中的`alg`字段决定；若该字段缺失，则采用声明中的`alg`字段值。

使用声明生成器信息的示例

```
{
      "claim_generator_info" : { "name": "Joe's
            Photo Editor", "version": "2.0",
            "操作系统": "Windows 10","图标": {
                  "url": "http://cdn.examplephotoagency.com/logo.svg", "hash":
                  "5bdec8169b4e4484b79aba44cee5c6bd"
            }
      }
}
```

10.3. 创建声明

10.3.1. 创建断言

在声明最终确定之前，需创建所有断言并存储于新创建的C2PA断言存储库中（详见本文后续说明）。

创建标准清单时，可能无法在声明创建时获取所有必需的绑定信息，此时应采用多步处理方法：先完成基础设置，后续再补充信息。

10.3.2. 声明准备

10.3.2.1. 添加断言与编辑

声明应包含created_assertions字段，并可包含gathered_assertions字段。这两个字段的组合值构成声明所"主张"的所有断言的URI引用列表，这些断言已添加至断言存储库。在标准清单中，`created_assertions`字段的值必须至少包含一个代表硬性绑定的断言。

若成分声明中的任何断言被删除，其URI引用应添加至列表中，该列表即为`deleted_assertions`字段的值。

10.3.2.2. 添加成分

在许多创作场景中，创作者并非创建全新的资产，而是引入其他现有资产作为创作基础——这些资产可以是衍生资产、组合资产或资产呈现形式。这些现有资产被称为"素材组件"，其使用情况通过"素材组件声明"记录在溯源数据中。

当某个组件包含一个或多个C2PA清单时，这些清单应插入该资产的C2PA清单存储库，以确保来源数据完整无损。此类组件清单将按第11.1.4节"C2PA盒子详情"所述方式添加至JUMBF。若C2PA清单存储库中已存在具有相同唯一标识符的清单，则需通过哈希比对两者。若完全一致，则忽略新清单；若存在差异，则需按第8章《*唯一标识符*》所述修改新清单的唯一标识符后再行添加。

若原料清单为远程存储，且声明生成器无法获取该清单，应使用错误代码 `manifest.inaccessible` 反映此情况。

10.3.2.3. 签名关联机制

签名不能作为签名负载的一部分，但由于其标签是预定义的，因此完整的 URI 引用也是已知的。因此，我们可以将该 URI 引用包含在声明中，方法是将声明的签名字段值设置为该 URI 引用。

注	此操作实现了声明与其签名的显式绑定。

10.3.2.4. 声明签名

签名的生成方法详见第13.2节“数字签名”。

对于标准清单和更新清单两种类型，`Sig_structure`的有效负载字段应为声明文档的序列化CBOR，并采用分离内容模式。

数字签名流程生成的序列化`COSE_Sign1_Tagged`结构将写入C2PA声明签名框。

10.3.2.5. 时间戳

10.3.2.5.1. 采用RFC 3161规范

若条件允许，声明生成器应使用符合RFC 3161标准的时间戳机构(TSA)获取可信时间戳，以证明签名本身确实在特定日期和时间存在，并将该时间戳作为副签名纳入`COSE_Sign1_Tag`ged结构中。

建议声明生成器获取并包含时间戳，以确保其清单保持有效。

如第15章《*验证*》所述，未包含时间戳的清单文件在签名凭证过期或被撤销时即失效。清单文件仅可包含一个时间戳。

注 本规范先前版本允许清单包含多个时间戳。

10.3.2.5.2. 选择有效负载

本规范的先前版本在时间戳的有效载荷字段中使用了与第10.3.2.4节"声明签名"中`Sig_signature`相同的值。此类有效载荷现称为"v1时间戳中的v1有效载荷"，并被视为已弃用。声明生成器不得创建此类有效载荷，但验证器若发现存在则应进行处理。

"v2时间戳"的"v2有效负载"即为第10.3.2.4节"声明签名"中创建的COSE_Sign1_Tagged结构的签名字段值。执行时间戳操作的声明生成器应使用"v2有效负载"。

注 签名字段的值包含整个序列化字符串（`bstr`），包括标识主类型和长度的字节（不仅限于字符串本身）。

10.3.2.5.3. 获取时间戳

所有时间戳均应按照RFC 3161所述方式获取，并满足以下附加要求：

- TimeStampReq结构（RFC 3161第2.4.1节）的MessageImprint应通过创建RFC 8152第4.4节中的ToBeSigned值计算得出，其中`Sig_structure`元素的值如下：
  - 上下文元素应为`CounterSignature`。
  - 有效负载元素应为第10.3.2.5.2节"选择有效负载"所述的值。
  - `Sig_structure`其余元素的定义参见第13.2.3节"计算签名"。
- 随后，使用TSA支持的第13.1节“哈希处理”中允许列表中的哈希算法对ToBeSigned值进行哈希处理，并将该哈希算法与值放入`MessageImprint`中。若TSA不支持允许列表中的任何哈希算法，则不可用于时间戳处理。
  - 在可能的情况下，哈希算法应与声明数字签名所用的哈希算法保持一致。
- 在向 TSA 发送请求时，应设置 `TimeStampReq` 结构的 `certReq` 布尔值，以确保响应中包含其证书链。

10.3.2.5.4. 时间戳存储

v1时间戳（已弃用）存储于COSE未受保护的标头中，其标签为字符串`sigTst`。若存在该标头，其值应为示例2“`tstContainer`的CDDL”中定义的`tstContainer`。从TSA接收的TimeStampResp结构内容应存储为`tstTokens`元素的val属性值。

v2时间戳应存储于COSE未受保护的头部中，其标签为字符串`sigTst2`。当存在时，该头部的值应为示例2“`tstContainer`的CDDL”中定义的`tstContainer`。从 TSA 接收的 `TimeStampResp` 结构中 `timeStampToken` 字段的值，应存储为 `tstTokens` 元素的 `val` 属性值。该值应采用 DER 编码的 RFC 3161 `TimeStampToken` 格式，并封装在 CBOR 字节字符串中。

注

v2时间戳等同于RFC 3161时间戳中COSE头参数的"CTT"模型。

戳令牌草案。该模式要求在进行时间戳记前必须完成完整的签名结构，从而使时间戳能够作为整个签名结构（包括实际证书）的副签名。

若未包含时间戳，则COSE未受保护的头部中不得出现任何时间戳字段（sigTst或`sigTst2`）。

示例2. `tstContainer` 的CDDL

```
; 基于JSON模式的tstContainer及相关结构的CBOR版本，详见
; https://forge.etsi.org/rep/esi/x19_182_JAdES/raw/v1.1.1/19182-jsonSchema.json tstContainer = {
   "tstTokens": [1* tstToken]
}

tstToken = {"val":
   bstr
}
```

注释

上述定义是对JAdES第5.3.4节及其JSON模式中子集的CBOR适配版本，但进行了如下修改：`val`字段的内容为字节字符串而非Base64编码字符串。

10.3.2.6. 凭证撤销信息

若签名者的凭证支持查询其在线凭证状态，且该凭证包含指向提供带时间戳凭证状态信息的服务的指针，则声明生成器应查询该服务，捕获响应，并按信任模型中对凭证的描述方式进行存储。若凭证撤销信息以这种方式附加，则签名后还应获取可信时间戳，具体操作参见第10.3.2.5节"时间戳"。

10.3.3. 索赔示例

10.3.3.1. 单一声明

以下是包含单个声明的图像的可视化表示，该声明内嵌有多个断言。

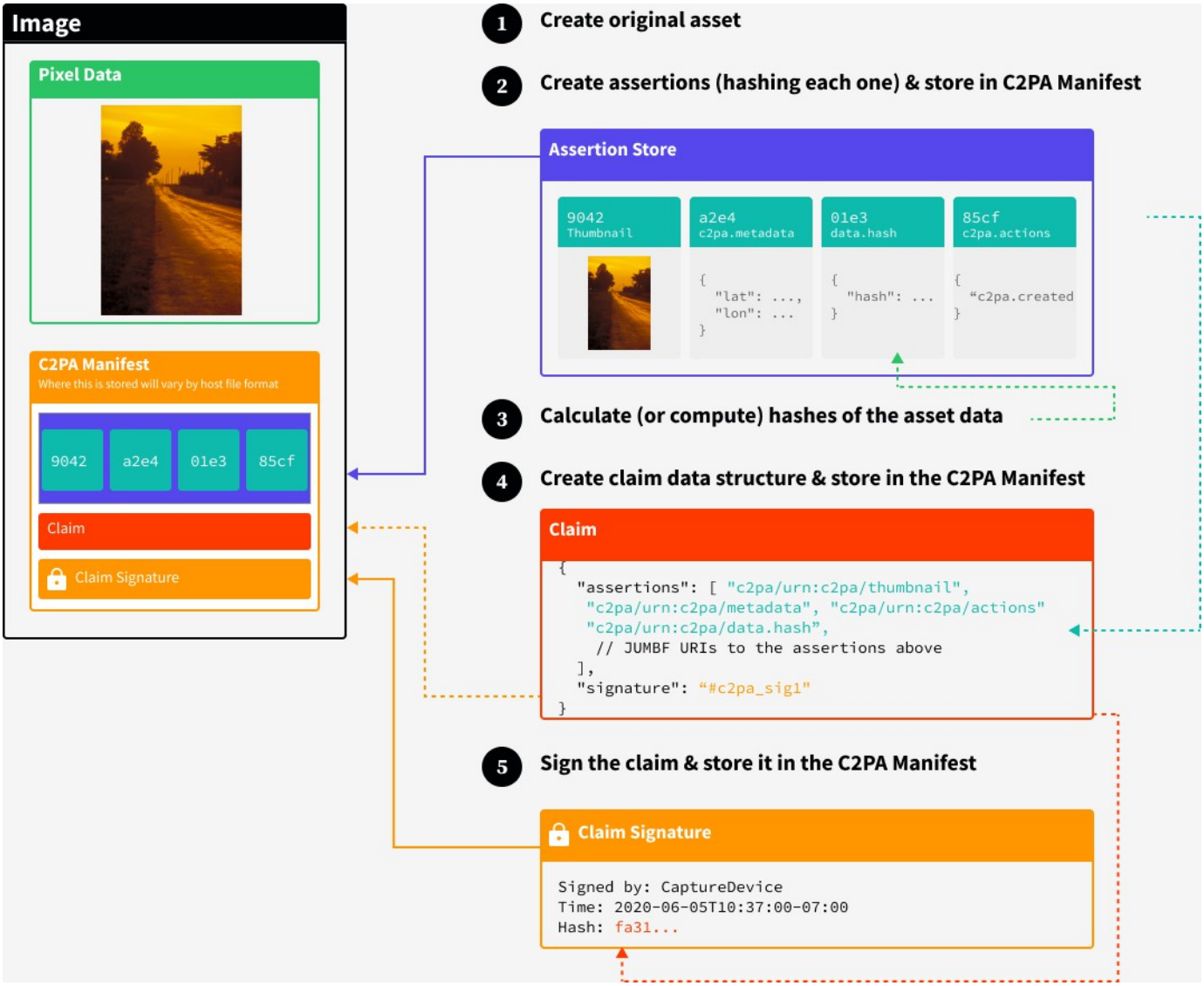

图5. 包含断言的单一主张

10.3.3.2. 多重声明

在此示例中，针对前例创建第二个声明时，已从原始声明中删除了其中一项断言。该场景的可视化呈现如下：

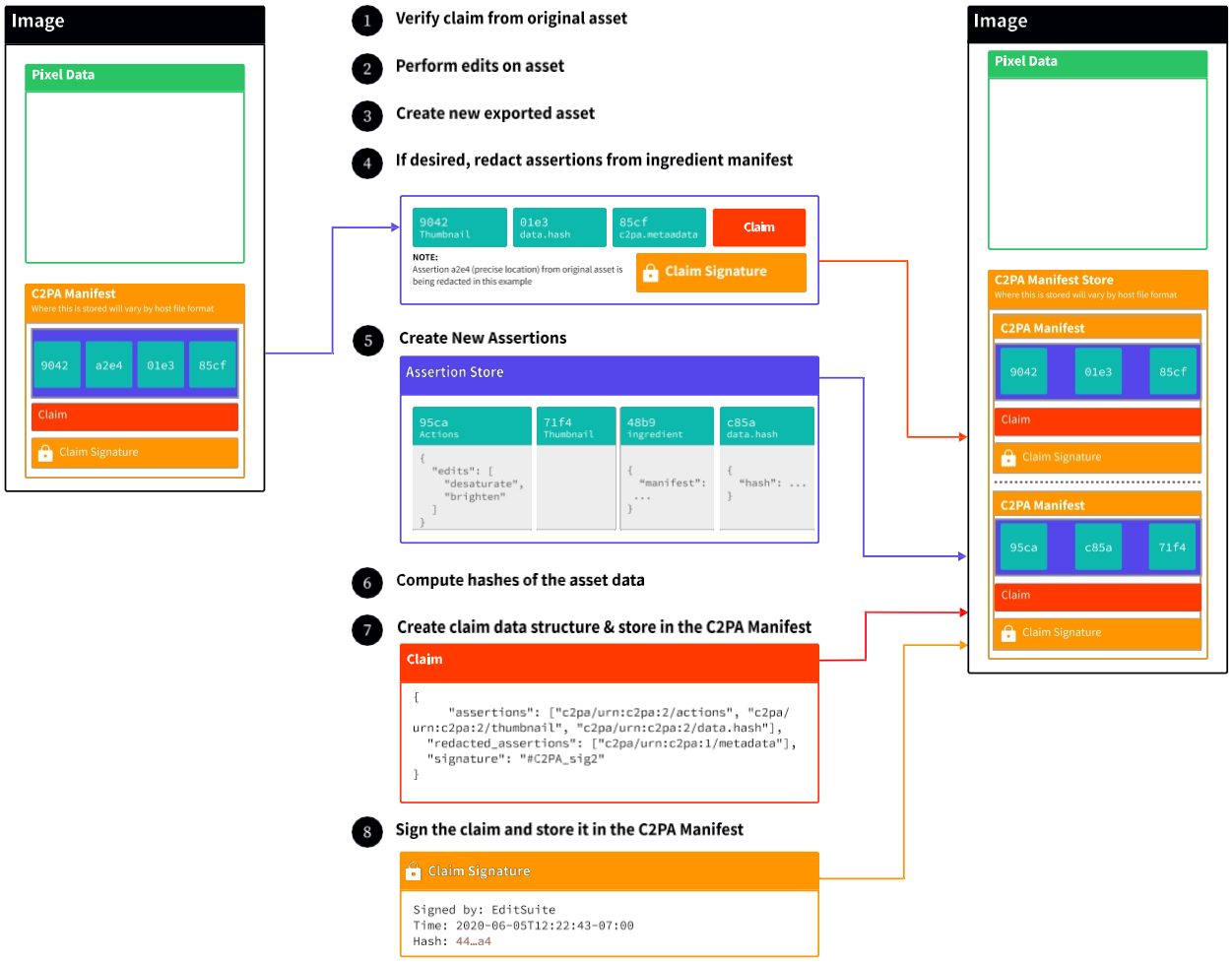

图6. 次级声明中删除断言

10.4. 多步处理

某些资产文件格式要求在签名清单之前固定C2PA清单存储库和资产内容的文件偏移量，以确保内容绑定能与所验证的内容正确对齐。遗憾的是，清单及其签名的大小在签名完成前无法精确知晓，这可能导致文件偏移量发生变化。

以JPEG 1文件为例，整个C2PA清单存储必须出现在图像数据之前，因此其大小将影响被验证内容的文件偏移量。

为实现此目标，应采取多步骤方法，类似于PDF中签名的处理方式。

10.4.1. 创建内容绑定

创建标准清单时，其声明应包含一个或多个内容绑定断言，以确保资产具有防篡改特性。

创建数据哈希断言并将其添加到断言存储库时，需考虑以下事项。

在许多情况下（例如JPEG 1格式），无法对资产进行完整哈希计算，因为清单会嵌入在文件中间位置，导致在计算资产哈希值时无法确定清单数据的大小或位置。通过允许在哈希过程中指定排除范围，可以避免这种循环依赖关系。当指定排除范围时，仅对未被任何排除规则覆盖的资产区域执行单次哈希计算。

若清单嵌入在JPEG 1文件APP11分段的中间位置，则声明创建者可将APP11分段排除在哈希计算之外。

为防止插入攻击，在可能的情况下应仅设置单一排除范围。当资源中清单的大小或位置（或两者）未知时，数据哈希断言中的起始值和长度值均应设为零，且填充值的大小应足够容纳二次写入过程中的数值。建议至少为16字节。填充密钥值应全部为0x00。

若采用填充机制，填充数据可能被修改而不导致验证失败。声明生成器应确保填充数据（或其他被排除的资产数据）的变更不会改变资产的解析方式。

注

对于JPEG 1文件，可通过消除填充或确保`JFIF APP11/C2PA`分段无法被缩短或转换为其他分段类型。此要求通过在所有包含清单的分段的数据哈希映射中包含所有C2PA清单分段头（APP11）和2字节长度字段来实现。此举可确保排除区域内任何数据变更不会被JPEG处理器误判。

10.4.2. 创建临时声明与签名

将新创建的数据哈希断言引用添加至声明的断言列表，提供临时哈希值（如空字符）。

至此临时声明已完成，可添加至正在创建的C2PA清单中。

由于当前声明仅为临时状态，因此无法对其签名。为确保声明签名框包含有效的CBOR结构，请参照RFC 8152第4.2节所述创建临时`COSE_Sign1_Tagged`结构。该结构由标签字节后接`COSE_Sign1`结构组成，后者为包含四个元素的CBOR数组。数组构造方式如下：

- 首元素为受保护的头部桶（参见RFC 8152第3节）。通过在此位置放置一个`bstr`字符串填充此位置。
- 第二个元素是未受保护的头部桶，它是一个CBOR映射。创建包含1对元素的映射：使用字符串填充作为标签，并将填充为零字节（0x00）的所需填充大小字符串作为值。建议初始填充大小为25千字节。
- 第三个元素是有效负载。在此处放置`nil`值（CBOR主类型7，值22）。

- 第四个元素是签名。在此处放置一个大小为0的bstr。

10.4.3. 完成C2PA清单

至此，构成该资产完整C2PA清单的所有模块均已完成，可（若尚未完成）构建为最终形态。该资产的C2PA清单将与所有组件清单合并，形成完整的C2PA清单存储库。活动清单必须作为C2PA清单存储库超级容器中的最后一个C2PA清单超级容器。随后可参照第11.3节"将清单嵌入各类文件格式"所述方法，将C2PA清单存储库嵌入资产。

10.4.4. 回溯并补充

现在C2PA清单存储已嵌入资产，其数据哈希断言中的起始偏移量和活动清单长度可进行更新。操作时必须保持断言框的大小不变，仅修改其数据内容。具体实现方式是调整填充字段的值，使其达到"填满"剩余字节所需的长度。

注： 推荐/确定性CBOR序列化中的填充采用可变长整数指定编码二进制数据的长度。当长度从0变为1字节，或从1变为2字节（依此类推）时，生成的填充长度会跳跃增加两个字节。这意味着并非所有填充都能通过单个填充字段。例如，可以创建24字节和26字节的填充，但无法创建25字节的填充。若出现此情况，可将所需填充量拆分至pad和pad2字段。例如：生成25字节填充时，声明生成器可将19字节编码至pad（生成20字节长度），另将4字节编码至pad2（生成5字节长度）。

数据哈希断言更新后，可对其进行哈希运算，并将哈希值写入先前用于存储位置的空闲区域。

此时声明已完整生成，可按第10.3.2.4节"声明签名"所述进行哈希计算与签名操作，生成的签名填充预分配空间。随后可按需缩减填充头部，确保声明签名框尺寸不变；由于该头部未受保护，修改其内容不会导致声明签名失效。

若序列化的COSE_Sign1_Tagged结构体超出C2PA声明签名框的预留大小，则应重复执行多步处理流程，并采用第10.4.2节"创建临时声明与签名"中选定的更大填充长度。若先前尝试中检索的撤销信息仍在有效期内（参见RFC 6960第4.2.2.1节），则可重复使用该信息；但新增声明需重新添加时间戳，且因填充操作导致文件偏移量变更。

C2PA清单可能包含本规范未定义的断言，且这些断言可能依赖文件布局。因此，声明生成器可能无法再修改数据哈希断言中的文件布局和/或偏移量。此时声明生成器应在创建断言前进行填充，确保断言最终确定后无需更改文件布局。

第11章. 清单

11.1. JUMBF的使用

11.1.1. 设计依据

为满足C2PA的诸多要求，C2PA清单需存储（序列化）至结构化二进制数据存储中，该存储需支持特定功能，包括：

- 支持在单一容器中存储多个清单（例如母清单与成分清单）。
- 通过统一资源标识符（URI）引用单个元素（涵盖清单内部及跨清单元素）。
- 能够明确标识元素中需进行哈希处理的部分。
- 支持存储C2PA预定义数据类型（如JSON和CBOR）。
- 支持存储任意数据格式（如XML、JPEG等）。

除支持上述所有要求外，我们选定的容器格式——ISO 19566-5:2023（JUMBF）——还原生支持JPEG格式家族，并与众多常见图像及视频文件格式采用的盒式模型（即ISOBMFF、ISO 14496-12）兼容。采用JUMBF格式既能获得所有相同优势（并额外支持URI引用等功能），又能兼容经典图像格式（如JPEG/JFIF、PNG）以及3D和文档格式（如PDF）。此序列化格式亦适用于原生不支持JUMBF的格式，或当C2PA清单存储与资产分离存储时（如独立文件或URI位置）。

注

鉴于多数标准断言及声明签名均采用CBOR序列化，虽曾考虑将整个C2PA清单采用CBOR格式，但最终未采纳该方案，因CBOR并非容器格式。

例如，要在CBOR内部存储一个"JSON数据块"，并确保其为JSON格式（而非其他格式），就需要专门设计用于存储此类数据的数据结构。随后还需定义父结构如何承载该数据结构。同样的概念也必须应用于JUMBF的每个原生特性。

虽然完全有可能将所有必需功能完全用CBOR重新实现，但这将耗费大量精力，且无法在所有实现中彻底消除对JUMBF/BMFF解析器的依赖。

11.1.2. 处理规则

C2PA清单消费者绝不应处理未包含在C2PA清单存储库中的断言、断言存储库、声明、声明签名或C2PA清单。此外，当C2PA清单消费者遇到其无法识别的JUMBF类型UUID的JUMBF盒或超级盒时，应跳过（并忽略）其内容。

注 这意味着C2PA清单消费者可处理已知的私有盒，但需忽略未知盒。

若任何JUMBF框或超级框的JUMBF描述框中同时设置了"*可请求*"和"*标签存在*"开关，则该框或超级框应在任何更新后的C2PA清单存储中予以保留。

注意 设置了这些开关的框旨在通过JUMBF URI进行引用，其移除可能导致下游工作流失败。

11.1.3. 扩展

11.1.3.1. 通用

本节描述本规范要求的JUMBF规范（ISO 19566-5:2023）扩展。

11.1.3.2. 压缩框

为支持清单压缩，C2PA新增了对brob内容框的支持。该框基于JPEG-XL（ISO/IEC 18181-2:2024）中的类似框，是一种内容框，其内容为标准清单或更新清单的Brotli压缩字节，具体描述详见压缩清单条款。该内容框的框标识符应为0x62726F62（'brob'）。

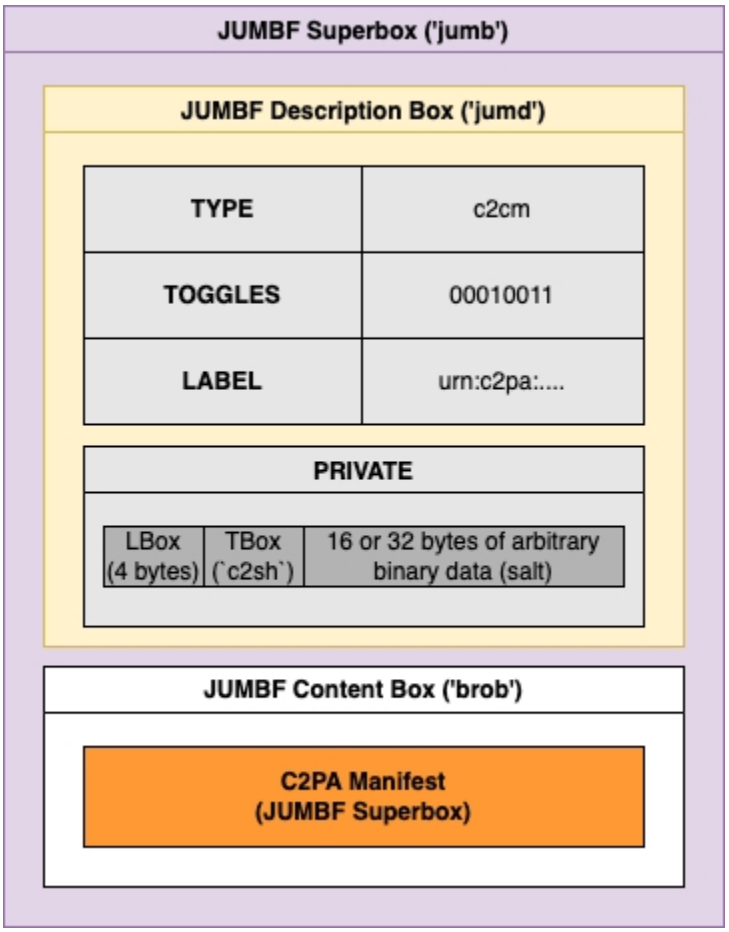

图7. 压缩清单示例

对压缩盒进行哈希处理的方式与其他盒子相同，具体方法详见第8.4.2.3节“对JUMBF盒子进行哈希处理”。

注

这意味着：当通过`activeManifest`字段从组件断言获取指向C2PA清单的hashed_uri引用时，其哈希值计算过程与任何其他JUMBF超级盒相同——即基于JUMBF描述盒和包含压缩有效负载的brob盒进行计算，但排除超级盒的头部。计算哈希值时无需先解压brob盒内容。

11.1.4. C2PA 盒子详情

11.1.4.1. JUMBF描述框

11.1.4.1.1. 标签

根据JUMBF规范（ISO 19566-5:2023, A.3）所述，标签应以UTF-8编码存储为ISO/IEC 10646字符。标签中禁止使用U+0000至U+001F（含）、U+007F至U+009F（含）范围内的字符，以及特殊字符'/'、';'、'?'和'#'。标签必须以空字符结尾。

作为JUMBF URI组成部分的标签，还不得使用字符U+FEFF、U+FFFF及U+D800-U+DFFF。

11.1.4.1.2. 开关

所有用于C2PA清单的JUMBF描述框（ISO 19566-5:2023, A.3）均需设置标签，此时应启用*标签存在*开关（`xxxxxx1x`）。此外，由于JUMBF URI在系统中用于引用各类数据框（如列举声明、原料引用等），应设置*可请求*开关（`xxxxxx11`）。

当按第8.4.2.3节"JUMBF框哈希处理"所述在PRIVATE框中包含盐值时，还需设置Private开关（`xxx1xxxx`）。

11.1.4.2. 清单存储

C2PA数据被序列化为符合JUMBF规范的盒式结构。最外层的盒被称为C2PA清单存储库，亦称内容凭证。图8“C2PA清单存储库”展示了一个包含单个C2PA清单的示例存储库：

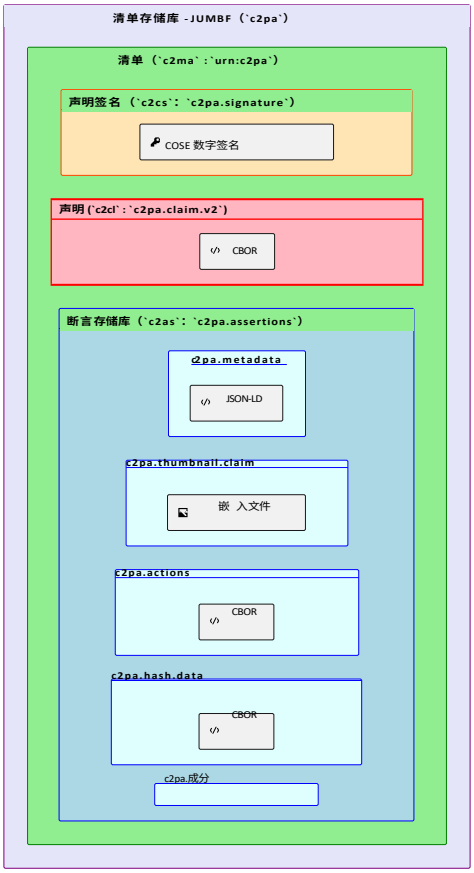

图8. C2PA清单存储库

C2PA清单存储库是一个由一系列其他JUMBF盒子和超级盒子组成的JUMBF超级盒子，每个盒子在其JUMBF描述盒中通过专属的JUMBF类型UUID和标签进行标识。C2PA清单存储库应具有标签`c2pa`、JUMBF类型UUID `63327061-0011-0010-8000-00AA00389B71`（`c2pa`），并包含一个或多个C2PA清单超级盒（亦称C2PA清单）。C2PA清单存储库还可能包含JUMBF类型UUID未在本规范中定义的JUMBF框和超级框。

注 允许其他数据框和超级数据框的存在，既能实现对C2PA的定制化扩展，也使得未来版本的规范可添加新数据框而不破坏兼容性。

每个C2PA清单应包含声明生成时创建的数据，包括C2PA断言存储库、C2PA声明及C2PA声明签名。C2PA清单还可包含JUMBF类型UUID未在本规范中定义的JUMBF框及超级框。

每个C2PA清单的JUMBF类型UUID应为`63326D61-0011-0010-8000-00AA00389B71`（`c2ma`），或 `6332756D-0011-0010-8000-00AA00389B71`（`c2um`）取决于清单类型。C2PA清单框应标注为`urn:c2pa`值标注，该值按唯一标识符部分所述方式计算。

11.1.4.3. 断言存储库

C2PA断言存储库作为超级盒子，其标签为`c2pa.assertions`，并采用JUMBF类型的UUID标识符`63326173-0011-0010-8000-00AA00389B71`（`c2as`）。其内应包含一个或多个JUMBF超框（称为

C2PA断言框）其JUMBF类型定义了包含断言数据的子框类型（ISO 19566-5:2023，附录B）。每个超级数据框均应具有标准声明中定义的标签，并包含一个JUMBF描述框、一个或多个JUMBF内容框，以及可能的填充框（ISO 19566-5:2023, A.4）。

JUMBF内容类型（ISO 19566-5:2023，附件B）框应为CBOR内容类型（`cbor`）、JSON内容类型（`json`）、嵌入文件内容类型（`bfdb` & `bidb`）或UUID内容类型（`uuid`），但允许使用JUMBF（ISO 19566-5:2023）及其修订版中定义的任何内容类型。此外，亦可使用ISO 19566-4:2020中描述的JUMBF保护框。

注：	包含其他数据格式/序列化的自定义断言（如加密数据） 通过使用包含自定义UUID后跟数据的UUID内容框实现支持（ISO 19566-5:2023, B.5）。

11.1.4.4. 声明与声明签名

C2PA声明框应具有标签`c2pa.claim.v2`，JUMBF类型UUID为`6332636C-0011-0010-8000-00AA00389B71`（`c2cl`），并包含单个CBOR内容类型框（`cbor`）。

C2PA声明签名框应具有标签`c2pa.signature`，JUMBF类型UUID为`63326373-0011-0010-8000-00AA00389B71`（`c2cs`），并包含单个CBOR内容类型框（`cbor`）。

11.1.4.5. 成分存储

当C2PA清单包含成分声明时，若某成分本身包含C2PA清单，则该清单应被完整纳入以确保溯源数据完整性。此类成分清单将作为资产本体C2PA清单的对等项添加至C2PA清单存储库。

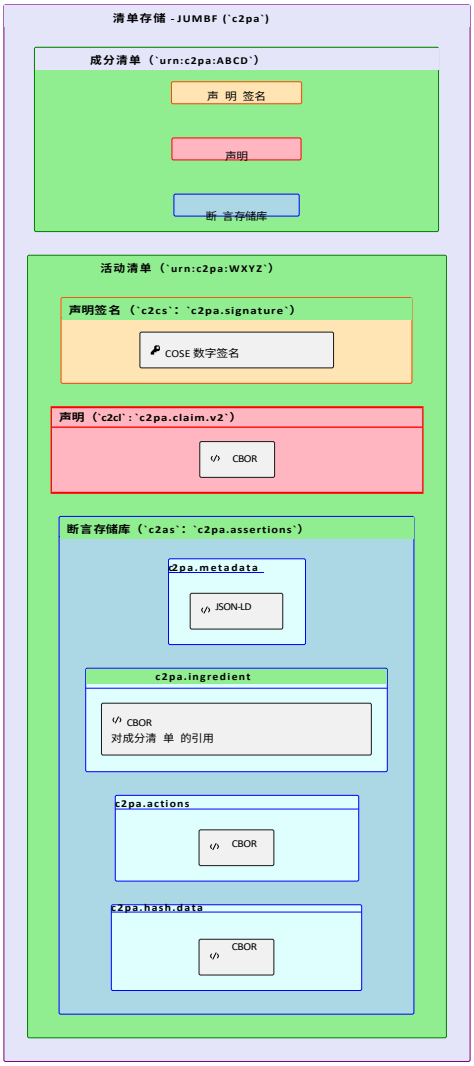

图9. 包含组件的C2PA清单存储库

11.1.4.6. 数据存储

重要

本节保留以供历史参考。数据框的概念已被

，现已弃用，转而采用标准断言方式——使用标准JUMBF嵌入文件内容类型数据框承载数据。有关嵌入数据断言的详细信息，请参阅第18.12节"嵌入数据"。

C2PA数据框存储区作为JUMBF超级框，仅可包含一个或多个CBOR内容类型框（`cbor`）。不得包含任何其他类型的JUMBF盒或超级盒。其标签应为`c2pa.databoxes`，JUMBF类型UUID为`63326462-0011-0010-8000-00AA00389B71`（`c2db`）。

CBOR内容类型盒应具有标签`c2pa.data`（用于嵌入数据）。

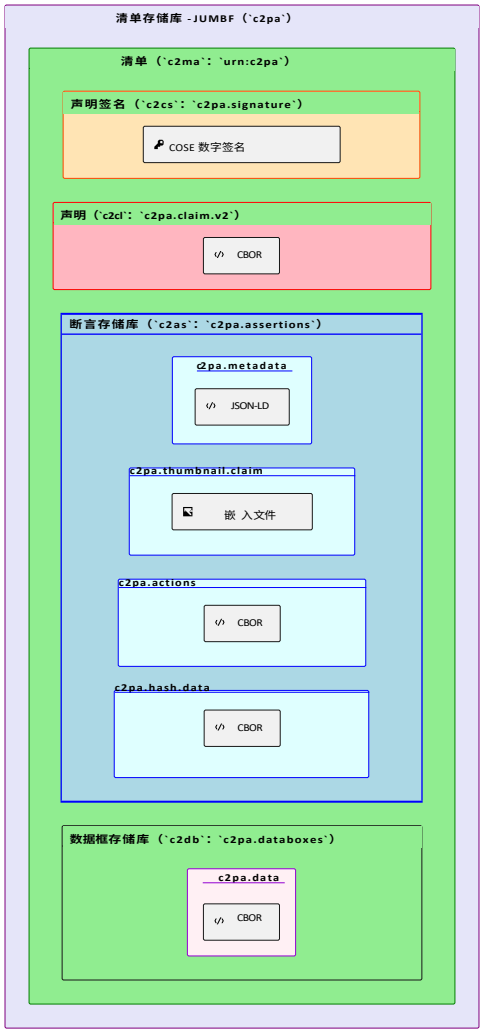

图10. 带数据框的C2PA清单存储库

11.2. 清单类型

11.2.1. 共同点

所有C2PA清单均应包含一个断言存储库，其中至少包含一个断言、一项声明及该声明的签名。

11.2.2. 标准清单

标准 C2PA 清单（JUMBF 类型 UUID：`63326D61-0011-0010-8000-00AA00389B71(c2ma)`）应包含与内容断言的精确硬绑定——根据清单所针对的资产类型和版本，可为 `c2pa.hash.data`、`c2pa.hash.boxes`、`c2pa.hash.collection.data`、`c2pa.hash.bmff.v2`（已弃用）或 `c2pa.hash.bmff.v3`。基于此要求，此类清单将成为C2PA来源数据中主要存在的清单类型。

清单消费者还应接受使用JUMBF类型UUID `63326D64-0011-0010-8000-00AA00389B71`（c2md）指定的标准C2PA清单，但声明生成器不得创建此JUMBF类型UUID的清单。

注 标准C2PA清单可作为活动清单或成分清单存在。

11.2.3. 更新清单

然而，某些来源工作流需要添加额外断言，但数字内容本身保持不变。在此类工作流中，应使用更新清单（JUMBF类型UUID：6332756D-0011-0010-8000-00AA00389B71（c2um））。

更新清单不得包含c2pa.hash.data、c2pa.hash.boxes、c2pa.hash.collection.data、c2pa.hash.bmff.v2（已弃用）或c2pa.hash.bmff.v3类型的断言，因内容未发生变更，故无需更新绑定关系。对于文件偏移量哈希（c2pa.hash.data），C2PA清单存储在更新后必须保持相同的文件起始偏移量——仅其长度可变。此外，该清单不得包含c2pa.hash.multi-asset断言。

更新清单可包含 c2pa.actions 或 c2pa.actions.v2 类型的断言，前提是这些断言中 actions 数组内每个操作的 action 字段值必须仅为以下值之一：

- c2pa.edited.metadata
- c2pa.opened
- c2pa.published
- c2pa.redacted

更新清单不得包含类型为 c2pa.actions 或 c2pa.actions.v2 的断言，且该断言不得包含此列表之外的操作字段。

更新清单可包含时间戳断言、证书状态断言或两者兼有。

注意 此为替代已弃用的时间戳清单功能的新方案。

更新清单不得包含缩略图断言。

注 这些要求的原因在于：此列表之外的操作字段或缩略图均暗示数字内容发生了变更。

更新清单应包含且仅包含一个c2pa.ingredient.v3断言，该断言需满足：(a)同时包含activeManifest和claimSignature字段，其值分别为指向C2PA清单和声明签名的URI引用（或包含c2pa.ingredient.v2或c2pa.ingredient.c2pa.ingredient字段且包含c2pa_manifest字段的c2pa.ingredient字段）；(b)关系字段的parentOf值为更新对象的父对象。v2或c2pa.ingredient，其中包含c2pa_manifest字段）的URI引用，且(b)其关系字段的parentOf值为更新对象。

注 该组件的 C2PA 清单可以是标准清单或更新清单。

11.2.4. 压缩清单

标准清单和更新清单均可采用上述Brotli压缩算法进行整体压缩。对于任一类型的清单，其`TYPE`字段值应为`c2cm`，label字段值应与压缩清单超级容器的标签完全一致，而`brob`内容框应包含整个清单超级容器的压缩字节数据。压缩标准清单示例参见图7“压缩清单示例”。

重要提示 本规范中凡提及标准清单或更新清单之处，压缩版标准清单或更新清单同样适用。

11.2.5. 时间戳清单（历史规范）

重要提示 此功能已被弃用，现已采用时间戳断言替代，请勿编写此功能。

声明生成器不应写入该字段，清单消费者也不应读取该字段。取而代之的是，应使用时间戳断言来实现相同目标。

注意 以下信息保留以供历史参考。

在某些来源追溯工作流中，标准清单或更新清单是在离线状态下创建的，此时无法在签名时从时间戳授权机构（TSA）获取符合RFC 3161标准的可信时间戳。为解决此问题，可采用时间戳清单（JUMBF类型UUID：`6332746D-0011-0010-8000-00AA00389B71 (c2tm)`），在后续操作中连接到TSA时添加时间戳。

11.3. 将清单嵌入各类文件格式

C2PA清单可嵌入多种文件格式，涵盖图像、视频、音频、字体及文档等媒体类型。附录A《*清单嵌入指南*》详细说明了如何将C2PA清单嵌入每种受支持的特定文件格式。

注 许多经典图像格式（如BMP）不支持嵌入任意数据，因此需要使用外部清单文件。

11.4. 外部清单

在某些情况下，可能无法（或不切实际）将C2PA清单存储嵌入资产中。此时，将C2PA清单存储于资产外部作为提供资产溯源的方案是可行的。C2PA清单应存储于清单存储库中，该位置需便于处理该资产的清单消费者通过引用或URI轻松定位。由于C2PA清单存储库属于JUMBF容器，其服务应采用JUMBF媒体类型`application/c2pa`。

注 本规范的早期版本曾为C2PA清单存储使用媒体类型`application/x-c2pa-manifest-store`。该媒体类型现已弃用。

使用外部清单文件的常见原因包括：

- 技术上可能不可行，例如使用`.txt`文件的情况。
- 可能不切实际，例如当C2PA清单存储库的大小超过资产的数字内容时。
- 可能存在适用性问题，例如会修改不应被修改的资产。

注 一个很好的例子就是为已存在的资源创建清单文件。

11.5. 嵌入外部清单的引用

若资产已嵌入XMP，且C2PA清单将外部存储，建议声明生成器在XMP中添加`dcterms:`provenance键，其值（URI引用）指向有效清单的存储位置。

注 本规范早期版本曾建议将此方法用于嵌入式清单引用。现该机制仅适用于外部清单。

由于字体不支持XMP，本条款针对字体部分描述了指定远程C2PA清单存储库URI的等效方法。

第12章 实体图

图11“C2PA实体图”展示了C2PA系统所有组件如何集成并相互关联。

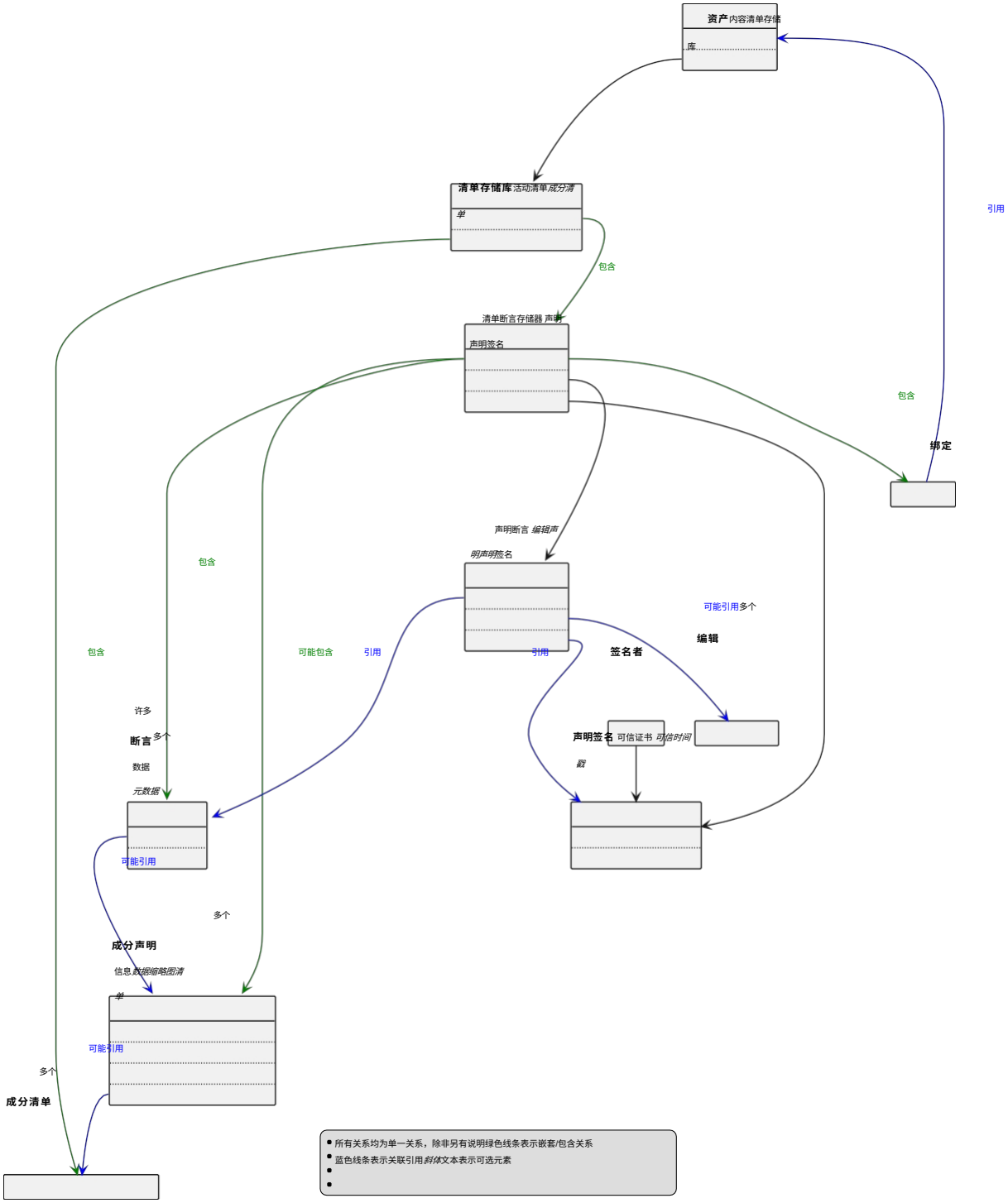

图11. C2PA实体图

第13章. 密码学

13.1. 哈希

所有依据本规范技术要求应用的密码哈希值，均须采用本节所述哈希算法生成。本节同时定义：

- 允许用于生成新内容哈希值且必须用于验证现有内容哈希值的哈希算法列表（允许列表）；
- 用于验证现有内容哈希值但禁止用于生成新内容哈希值的哈希算法列表（弃用列表）。

注 本节不适用于第18.10节“软绑定”中所述的软绑定算法。

注 本节不适用于本规范外部定义的自定义断言所使用的算法。

每种算法最多只能出现在一个列表中。针对多种输出长度实例化的算法（如不同长度的SHA2算法）应视为不同算法，每个实例化版本均需单独列出。若某算法未出现在任一列表中，则该算法被禁止使用且不得支持。为实现算法禁用，可将算法从列表中移除。因此，实现不得以可选方式支持额外算法。

实施者在发布软件更新时，应参考规范当前版本的本节内容，并确保其支持的算法符合该规范。

这些列表规定了创建哈希值的允许算法，以及作为字符串算法标识符（通常称为`alg`）在C2PA数据结构对应字段中使用的标识符。哈希函数的输出应以二进制值形式存储，通过CBOR编码为字节字符串（主要类型2），并声明其长度。凡包含哈希函数输出值的字段，其所属结构体、外层结构体、或声明映射结构体/声明映射v2结构体中，均须存在算法标识符字符串字段以声明所用算法。哈希算法标识符字段应仅存在于上述位置之一，若结构体及其外层结构体中存在多个标识符，则采用最近的标识符。"最近"的定义优先级依次为：哈希值的同级字段、其直接包含结构，直至根结构。若上述位置均未存在标识符，则应采用声明映射或声明映射v2结构中的alg字段。

允许列表为：

- SHA2-256（"sha256"）；
- SHA2-384（"sha384"）；
- SHA2-512（"sha512"）。

注 为保持与数字签名算法允许列表的一致性，SHA-3哈希算法家族未列入允许列表，因为COSE尚未建立使用SHA-3算法作为哈希算法的数字签名算法。

弃用列表为空。

13.2. 数字签名

所有依据本规范技术要求应用的数字签名，均须采用本节所述的数字签名算法及密钥类型生成。本节同时定义：

- 允许用于生成新索赔签名的数字签名算法和密钥类型列表，同时也是验证现有索赔签名所必需的（允许列表）；
- 用于验证现有声明签名但禁止用于生成新声明签名的数字签名算法及密钥类型列表（弃用列表）。

注 本节不适用于本规范外部定义的自定义断言所使用的数字签名。

这些列表通过引用相关标准中的算法标识符来确定允许使用的算法和密钥类型，这些标准定义了COSE算法及其与CBOR标识符的映射关系，包括但不限于RFC 8152和RFC 8230。这些标准还规定了签名方案中使用的哈希算法。第13.1节"哈希处理"中的规定不适用于此处哈希算法的使用；若数字签名算法及密钥类型列表中包含某数字签名算法，则签名方案中使用其指定的哈希算法应被允许并遵循。

注 下列列表中的括号注释仅为辅助说明，供读者参考。

13.2.1. 签名算法

允许列表如下：

- ES256（ECDSA配合SHA-256）；
- ES384（ECDSA配合SHA-384）；
- ES512（ECDSA配合SHA-512）；
- PS256（RSASSA-PSS 配合 SHA-256 及 MGF1 配合 SHA-256）；
- PS384（采用SHA-384的RSASSA-PSS及采用SHA-384的MGF1）；
- PS512（采用SHA-512的RSASSA-PSS及采用SHA-512的MGF1）；
- EdDSA（Edwards曲线DSA）。
    - 仅限Ed25519实例。不允许使用其他EdDSA实例。

已弃用的列表为空。

根据RFC 8152第8.1节（针对ECDSA）、第8.2节（针对EdDSA）以及RFC 8230第2节和第4节（针对RSASSA-PSS）的要求，实现方必须验证用于签名或验证操作的密钥是否适用于所选算法。

为方便起见，现将相关要求归纳如下：

- ECDSA要求使用基于P-256、P-384或P-521椭圆曲线的椭圆曲线密钥。
    - 虽然建议ES256使用P-256密钥、ES384使用P-384密钥、ES512使用P-521密钥，但并非强制要求。所有ECDSA算法选项均应接受上述任意曲线的密钥。
- Ed25519要求使用edwards25519椭圆曲线上的椭圆曲线密钥。
- RSASSA-PSS 要求 RSA 密钥的模长至少为 2048 位。

实现应拒绝使用不符合算法选择要求的密钥生成或验证签名。实现可拒绝模长超过16384位的RSA密钥。

13.2.2. COSE 的使用

针对CBOR编码声明的签名由CBOR对象签名与加密（COSE）生成，具体实现遵循RFC 8152第4.2节和第4.4节规定。

注： 有效载荷既可存在于COSE签名内部，也可单独传输（"分离式"）。

"，详见RFC 8152第4.1节）。在"分离内容"模式下，签名数据存储于COSE_Sign1_Tagged结构外部，且该结构的有效载荷字段始终为空。

无论有效负载是否包含在COSE_Sign1_Tagged签名中，当构建Sig_structure以计算或验证数字签名时，其有效负载字段的内容必须填充本规范中数字签名特定用途所描述的外部数据。Sig_structure的有效负载字段绝不能为空。

计算或验证标准清单或更新清单签名时，Sig_structure的有效负载字段将包含声明JUMBF框的内容，具体参见第10.3.2.4节"声明签名"及第11.1节"JUMBF的使用"。

13.2.3. 签名计算

签名的计算或验证应遵循RFC 8152第4.4节所述方法。Sig_structure的构造需满足以下附加要求：

- 上下文元素的值应为Signature1，除非本规范中特定数字签名应用场景要求使用CounterSignature替代。Signature不得使用。
- 有效负载元素的值将由本规范中每次使用数字签名时单独规定。

- external_aad 元素应为长度为零的 bstr。不得使用外部认证数据。
- 指定签名算法的alg头部应存在于RFC 8152第3.1节定义的body_protected元素中。

> **注** alg头部为标准COSE头部，因此始终包含在受保护的标头映射，其标签为整数1，该定义遵循IANA COSE标头参数注册表规范。字面字符串alg绝不作为标签使用。使用COSE_Sign1时，sign_protected元素始终省略。

C2PA结构中的所有数字签名均应采用RFC 8152第4.2节定义的COSE_Sign1_Tagged结构。

4.2 节定义的 COSE_Sign1_Tagged 结构。COSE_Sign1_Tagged 的构造需满足以下附加要求：

- 上述Sig_structure中的alg头部必须出现在受保护头部桶中。
- 有效负载字段的值以及有效负载是否包含在签名中或分离，将由本规范中每次使用数字签名时指定。当有效负载指定为分离时，此处的值应为空。相反，当有效负载包含在签名中时，有效负载的二进制内容将作为字符串存储在此字段中。

> **注** COSE在RFC 8152第1.3节中将nil定义为主类型7、值22，并使用此值专用于表示分离内容。长度为零的字节数组（主类型2）不能用于指示分离内容。

13.2.4. 添加声明签名时间

声明生成器也可通过添加一个值为NumericDate的iat受保护头来建立"声明签名时间"。若存在该头，则表示签名生成的时间。

> **注** NumericDate 是 CBOR 数值日期（如 RFC 8949 第 3.4.2 节所述），但省略了开头标签 1（基于纪元的日期/时间）。本规范中未在其他任何地方使用该类型。

> **注** 本建议基于JAdES的进程内更新，旨在提供一种非信任时间戳——该时间戳不用于证书有效性验证，但可用于用户体验场景。当声明生成器无法访问可信时间源，却仍需提供签名时间戳时，此机制具有实用价值。

13.2.5. 签名验证

在生成签名时，若声明生成器同时具备验证器功能，则应依据第14章 *《信任模型》* 验证签名凭证是否有效，若无效则发出警告。声明生成器仍可根据需求允许使用该凭证进行签名。当已知本地声明生成器的验证器配置与资产预期受众使用的验证器配置不同时，此操作可能具有合理性。

13.2.6. 密码学验证

在验证签名时，会生成一个内存中的Sig_structure结构体。其body_protected字段将填充来自COSE_Sign1_Tagged结构体（RFC 8152第4.4节）中受保护的头部桶的内容。对于payload字段，若签名中指定了有效负载存在，则该字段将从COSE_Sign1_Tagged结构体的payload字段中填充。若有效负载被标记为分离状态，则COSE_Sign1_Tagged结构的有效负载字段将为空。此时，Sig_structure的有效负载字段内容应从生成签名时使用的外部源填充。这些外部源在本规范中数字签名使用处均有明确定义。

13.2.7. 签名者图标的包含

C2PA清单消费者可能希望显示签名方的图标或徽标。为定位此类图形，应在嵌入式证书中查找符合RFC 9399定义的徽标。若未包含徽标，清单消费者可采用实现依赖的方式使用其他来源的图标或徽标。

第14章 信任模型

注 本节中"用户"特指在消费及创作场景中使用C2PA合规验证器的自然人操作者。

14.1. 概述

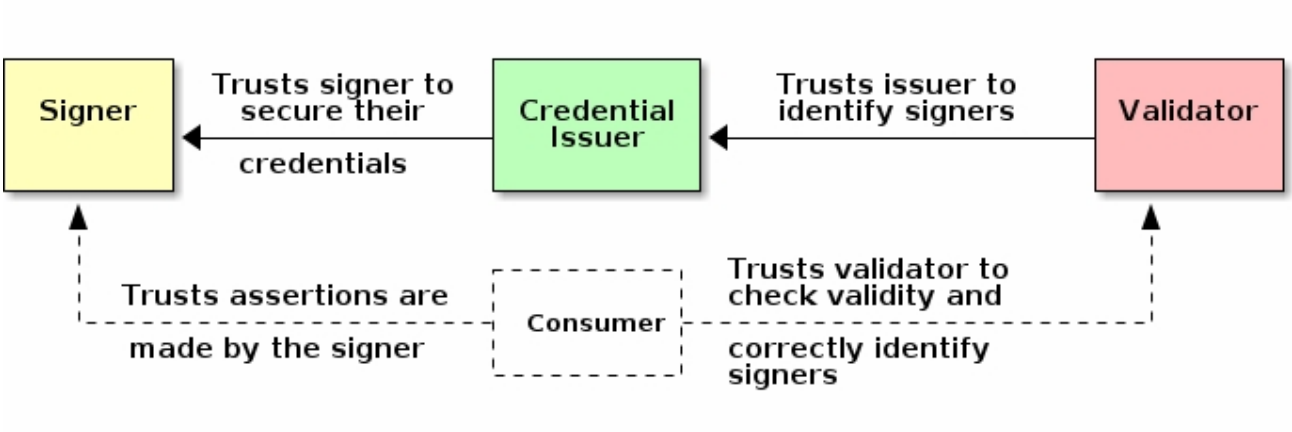

图12. C2PA信任模型示意图

图12“C2PA信任模型示意图”以黄色、绿色和红色标注了信任模型中定义的三个实体，该模型关注对签名者身份的信任。下方虚线部分为消费者（未在信任模型中定义），其通过签名者的身份及其他信任信号来判断针对资产所作声明的真实性。

14.2. 签名者的身份

信任模型中的身份是将加密签名密钥（即凭证）与签名者关联的手段，以此为基础，根据声明签名或使用该密钥签名的任何结构（包括但不限于断言和声明）来做出信任决策。

凭证应列于所有C2PA清单中用于数字签名的COSE_Sign1_Tagged结构的COSE受保护头部中。受保护头部与非受保护头部的联合中，身份凭证仅允许出现一个实例。未包含凭证或包含两个及以上凭证的COSE_Sign1_Tagged结构将被拒绝。重复使用同一凭证（包括分别出现在受保护和不受保护的标头中）也属于两个及以上凭证的情况，同样将被拒绝。

注 本规范旧版本亦允许凭证出现在COSE非保护头部中。

凭证在标头值中的存储方式、信任链的构建方式均有明确定义，更多信息可参见第 14.5 节“X.509 证书”。

14.3. 验证状态

14.3.1. 一般

验证器是一种清单消费者，它将针对该资产及其关联的有效清单生成若干验证声明。获取这些声明的过程详见验证章节。消费该资产的参与者（通常通过其用户代理及其用户界面）需解读这些声明，从而对所消费资产的来源形成自身结论。这些结论将基于声明内容及资产本身的信息得出。

14.3.2. 清单状态

基于这些声明，C2PA清单可能呈现以下状态：

- 格式正确
- 有效
- 可信

注 任何可信清单均属有效清单，任何有效清单均属格式正确清单。

14.3.3. 资产状态

若验证器报告：自活动清单生成[第15.12节"验证资产内容"]以来，受内容绑定覆盖的资产部分未被修改，且其活动清单为有效或可信状态，则该资产本身即为有效资产。

14.3.4. 格式正确的清单

若验证确认以下各项均成立，则C2PA清单为格式正确：

- 清单内容遵循本规范的规范性要求，并通过验证流程完成验证。
- 仅包含该清单类型允许的断言[第15.10.1节，“验证清单类型的正确断言”]。
- 清单中的声明满足所有声明要求[第15.10.3节，“声明验证”]。
- 清单中存在的任何成分均满足成分的所有要求[第15.11节，“验证成分”]。

14.3.5. 有效清单

若验证确认以下各项均成立，则C2PA清单视为有效：

- 清单格式正确[第14.3.4节，“格式正确的清单”]。
- 自清单签名以来，清单未被修改[第13.2.6节，“加密验证”]。
- 声明签名收到成功代码 `claimSignature.validated` [第15.7节，“验证签名”]。
- 验证的的声明签名有效性期限接收签名成功代码的 `claimSignature.insideValidity` [第15.8节“验证时间戳”]。
- C2PA清单签名者的凭证未因签名凭证失效代码`signingCredential.ocsp.revoked`或`signingCredential.ocsp.unknown`而被拒绝[第15.9节，“验证凭证吊销信息”]。

如果 C2PA 清单有效，则该清单的声明可归因于声明生成器，该生成器通过声明的
声明生成器信息字段[第10.2.3节“声明生成器信息”]标识。

14.3.6. 可信清单

当验证确认以下所有条件均成立时，C2PA清单即为可信：

- 清单有效 [第 14.3.5 节“有效清单”]。
- C2PA清单的签名凭证收到`signingCredential.trusted`的成功代码
  [第15.7节，“验证签名”]。

14.4. 信任列表

14.4.1. C2PA签名者

验证者应维护以下信息以评估C2PA签名者：

- 已接受的扩展密钥用途（EKU）值列表。
- 针对每个接受的EKU值，提供X.509证书信任锚列表。

对于 `c2pa-kp-claimSigning` (1.3.6.1.4.1.62558.2.1) EKU，信任锚列表应包含但不限于 C2PA 提供的签名者信任锚（即 C2PA 信任列表）。

注 其中某些列表可以为空。

除 C2PA 信任列表中为 `c2pa-kp-claimSigning` EKU 提供的信任锚列表外，验证器应允许用户为该 EKU 及/或其他 EKU （如 `id-kp-emailProtection` (1.3.6.1.5.5.7.3.36)或id-kp-documentSigning (1.3.6.1.5.5.7.3.36)）配置额外信任锚。5.7.3.4）或 `id-kp-documentSigning`（1.3.6.1.5.5.7.3.36））。验证器应提供默认选项或由外部方维护的列表，用户可选择加入以填充验证器的 C2PA 签名者信任锚存储库。

注 本规范的先前版本要求证书中必须包含 `id-kp-emailProtection` 或 `id-kp-claimSigning` 扩展，因此在签名者证书中加入至少其中一种扩展（配合 `c2pa-kp-claimSigning` 使用），可提升与旧版验证器的兼容性。

> `kp-documentSigning` EKU，因此在签名者证书中包含上述任一EKU并 配合`c2pa-kp-claim`Signing，可提升与旧版验证器的兼容性。

14.4.2. 时间戳机构

验证者应维护时间戳签发机构（TSA）的X.509证书信任锚列表，该列表应与C2PA签名者的列表分开。此列表应包括但不限于C2PA提供的TSA信任锚（即C2PA TSA信任列表）。

注	该列表可为空。

除C2PA TSA信任列表中提供的信任锚列表外，验证器应允许用户配置额外的TSA信任锚存储库，并应提供默认选项或由外部方维护的可供用户选择加入的列表，以便为验证器的可信时间戳机构信任锚存储库填充数据。

14.4.3. 私有凭证存储

验证器还可允许用户创建并维护签名凭证的私有凭证存储库。该存储库旨在作为用户基于带外关系选择信任的凭证"地址簿"。若存在私有凭证存储库，其仅适用于验证签名的C2PA清单，不得用于验证时间戳。该存储库仅允许直接信任签名者凭证；存储库条目不得用于签发凭证，且在验证过程中不得作为信任锚点。

验证器不得预先配置任何私有凭证存储库条目。

验证器仅应响应用户信任凭证的请求向私有凭证存储库添加条目。同理，验证器仅应响应用户停止信任凭证的请求从私有凭证存储库移除条目。

14.5. X.509证书

X.509证书的存储方式遵循RFC 9360（*CBOR对象签名与加密（COSE）：承载和引用X.509证书的头部参数*）的定义。为方便起见，`x5chain`的定义如下所示。

重要	本规范在RFC 9360要求基础上增加了额外规范，具体内容见引文后补充条款。特别要求所有中间证书签名者证书链中的所有中间证书机构证书必须包含在 `x5chain` 头部中，并要求声明生成器始终将 `x5chain` 头部放置在受保护的头部桶中。

> x5chain：该标头参数包含有序排列的X.509证书数组。证书排序应以包含终端实体密钥的证书为首，随后是签发该证书的证书，依此类推。无需包含整个

若存在合理依据表明依赖方已持有或可定位缺失证书，则该元素中必须包含链。这意味着依赖方仍需执行路径构建，但本头部参数中已提出候选路径。

信任机制必须将此参数中的任何证书视为不可信输入。参数中存在自签名证书时，在未获得带外确认的情况下，不得触发信任锚集的更新。由于该标头参数内容属于不可信输入，其可置于受保护或非受保护标头桶中。若通过非受保护标头桶发送该参数，中间节点可移除或添加证书。

终端实体证书必须通过COSE进行完整性保护。具体实现方式包括：在受保护头部中发送标头参数；在未受保护头部中发送'x5chain'参数，同时在受保护头部中发送'x5t'参数；或将终端实体证书包含在external_aad中。

该头部参数支持在消息中承载单个X.509证书或X.509证书链。

- 若传输单个证书，则将其封装在CBOR字节字符串中。
- 若传输多个证书，则使用字节串的CBOR数组，每个证书对应独立字节串。

验证器仅需持有其信任锚点的证书。因此，在签名过程中创建x5chain头部时，声明生成器应将签名者的证书及所有中间证书颁发机构纳入头部值中。信任锚点的证书（亦称根证书）不应包含在内。

首张或唯一证书的`subjectPublic`KeyInfo元素将作为验证签名的公钥。tbsCertificate序列中的`validity`元素提供证书的时间有效期。

x5规范先前版本要求声明生成器仅写入字符串标签 x5chain，以规避整数标签 `33` 未被标准化的极低概率风险。

整数标签`33`现已标准化，本规范将其作为标准采用，并弃用字符串标签。因此：

- 声明生成器在将此标头插入COSE签名时，应仅使用整数`33`作为标签。声明生成器仍可继续写入字符串标签`x5chain`，但此行为现已弃用，声明

生成器应更新为仅使用整数标签。声明生成器必须将此标头置于COSE签名的受保护标头区块中，如上所述。

- 验证者应接受字符串 `x5chain` 或整数 `33` 作为此标头的标签。若同时存在两种标签，验证者应使用整数标签 `33` 的标头，并忽略字符串 `x5chain` 标签的标头。为保持与本规范旧版本的兼容性，验证者应接受来自受保护桶或非受保护桶的标头。根据第14.2节"签名者身份"规定，若该标头同时出现在受保护和未受保护桶中且标签相同，验证者应因存在多重凭证而拒绝该声明签名，判定其格式错误。

14.5.1. 证书配置文件

14.5.1.1. 通用要求

本节规定了验证X.509证书是否可作为签名凭证的要求，具体参见第15.7节“验证签名”。

所有证书均须满足以下要求。

- 签名算法字段的算法字段应为以下值之一：

  ecdsa-with-SHA256

  RFC 5758 第3.2节

  ecdsa-with-SHA384

  RFC 5758 第3.2节

  ecdsa-with-SHA512

  RFC 5758, 第3.2节

  sha256WithRSAEncryption

  RFC 8017，附录A.2.4

  sha384WithRSAEncryption

  RFC 8017，附录A.2.4

  sha512WithRSAEncryption

  RFC 8017，附录A.2.4

  id-RSASSA-PSS

  RFC 8017，附录A.2.3

  id-Ed25519

  RFC 8410 第3节

- 若签名算法字段（signatureAlgorithm）的算法字段为 id-RSASSA-PSS，则参数字段（parameters）应为 RSASSA-PSS-params 类型。其字段须满足 RFC 8017 附录 A.2.3 中定义的以下要求：
  - 必须包含hashAlgorithm字段。
  - hashAlgorithm字段的algorithm字段应为RFC 8017附录B.1中定义的下列值之一：
    - id-sha256。
    - id-sha384。
    - id-sha512。
  - maskGenAlgorithm 字段必须存在。
  - maskGenAlgorithm 字段的 parameters 字段中的 algorithm 字段应等于哈希算法字段中的算法字段。
- 如果证书的subjectPublicKeyInfo字段中的算法字段为id-ecPublicKey，则参数字段应为RFC 5480第2.1.1.1节中下列命名的曲线之一：
  - prime256v1.
  - secp384r1.
  - secp521r1.
- 若证书subjectPublicKeyInfo的算法字段为rsaEncryption或id-RSASSA-PSS，则参数字段的模数字段长度应至少为2048位。

除X.509证书私有凭证存储库中的证书外，所有证书均须满足以下附加要求方可被接受：

- 版本应为 RFC 5280 第 4.1.2.1 节规定的 v3。
- 根据RFC 5280第4.1.2.8节规定，TBSCertificate序列中的issuerUniqueID和subjectUniqueID可选字段不得存在。
- 基本约束扩展应遵循RFC 5280第4.2.1.9节规定。具体而言，下列条件之一必须成立：
  - 若证书公钥可用于验证证书签名，则基本约束扩展必须存在且cA布尔值为真。
  - 若认证公钥不可用于验证证书签名，则基本约束扩展必须缺失，或该扩展中的cA布尔值不得设置，且密钥使用扩展中的keyCertSign位不得设置。
- 根据RFC 5280第4.2.1.1节规定，任何非自签名证书均应包含授权密钥标识符扩展。

- 根据RFC 5280第4.2.1.2节规定，任何充当CA的证书均应包含主题密钥标识符扩展。该扩展也应存在于终端实体证书中。
- 根据RFC 5280第4.2.1.3节规定，密钥用途扩展必须存在且应标记为关键属性。用于签署C2PA清单的证书应设置数字签名位。仅当基本约束扩展中的cA布尔值被设置时，才应设置密钥证书签名位。
- 根据RFC 5280第4.2.1.12节规定，当证书缺少基本约束扩展或cA布尔值未被声明时，扩展密钥用途（EKU）扩展必须存在且非空。此类证书通常称为"终端实体"或"叶"证书。
  - 不得包含任何扩展密钥用途（anyExtendedKeyUsage，2.5.29.37.0）扩展。
  - 用于签署时间戳副署的证书必须适用于 id-kp-timeStamping (1.3.6.1.5.5.7.3.8) 用途。
  - 用于为证书签名 OCSP 响应的证书应适用于 id-kp-OCSPSigning (1.3.6.1.5.5.7.3.9) 用途。
  - 若证书对 id-kp-timeStamping 或 id-kp-OCSPSigning 有效，则仅对其中一个用途有效，且对其他用途无效。
  - 证书不应在上述用途之外用于任何其他目的，但本配置文件未提及且配置存储库中的EKU列表中未包含的任何EKU的存在，均不应导致证书被拒绝。

14.5.1.2. 证书信任链

当验证证书作为签名凭证时，若该证书存在于私有凭证存储库中的
验证时间戳时不参考私有凭证存储库。

如果证书不在私有凭证存储库中，或验证器未实现该存储库，则应根据RFC 5280第6节所述流程构建并验证信任链，该流程需针对特定用途（签名、时间戳或OCSP签名）及该用途对应的信任锚存储库进行适配。若验证算法出现任何失败，则该链应被拒绝。构建证书链时绝不包含私有凭证存储库；私有凭证存储库中的证书不能充当CA。

仅允许使用终端实体证书对C2PA声明或时间戳进行签名。证书颁发机构（CA）证书不得用于此类目的。若将任何CA证书（其基本约束扩展中的cA布尔值被声明为有效）用于验证C2PA声明、时间戳或OCSP响应上的签名，则应以signingCredential.untrusted失败代码予以拒绝。

验证者应确保签名证书获得其使用目的的授权，并拒绝用于未经授权目的的证书。若证书的扩展密钥用途扩展中包含该用途的扩展密钥用途对象标识符（OID）（参见RFC 5280第4.2.1.12节），则该证书即被视为获得特定用途的授权。

在验证用于签署C2PA声明的证书时，签名证书应至少包含一个有效验证者关联信任锚列表的EKU（参见第14.4.1节“C2PA签名者”），且验证者应

仅使用与证书中EKU关联的信任锚点。

验证用于签署时间戳的证书链时，签名证书必须包含`id-kp-timeStamping`（1.3.6.1.5.5.7.3.8）EKU。

验证用于签署OCSP响应的证书链时，签名证书应具有`id-kp-OCSPSigning`（1.3.6.1.5.5.7.3.9）扩展键值。

除通过私有凭证存储库接受的X.509证书外，验证器应验证证书是否符合证书配置文件要求，并拒绝不符合要求的证书。这包括要求证书必须包含扩展密钥用途扩展，且证书仅被授权用于本节列出的三种用途之一：C2PA签名、时间戳签名或OCSP响应签名。

如证书配置文件所述，签发证书的认证机构（CA）证书无需包含EKU扩展，通常也不会包含。若存在该扩展，则应予以忽略。此要求仅适用于为C2PA清单、时间戳或OCSP响应签名的终端实体证书。CA证书不得用于为C2PA清单、时间戳或OCSP响应签名。

14.5.2. 证书吊销

X.509证书支持吊销状态查询。声明生成器应采用在线证书状态协议（OCSP，RFC 6960）及OCSP钉接机制（概念源自RFC 6066第8节，但本条款描述为其实现方式）来实现吊销功能。声明生成器不得使用证书吊销列表（CRL，RFC 5280）。

> **注：** 使用证书撤销列表（CRL）需要为每个证书颁发机构下载完整的已撤销证书列表
> 这可能耗费大量时间。虽然CRL可采用与OCSP响应相同的方式进行附加，但CRL相对于OCSP响应的潜在体积也使得这种做法不可取。

符合规范的CA应在其签发的证书中包含一个AuthorityInfoAccess（AIA）扩展（RFC 5280第4.2.2.1节），以提供访问该CA运营的OCSP服务的访问信息。

若证书包含AIA扩展，撤销信息应存储于COSE_Sign1结构的未保护头部，采用字符串标签`rVals`，其值的模式应遵循示例3"`rVals`的`CDDL`"中的rVals规则：

示例3. `rVals`的CDDL

```
; 基于https://www.etsi.org/deliver/etsi_ts/119100_119199/11918201/01.01.01_60/ts_11918201v010101p.pdf第5.3.5.2节JSON模式的rVals及相关结构CBOR版本
rVals = {
   "ocspVals": [1* bstr]
}
```

注释	上述定义是对JAdES第5.3.5.2节中模式子集的CBOR适配，仅存储OCSP响应，且以二进制字符串形式存储。

在签名声明前，若签名者证书包含AIA扩展，声明生成器应查询该扩展中指定的OCSP服务，捕获响应并将其存储于rVals头部的`ocsp`Vals数组元素中。声明生成器应对声明签名中包含的任何中间CA证书执行相同操作。

接收声明后，应依据RFC 6960第3.2节验证附加的OCSP响应。

关于声明签名后验证证书吊销状态的流程，详见《验证凭证吊销信息》章节。

第15章 验证

15.1. 验证流程

15.1.1. 描述

C2PA清单的验证是一个多步骤过程，涉及验证清单中包含的断言、声明及相关声明签名，同时（仅针对活动清单）验证任何关联的硬绑定。该验证过程由验证器执行，该验证器是实现本条款所述验证算法的硬件或软件实体。

15.1.2. 验证阶段

以下各条款按无特定顺序列举了这些阶段：

- 第15.10节“断言验证”：断言验证流程。
- 第15.11节，“验证成分”：验证成分（如有）。
- 第15.8节，“验证时间戳”：验证时间戳。
- 第15.9节，“验证凭证撤销信息”：验证凭证撤销信息。
- 第15.7节，“验证签名”：验证声明签名。
- 第15.12节，“验证资产内容”：验证资产内容。

如第14.3节"验证状态"所述，基于上述步骤的结果，C2PA清单可被判定为"格式正确"、"有效"或"可信"。

15.1.3. 可视化呈现

图13“验证声明”直观展示了验证C2PA清单的过程。

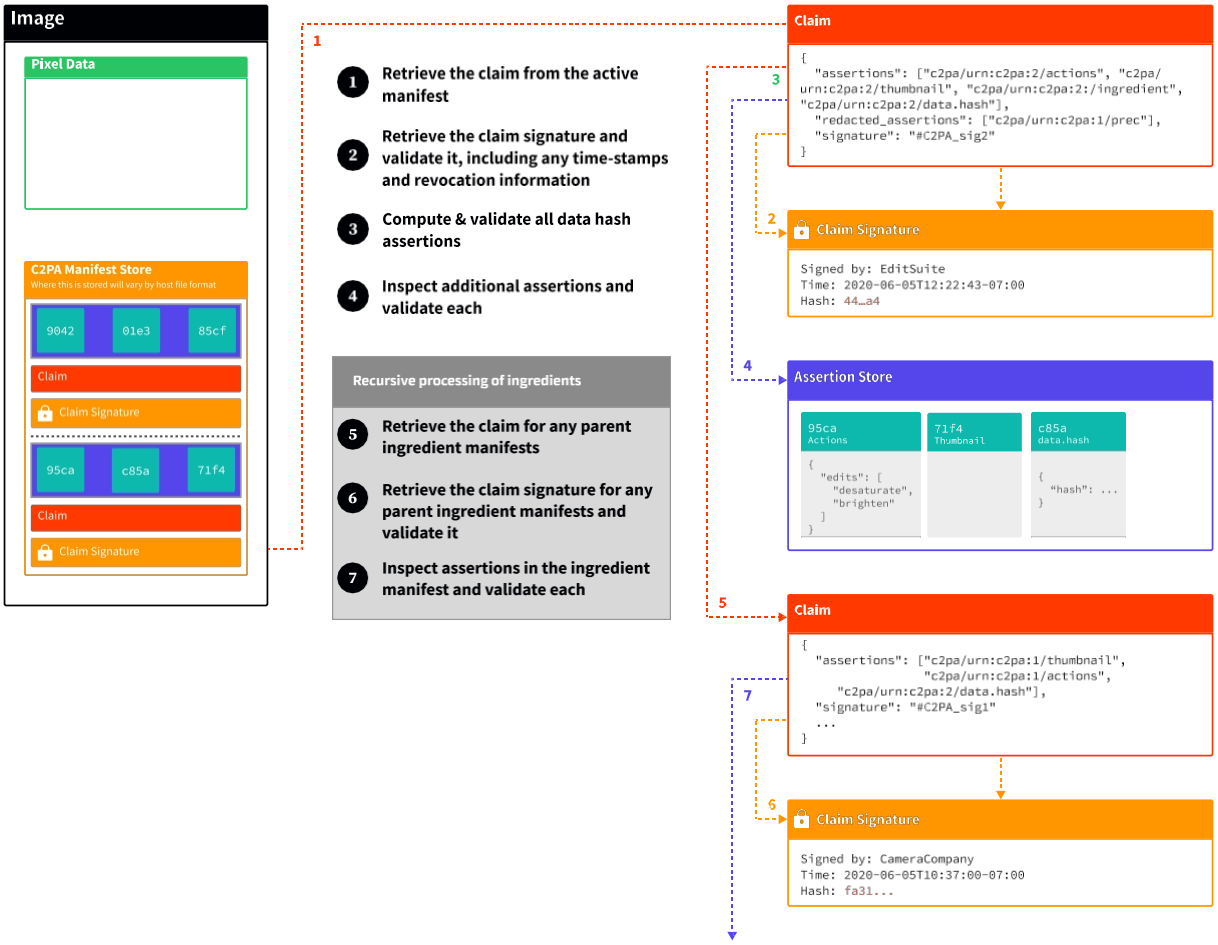

图13. 验证声明

注	若视觉呈现与文本存在任何差异，以文本为准。

15.2. 返回验证结果

15.2.1. 通用

验证算法应为资产C2PA清单存储库中的所有清单返回综合验证结果集，包括当前活动清单以及通过成分断言引用的存储库内所有其他清单。

验证结果通过一组标准状态代码表示，包括成功、信息和失败代码，具体定义详见第15.2.2节“标准状态代码”。当声明生成器需要记录特定流程状态信息时，允许使用自定义状态代码。自定义代码应遵循实体特定命名空间的相同语法规则，例如：`com.litware`.

当声明生成器通过成分声明添加成分资产时，应作为验证器，并对该成分执行本节所述的完整验证算法。声明生成器应记录

验证结果 来自 该 成分， 根据 该 规范 CDDL 定义 模式， 作为 该 值 该 成分的成分断言中的validationResults字段。

CDDL 验证结果规范

```
; 验证代码

; 成功代码
$status-code /= "assertion.accessible"
$status-code /= "assertion.bmffHash.match"
$status-code /= "assertion.boxesHash.match"
$status-code /= "断言.集合哈希匹配"
$status-code /= "assertion.dataHash.match"
$status-code /= "断言.哈希URI匹配"
$status-code /= "声明签名有效性范围"
$status-code /= "声明签名.验证通过"
$status-code /= "ingredient.claimSignature.validated"
$status-code /= "组件.清单.验证通过"
$status-code /= "签名凭证.OCSP未撤销"
$status-code /= "签名凭证受信任"
$status-code /= "时间戳受信任"
$status-code /= "时间戳.验证通过"

; 信息性代码
$status-code /= "ingredient.unknownProvenance"
$status-code /= "清单来源未知"
$status-code /= "签名凭证.OCSP不可访问"
$status-code /= "签名凭证.OCSP.跳过"
$status-code /= "签名时间在有效期内"
$status-code /= "签名时间超出有效期"
$status-code /= "时间戳格式错误"
$status-code /= "时间戳不匹配"
$status-code /= "时间戳超出有效期"
$status-code /= "时间戳不可信"
$status-code /= "断言.数据哈希.存在额外排除项"

; 失败代码
$status-code /= "算法已弃用"
$status-code /= "algorithm.unsupported"
$status-code /= "断言.操作.成分不匹配"
$status-code /= "断言操作格式错误"
$status-code /= "断言操作：内容被编辑"
$status-code /= "断言操作编辑不匹配"
$status-code /= "断言.bmffHash.格式错误"
$status-code /= "assertion.bmffHash.不匹配"
$status-code /= "断言.boxesHash.不匹配"
$status-code /= "assertion.boxesHash.格式错误"
$status-code /= "断言.盒子哈希.未知盒子"
$status-code /= "断言.云数据.硬绑定"
$status-code /= "断言.云数据.操作"
$status-code /= "断言.云数据.硬绑定"
$status-code /= "断言.云数据.格式错误"
$status-code /= "断言.集合哈希.文件数量错误"
```

```
$status-code /= "assertion.collectionHash.无效URI"
$status-code /= "assertion.collectionHash.格式错误"
$status-code /= "断言.集合哈希.不匹配"
$status-code /= "断言.数据哈希.格式错误"
$status-code /= "断言.数据哈希.不匹配"
$status-code /= "断言.哈希URI不匹配"
$status-code /= "断言.不可访问"
$status-code /= "断言.成分.格式错误"
```

```
$status-code /= "断言.json.无效"

$status-code /= "断言.缺失"
$status-code /= "assertion.multipleHardBindings"
$status-code /= "assertion.notRedacted"
$status-code /= "assertion.outsideManifest"
$status-code /= "assertion.selfRedacted"

$status-code /= "assertion.未声明"

$status-code /= "声明.cbor.无效"

$status-code /= "声明.硬绑定缺失"

$status-code /= "声明格式错误"

$status-code /= "声明缺失"

$status-code /= "声明项重复"

$status-code /= "声明签名缺失"

$status-code /= "声明签名不匹配"

$status-code /= "声明签名超出有效期"

$status-code /= "general.error" ; 当其他情况均不适用时

$status-code /= "哈希URI缺失"

$status-code /= "哈希URI不匹配"

$status-code /= "组件.声明签名.缺失"

$status-code /= "组件.声明签名.不匹配"

$status-code /= "成分清单缺失"

$status-code /= "成分清单不匹配"

$status-code /= "清单压缩无效"

$status-code /= "清单不可访问"

$status-code /= "清单文件存在多个父项"

$status-code /= "清单时间戳无效"
$status-code /= "manifest.timestamp.wrongParents"
$status-code /= "manifest.update.invalid"

$status-code /= "清单更新错误父节点"

$status-code /= "签名凭证无效"

$status-code /= "签名凭证OCSP已撤销"

$status-code /= "签名凭证OCSP未知"

$status-code /= "签名凭证不可信"


; 自定义状态码
$status-code /= tstr .regexp "([\\da-zA-Z_-]+\\.)+[\\da-zA-Z_-]+"

status-map = {
   "code": $status-code,                    ; 描述状态的标签格式字符串
   ? "url": jumbf-uri-type,                 ; 指向适用此状态码的JUMBF盒子的JUMBF URI引用
   ? "explanation": tstr .size (1..max-tstr-length), ; 可读性状态说明字符串
   ? "success": bool                        ; 已弃用。代码是否反映成功（true）或失败（false）
}

状态码映射 = {
     "成功": [* $status-map],                          ; 验证成功代码数组。可能为空。
     "informational": [* $status-map],                 ; 包含验证信息性代码的数组。可能为空。
     "failure": [* $status-map]                        ; 验证失败代码数组。可能为空。
}

; 包含清单及其成分验证结果的对象 validation-results-map = {
     ? "activeManifest": $status-codes-map,; 组件有效清单的验证状态代码。若组件为C2PA资产则存在，否则不存在。
     ? "ingredientDeltas": [* $ingredient-delta-validation-result-map] ; 列出每个成分清单当前与先前验证结果之间的任何
```

```
当前与先前验证结果之间的变更/增量列表。若该成分为C2PA资产则存在。
}

ingredient-delta-validation-result-map = {
      "ingredientAssertionURI": jumbf-uri-type, ; 指向成分断言的JUMBF URI引用
      "validationDeltas": $status-codes-map                    ; 该成分当前有效清单的验证结果
}
```

15.2.2. 标准状态码

15.2.2.1. 成功代码

表 2. 验证成功代码

值	含义	url 使用
assertion.accessible	非嵌入式（远程）断言在验证时可访问。	C2PA 断言
assertion.bmffHash.match	基于盒装资产的哈希值与 BMFF 哈希断言中声明的哈希值匹配。	C2PA 断言
assertion.boxesHash.match	基于盒子的资产哈希值与通用盒子哈希声明中的哈希值匹配。	C2PA 断言
assertion.collectionHash.match	集合中所有资产的哈希值与集合数据哈希声明中的哈希值匹配。	C2PA 断言
assertion.dataHash.match	资产字节范围的哈希值与数据哈希断言中声明的哈希值匹配。	C2PA 断言
assertion.hashedURI.match	引用的断言的哈希值与声明中断言的哈希URI对应的哈希值匹配。	C2PA 断言
assertion.multiAssetHash.match	多资产哈希断言中某部分的哈希值与该断言的多资产哈希映射中对应的哈希值相匹配。	C2PA 断言
声明签名有效期内	声明中引用的声明签名是在签名凭证有效期内创建的	C2PA声明签名框
claimSignature.validated	声明中引用的声明签名已通过验证	C2PA 声明签名框

成分.声明签名.验证	该成分的C2PA声明签名框的哈希值已成功验证。	C2PA 断言

值	含义	url用法
ingredient.manifest.validated	该成分的C2PA清单框哈希值已成功验证。	C2PA 断言
签名凭证的OCSP未撤销状态正常	签名凭证在签名时未被撤销。	C2PA声明签名框
签名凭证.受信任	签名凭证受信任	C2PA声明签名框
时间戳.受信任	时间戳凭证被列入验证器的可信锚列表中，该列表用于时间戳机构。	C2PA声明签名框
timeStamp.validated	该时间戳格式正确，其消息印记与声明签名匹配，且在时间戳凭证有效期内生成。	C2PA声明签名框

15.2.2.2. 信息代码

表 3. 验证信息代码

值	含义	url 使用
algorithm.deprecated	该算法已被弃用。	C2PA声明框或C2PA断言
成分来源未知	该成分没有 C2PA 清单。	C2PA 断言
签名凭证.OCSP不可访问	验证器尝试执行在线OCSP检查，但未收到响应。	C2PA声明签名框
签名凭证OCSP被跳过	验证器选择不执行在线 OCSP 检查。	C2PA 声明签名框
签名时间在有效期内	签名声明的时间（位于签名的iat头部）处于签名者证书链的有效期内，且早于任何对应可信时间戳的时间。	C2PA声明签名框
timeOfSigning.outsideValidity	声明的签名时间（在签名的iat头中）超出签名者证书链的有效期，或晚于任何对应可信时间戳的时间。	C2PA 声明签名框

timeStamp.格式错误	声明签名头中包含的时间戳响应格式不符合 RFC 3161 规范	C2PA 声明签名框

值	含义	url用法
timeStamp.mismatch	时间戳与声明内容不匹配。	C2PA 声明签名框
timeStamp.outsideValidity	签名中的签名时间戳属性是在 TSA 证书有效期之外创建的。	C2PA 声明签名框
timeStamp.untrusted	该时间戳凭证未列入验证器的TSA信任列表。	C2PA声明签名框

15.2.2.3. 失败代码

表 4. 验证失败代码

值	含义	url 使用
algorithm.unsupported	算法未指定或不受支持。	C2PA声明框或C2PA断言
assertion.action.ingredientMatch	需要关联成分的操作要么没有指定成分，要么指定的成分无法定位	C2PA 断言
assertion.action.malformed	操作断言格式错误。	C2PA 断言
assertion.action.redacted	在声明创建时，操作断言被编辑。	C2PA 断言
assertion.action.redactionMis 匹配	需要关联编辑的操作要么未关联编辑，要么指定的编辑无法定位	C2PA 断言
assertion.bmffHash.格式错误	BMFF哈希断言格式错误。	C2PA 断言
assertion.bmffHash.mismatch	基于箱式资产的哈希值与BMFF哈希声明中的哈希值不匹配。	C2PA 断言
assertion.boxesHash.malformed	通用盒子哈希声明格式错误。	C2PA 断言
assertion.boxesHash.mismatch	通用盒状资产格式的哈希值与通用盒哈希断言中声明的哈希值不匹配。	C2PA 断言
assertion.boxesHash.unknownBox	发现了一个非预期的盒子	C2PA 断言

assertion.cloud-data.actions	更新清单包含一个云数据断言，该断言引用了一个操作断言。	C2PA 断言
assertion.cloud-data.hardBinding	强绑定声明位于云数据声明中。	C2PA 断言
值	**含义**	**URL 使用**
云数据断言格式错误	云端数据断言不完整	C2PA 声明
assertion.cbor.invalid	断言的cbor无效	C2PA 断言
assertion.collectionHash.incorrectFileCount	集合数据哈希断言中列出的资产在集合中缺失。	C2PA 断言
assertion.collectionHash.invalidURI	集合数据哈希断言中某项资产的 URI 包含文件部分 '..' 或 '.'。	C2PA 断言
assertion.collectionHash.malformed	集合哈希断言不完整	C2PA 断言
assertion.collectionHash.mismatch	集合中某项资产的哈希值与集合数据哈希断言中声明的哈希值不匹配。	C2PA 断言
assertion.dataHash.格式错误	数据哈希断言格式错误。	C2PA 断言
assertion.dataHash.mismatch	该资产字节范围的哈希值与数据哈希断言中声明的哈希值不匹配。	C2PA 断言
assertion.dataHash.redacted	声明创建时，某个硬性绑定断言被编辑处理。	C2PA 断言
assertion.hashedURI.mismatch	清单中引用的断言哈希值与声明中该断言哈希URI对应的哈希值不匹配。	C2PA 断言
assertion.inaccessible	验证时无法访问非嵌入式（远程）断言。	C2PA 断言
assertion.ingredient.malformed	成分声明不完整	C2PA 声明
assertion.json.invalid	声明的 JSON(-LD) 无效	C2PA 声明
assertion.missing	清单声明中列出的断言在资产清单中缺失。	C2PA 声明框
assertion.multiAssetHash.malformed	多资产哈希断言格式错误。	C2PA 断言
assertion.multiAssetHash.missingPart	多部分资产的必备组件无法定位。	C2PA 断言

assertion.multiAssetHash.mismatch	多部分资产中某部分的哈希值与多资产哈希断言中声明的哈希值不匹配。	C2PA 断言
assertion.multipleHardBindings	清单中包含多个硬性绑定断言。	C2PA 断言存储框

值	含义	url 使用
assertion.notRedacted	该声明在声明中被标记为已编辑，但在清单中仍存在。	C2PA 断言
assertion.outsideManifest	声明中列出的断言未与该声明位于同一 C2PA 清单中	C2PA声明框
assertion.selfRedacted	该断言被其所属声明标记为已编辑。	C2PA 声明框
assertion.timestamp.malformed	时间戳断言格式错误。	C2PA 断言
assertion.undeclared	在清单中发现一项未在声明中明确声明的断言。	C2PA 断言
claim.cbor.invalid	该声明的 cbor 无效。	C2PA 声明框
claim.hardBindings.missing	未检测到硬绑定。	C2PA 声明框
声明格式错误	清单中引用的声明的数据/字段不正确。	C2PA 声明框
声明.缺失	清单中引用的声明无法找到。	C2PA 声明框
声明.多重	清单中存在多个索赔框。	C2PA 声明框
声明签名缺失	声明中引用的声明签名在其清单中无法找到。	C2PA 声明签名框
声明签名不匹配	声明中引用的声明签名验证失败。	C2PA 声明签名框
claimSignature.outsideValidit y	声明中引用的声明签名是在签名凭证有效期之外创建的。	C2PA 声明签名框
general.error	当发生未在此处具体列出的错误时使用的值。	C2PA声明框或C2PA断言
hashedURI.missing	无法定位由hashed_uri指向的数据	C2PA 声明

hashedURI.mismatch	给定哈希URI的哈希值与目标URI数据对应的哈希值不匹配	C2PA 断言
ingredient.claimSignature.mis sing	未找到引用的成分 C2PA 声明签名。	C2PA 断言
值	**含义**	**url 使用**
成分.声明签名.不匹配	嵌入式C2PA清单的C2PA声明签名哈希值与成分声明中 claimSignature字段的hashed_uri值声明的哈希值不匹配。	C2PA 声明
ingredient.manifest.missing	未找到引用的组件 C2PA 清单。	C2PA 声明
ingredient.manifest.mismatch	嵌入式 C2PA 清单的哈希值与成分断言中 activeManifest 字段的 hashed_uri 值声明的哈希值不匹配。	C2PA 声明
manifest.compressed.invalid	压缩后的清单文件无效。	C2PA 声明框
manifest.inaccessible	验证时无法访问非嵌入式（远程）清单文件。	C2PA 声明框
manifest.multipleParents	清单中包含多个成分，它们之间的关系为父级关系。	C2PA声明框
manifest.timestamp.invalid	该清单为时间戳清单，但包含了不允许的（非成分）断言。	C2PA 声明框
manifest.timestamp.wrongParen ts	清单是时间戳清单，但包含零个或多个 parentOf 成分。	C2PA声明框
manifest.update.invalid	清单是更新清单，但包含不允许的断言，例如硬绑定或操作断言。	C2PA 声明框
manifest.update.wrongParents	该清单是更新清单，但包含零个或多个 parentOf 元素。	C2PA 声明框
签名凭证无效	签名凭证无效，无法用于签名。	C2PA 声明签名框

签名凭证OCSP撤销状态	OCSP响应表明签名凭证已被签发者撤销。	C2PA 声明签名框
签名凭证.ocsp.unknown	OCSP响应包含未知 状态	C2PA 声明签名框
签名凭证.未受信任	该签名凭证未出现在验证器的任何适用信任列表中。	C2PA声明签名框

15.3. 显示清单信息

清单消费者不应显示来自无效清单或无效资产的数据。若清单消费者选择显示此类数据，则必须在显示内容中包含：

- 关于数据无效性的警告，
- 数据不得归因于清单签名者的警示，若为成分清单，还需额外标注数据亦不归因于资产清单签名者。

注意	在编写场景时，应更醒目地提示警告，以便创作者能够是否继续使用该资产。

15.4. 确定哈希算法

15.4.1. 针对哈希URI

C2PA清单的各个部分均采用hashed_uri结构来封装URI、其哈希值以及（可选的）用于计算哈希的算法。若hashed_uri结构中存在`alg`字段，则应将其作为哈希算法使用。若`hashed_uri`结构中未包含`alg`字段，则应通过评估最近包含`alg`字段的嵌套结构来确定哈希算法。若上述位置均未找到alg字段，则应使用声明中的alg字段值作为哈希算法。若所有位置均未包含alg字段，则声明应被拒绝并返回失败代码`algorithm.unsupported`。

15.4.2. 针对哈希扩展URI

C2PA清单的部分内容采用hashed_ext_uri结构封装外部URI、其哈希值及计算哈希所用的算法。若hashed_ext_uri结构中存在`alg`字段，则应采用该字段作为哈希算法。若hashed_ext_uri结构中未包含alg字段，则应使用`algorithm.unsupported`失败代码。

注	`alg`字段在`hashed_ext_uri`中为必填项，因此无需通过递归过程确定哈希算法。

15.4.3. 算法验证

一旦确定哈希算法，应将其与第13.1节“哈希”中的允许列表或弃用列表中的值进行比对。若该算法未出现在任一列表中，则应以算法不支持（`algorithm.unsupported`）失败代码拒绝该声明。若该算法出现在弃用列表中，则应向该声明发出算法已弃用（`algorithm.deprecated`）信息代码。

15.5. 定位活动清单

15.5.1. 通用规则

C2PA清单存储库超级框中的最后一个C2PA清单超级框应视为活动清单，但定位C2PA清单存储库可能需要在多个潜在位置进行搜索。

15.5.2. 嵌入式

15.5.2.1. 通用规则

验证器应通过资产内嵌的有效验证程序，在标准清单嵌入位置定位C2PA清单存储库。但若资产通过HTTP连接获取，验证器可参照下文"链接标头条款"所述，通过查找链接标头判断C2PA清单存储库是否存在。

NOTE

检查链接标头（若存在）可让验证器确定C2PA清单存储是否存在，而无需下载整个资源。这对大型资源或流式传输资源尤为有用。

若资产中存在多个C2PA清单存储，则应视为全部无效，验证过程应视同未找到任何清单。当该资产作为组件添加时，所有嵌入的C2PA清单均不得包含在组件断言中。

15.5.2.2. PDF 的特殊注意事项

PDF文件支持名为"增量更新"的技术，该技术将信息追加至文档末尾而非修改原始内容。这要求PDF文件支持多个C2PA清单存储库——尽管每个更新部分仅允许存在一个。

若单个更新段落中存在多个C2PA清单存储，则应视为全部无效，验证过程应视同未找到任何清单。但PDF早期更新版本或原始PDF中存在的C2PA清单存储仍视为有效，并按相应规则处理。

15.5.3. 引用或URI方式

15.5.3.1. 通过引用

若未嵌入C2PA清单存储，应通过以下方式尝试在远程位置定位：

- 若资产通过HTTP连接获取，下文"链接标头"条款将说明如何通过链接标头查找清单。
- 如果资产在标准资产位置（即C2PA清单之外）存在任何XMP文件，且该XMP包含`dcterms:provenance`键，则应使用提供的URI定位活动清单。

- 若资产为含C2PA表的字体文件，且其activeManifestUriLength非零，则应使用指定URI定位活动清单。
- 若未找到 C2PA 清单存储库，验证器应在相同路径或 URI 下查找文件，但需确保文件扩展名为 `.c2pa`。若仍未找到 C2PA 清单存储库，验证器可自行判断最合适的额外位置进行搜索，例如文件系统的子文件夹。

注 验证器不仅限于上述位置，还可选择在其他位置进行搜索。

若清单文档记载存在于远程位置但实际缺失，或该位置当前不可用（如离线场景），则应使用 `manifest.inaccessible` 错误代码报告此情况。

有关C2PA清单存储库的IANA媒体类型的信息可在外部清单部分中找到。

15.5.3.2. 通过 `Link` 标头

若资产通过HTTP连接获取，验证器应在HTTP响应头中查找符合RFC 8288定义的`Link`标头，该标头需包含`rel=c2pa-manifest`参数。若存在该标头，则可通过其URI引用获取C2PA清单存储。该URI将采用标准http或https格式，例如https://c2pa.org/image.c2pa。

也可通过JUMBF URI片段引用嵌入资产内的C2PA清单存储库。该URI需包含指向C2PA清单存储库超级框https://c2pa.org/image.jpg#jumbf=c2pa的JUMBF URI片段。禁止引用C2PA清单存储库内的特定C2PA清单，验证器应忽略JUMBF URI片段中的任何子标签部分。

注 HTTP指RFC 7230定义的*超文本传输协议*，而非特定URL方案`http://`。

15.5.4. 解压缩

如前所述，标准清单和更新清单均可进行压缩。当遇到压缩清单时，验证器应在执行标准验证流程前对其进行解压。若压缩清单的brob框所含数据既非标准清单也非更新清单，或解压失败，验证器应以失败代码`manifest.compressed.invalid`拒绝该清单。

15.5.5. 验证匹配

验证者可能需要验证定位到的C2PA清单存储是否确实与该资产相关联。

若定位到C2PA清单存储库，则应使用其活动清单中的硬绑定断言来验证：该清单是否匹配，以及资产是否在未更新清单的情况下被修改。若硬绑定不匹配，则无法确定原因属于：(a) 资产被修改，或 (b) 定位到了错误的C2PA

清单存储库导致。因此验证者应将此视为硬绑定不匹配，并根据所用数据哈希断言类型拒绝清单：若使用数据哈希断言则返回 assertion.dataHash.mismatch 失败代码，若使用通用箱哈希断言则返回 assertion.boxesHash.mismatch，若使用集合数据哈希断言则返回 assertion.collectionHash.mismatch，若使用 BMFF 哈希断言则返回 assertion.bmffHash.mismatch。

15.6. 索赔定位与验证

15.6.1. 定位

一旦定位到待验证的清单文件（下文称为"当前清单"），即可通过在当前清单中查找标记为c2pa.claim.v2（或旧版索赔结构文件的c2pa.claim）且JUMBF类型UUID为6332636C-0011-0010-8000-00AA00389B71（c2cl）的JUMBF类型UUID。需注意新旧声明格式（分别标记为c2pa.claim.v2与c2pa.claim）的JUMBF类型UUID保持一致。当前清单中仅允许存在一个此类标记框。若检测到多个标记框，则应以失败代码 claim.multiple 拒绝该 C2PA 清单。

15.6.2. 验证

如果声明的内容不符合规范的CBOR格式，则该声明应被拒绝，并返回失败代码 claim.cbor.invalid。

注	规范CBOR格式定义详见RFC 8949附录C。

对于"c2pa.claim.v2"声明，CBOR对象中应包含以下字段。若存在任何缺失，则声明应被拒绝并返回失败代码：claim.malformed。

- instanceID
- 签名
- 创建断言
- 声明生成器信息

若 claim_generator_info 字段未包含 name 字段，则声明应被拒绝，并返回失败代码 claim.malformed。

如果在声明映射（claim-map）或声明映射v2（claim-map-v2）的声明生成器信息字段（claim_generator_info）所引用的生成器信息映射（generator-info-map）中存在图标字段，则其值应按照第15.10.3.3节“引用验证”所述进行验证。

15.7. 验证签名

从声明的签名字段值中检索签名的 URI 引用并解析该 URI

用于获取COSE签名的引用。签名应嵌入第11.1.4节"C2PA框详情"所述的同一清单中。若签名URI未指向同一C2PA清单框内的位置（即self#jumbf位置），则声明应被拒绝。若该字段不存在或URI无法解析，则声明应以failure code `claimSignature.missing`被拒绝。

若签名与声明未包含在同一C2PA清单中，则该C2PA清单应视为无效。

对于所有类型的C2PA清单，签名所用凭证的验证应遵循第14章《*信任模型*》进行。

若凭证不符合其类型要求，则应以失败代码 `signingCredential.invalid` 拒绝该声明。若签名算法未列入第13.2节"数字签名"中的允许或弃用列表，则应以失败代码 `algorithm.unsupported` 拒绝该声明。

随后需验证从凭证到适用信任锚列表条目间的信任链。若无法验证该信任链，则应以失败代码`signingCredential.untrusted`拒绝该声明；否则声明签名应被赋予成功代码`signingCredential.trusted`。

若声明尚未被拒绝，则应按第13.2节"数字签名"中规定的流程继续验证。若签名验证失败，则声明应被拒绝并返回失败代码`claimSignature.mismatch`；否则声明签名应被赋予成功代码`claimSignature.validated`。

在本章剩余部分中，"标头"指COSE签名中受保护与不受保护标头参数的并集。除非第13.2节"数字签名"或第14.5节"X.509证书"另有规定，标头可出现在任一集合中。COSE标头详见RFC 8152第3节。

15.8. 验证时间戳

15.8.1. 获取时间戳令牌

15.8.1.1. 嵌入声明签名中

若`sigTst`或sigTst2头部存在，则tstTokens数组应仅包含单个`tstToken`。若头部包含多个`tstToken`，验证器应发出`timestamp.malformed`信息代码并忽略这些时间戳。

支持 `sigTst` 的验证器应执行以下步骤来验证时间戳响应：

- 从 `tstToken` 中获取 `val` 属性，该属性应为符合 RFC3161 标准的 `TimeStampResp`（时间戳响应）。

- 检查PKIStatusInfo字段的值，该值即为TimeStampResp的status字段值。
  - 若其值非0（授予）或1（带修改授予），验证器应发出timeStamp.malformed 信息代码，忽略该时间戳。
  - 若该值为0（授予）或1（授予并修改），则继续执行后续时间戳验证流程（详见下文说明）。
- 检索 TimeStampResp 中 timeStampToken 字段的值，用于后续验证流程。

sigTst2 的验证器应从 tstToken 中检索 val 属性，该属性应为符合 RFC3161 标准的时间戳令牌（TimeStampToken, TST）。

15.8.1.2. 由时间戳断言引用

若验证者已在sigTst或sigTst2头部定位到通过验证的时间戳令牌（依据第15.8.2节"验证时间戳令牌"），则应跳过此步骤。当不存在此类头部或其中定位的时间戳令牌未通过验证时，应执行此步骤。

若验证者先前已定位任何时间戳声明，并将其维护在C2PA清单标识符与时间戳令牌的映射表中，则验证者应检查当前C2PA清单的标识符是否存在于该映射表中。若存在，验证器应在第15.8.2节"验证时间戳令牌"所述的验证流程中，使用映射中与该标识符关联的时间戳令牌。若映射中存在多个该标识符对应的时间戳令牌，验证器应逐一尝试直至找到通过验证的令牌（其余令牌应忽略）。若标识符未出现在映射中，则不触发错误，这仅表示当前上下文中不存在与该C2PA清单关联的时间戳令牌。

15.8.2. 验证时间戳令牌

所有验证器均应按以下流程继续操作：

- 如果时间戳令牌中的签名算法未列入第13.2节"数字签名"中的允许或弃用列表，则验证器应发出时间戳.untrusted信息代码并忽略该时间戳。
- 根据RFC 2630第5.6节所述方法验证timeStampToken中的签名。若签名无效，验证器应发出timestamp.mismatch信息代码并忽略该时间戳。
- 若时间戳令牌（timeStampToken）的签名字段（messageImprint）为空，则验证器应发出时间戳.格式错误信息代码并忽略该时间戳。

  timestamp.malformed 信息代码并忽略该时间戳。
- 若消息印记哈希算法未列入第13.1节"哈希算法"的允许或弃用列表，验证器应发出timestamp.untrusted信息代码并忽略时间戳。
- 验证消息指纹字段（位于时间戳令牌中）的值是否与声明（v1,

sigTst) 或待验证C2PA清单中COSE_Sign1_Tagged结构的签名字段(v2, sigTst2)，具体参见第10.3.2.5.2节"选择有效负载"。若值不匹配，验证器应发出timestamp.mismatch信息代码并忽略该时间戳。

- 验证时间戳令牌是否包含证书字段，该字段提供的证书集中能否找到TSA证书，以及能否从TSA证书构建至C2PA TSA信任列表（或验证器为此目的持有的其他信任锚列表）条目的信任链。若证书无法定位或无法构建信任链，验证器应发出 timestamp.untrusted 信息代码并忽略该时间戳。
- 验证时间戳令牌中genTime字段所示的认证时间是否位于TSA签名证书及所有CA证书（直至信任锚点）的有效期内。若不符合要求，验证器应发出timestamp.outsideValidity信息代码并忽略该时间戳。
- 若时间戳验证未因上述任一条件终止或失败，验证器应发出 timeStamp.trusted 和 timeStamp.validated 成功代码。
- 若验证器同时发出 timeStamp.trusted 和 timeStamp.validated 成功代码，则验证器应验证时间戳机构（TSA）所认证的时间（位于 timeStampToken 的 genTime 字段中）是否处于声明签名证书及所有CA证书（直至信任锚点）的有效期内。若不符合，验证器应以claimSignature.outsideValidity失败代码拒绝该声明。

注： 时间戳即使在签名机构的签名凭证过期后仍然有效，因此只要所证明的时间落在时间戳机构证书的有效期内。这是仅对时间戳机构延伸的特殊信任类型。

在验证时，当存在可信且已验证的时间戳时，验证者应使用该认证时间（而非当前时间）来判定签名证书和时间戳机构证书的时间有效性。

注 本文件不要求在签名时捕获时间戳机构证书的吊销状态，亦不要求在验证时对此进行验证。

若sigTst和sigTst2标头均不存在，或至少存在其中一个但其时间戳令牌不符合上述要求，则当验证时的当前时间处于签名者证书及所有CA证书（直至信任锚点）的有效期内时，C2PA清单即为有效。若满足上述条件，验证器应返回成功代码 claimSignature.insideValidity；若不满足，则应以失败代码 claimSignature.outsideValidity 拒绝该 C2PA 清单。

15.8.3. 验证"声明的签名时间"

验证者可选择验证由iat受保护头部值所证明的"声明签名时间"。若存在iat头部，验证者可验证该证明时间是否位于签名者证书及所有CA证书（直至信任锚点）的有效期内，且不晚于任何关联可信时间戳所证明的时间。若验证者执行此值验证且其处于有效期内，

验证器应返回`timeOfSigning.insideValidity`信息代码；若超出有效期，则应返回`timeOfSigning.outsideValidity`信息代码。

15.9. 验证凭证吊销信息

验证器应尝试查明签名者证书及其信任链中所有CA证书的吊销状态。

对于CA证书，验证者应依据RFC 5280第4.2.2.1节所述的权威信息访问(AIA)扩展确定撤销状态。若AIA扩展表明OCSP可用，验证者应利用C2PA清单中包含的相关OCSP响应。

若验证器判定CA证书在可信时间戳标识的时间点（或无可信时间戳时在当前时间点）已被吊销，则应以`signingCredential.untrusted`失败状态拒绝该声明签名。

对于签名者的证书，验证器应采用以下流程：

- 若证书不支持吊销状态，或证书颁发者未提供查询吊销状态的方法，验证器应将凭证视为未被吊销。
- 若声明生成器在COSE_Sign1结构的rVals头部"钉接"了OCSP响应，验证器应按第15.9.1节"通过C2PA清单存储库中的OCSP响应确定吊销状态"所述方式解码并验证这些钉接的OCSP响应。
- 若后续声明生成器在C2PA清单存储库的其他C2PA清单中添加了证书状态声明，验证器应在第15.9.1节"通过C2PA清单存储库中的OCSP响应确定吊销状态"所述的验证流程中使用这些OCSP响应。若发现该证书存在多个OCSP响应，验证器应逐一尝试直至其中一个通过验证（随后应忽略其余响应）。

若在C2PA清单存储库中未找到吊销信息，且验证器处于在线状态，同时验证器需要验证证书的吊销状态，则验证器应通过查询OCSP响应器来确定证书的吊销状态，具体操作参见第15.9.2节“通过在线OCSP响应确定吊销状态”。

15.9.1. 通过C2PA清单存储库中的OCSP响应确定吊销状态

验证者应按RFC 6960的要求解码OCSP响应，特别是第3.2节中的第1至4项要求。若OCSP响应被接受，且满足以下所有要求，则可确定相关证书在签名时未被撤销。

- 声明签名包含由有效签名时间戳提供的认证时间戳。
- 在OCSP响应的`tbsResponseData`字段的响应数组中存在一个`SingleResponse`，使得以下所有条件均成立：

- 当前时间不早于 `thisUpdate`。
- 时间戳的认证时间：
  - 早于thisUpdate，或
  - 若存在nextUpdate字段，则该时间戳位于 `(thisUpdate, nextUpdate)` 区间内，或
  - 若未包含nextUpdate字段，则该时间戳需落在 `(thisUpdate, producedAt + 24小时)` 区间内，其中producedAt为包含响应数据的字段。
- 单一响应的`certStatus`字段为有效，或已撤销但撤销原因标记为 `removeFromCRL`。

> **注** 在撤销原因的值中，`removeFrom`CRL具有独特性，因为它相当于一个有效的响应。尽管属于撤销响应类型，但此响应表明该证书先前因完整性问题被暂时"暂停"（certificateHold原因），但问题现已解决，签发者声明该证书仍可信赖（参见RFC 5280）。

- 响应的OCSP签名者符合RFC 6960第4.2.2.2节定义的"授权响应者"身份。

验证方应检查任何撤销响应的`revocationReason`字段，以区分`removedFromCRL`情况与实际吊销情况。

若C2PA清单存储库中任何OCSP响应满足上述条件，则签名时应视为证书未被吊销，验证器应返回`signingCredential.ocsp.notRevoked`成功代码。

否则，当C2PA清单存储库中的OCSP响应满足上述所有条件但`certStatus`字段为`revoked`时，该证书在签名时应被视为已撤销，声明应被拒绝并返回`signingCredential.ocsp.revoked`失败代码。

15.9.2. 通过在线OCSP响应确定吊销状态

若针对特定证书，C2PA清单存储库中所有OCSP响应均不符合第15.9.1节规定条件，“通过C2PA清单存储中的OCSP响应确定吊销状态”，或声明签名未包含时间戳时，验证器可选择依据RFC 6960规范，使用通过RFC 6960第3.1节获取的响应方访问位置查询OCSP响应方。

> **注** 查询凭证状态的方法可能向观察者暴露被验证资产的身份，因此此查询为可选操作。

如果验证器选择不执行执行一个在线OCSP检查，它应签发签名凭证.ocsp.跳过`签名凭证.OCSP跳过`信息代码。

如果验证器尝试查询OCSP响应器但无法收到响应，则验证器应发出`signingCredential.ocsp.inaccessible`信息代码。

若收到响应且该响应符合RFC 6960第3.2节要求1至4，则需满足以下任一条件方可确认签名者证书在签名时未被吊销：

- 该声明signature具有具有有效时间戳，且该经认证的时间落在在该响应的(本次更新，下次更新)时间间隔内，或
- 该声明签名未包含有效时间戳，但当前实际时间处于响应的响应的(本次更新，下次更新)时间段内，且同时满足以下两个条件：
- 响应中的certStatus字段状态良好，或已被撤销但撤销原因标记为removeFromCRL，且
- 响应的OCSP签名者符合RFC 6960第4.2.2.2节定义的"授权响应者"。

若响应的certStatus字段显示证书已撤销，但撤销原因非removeFromCRL，则需同时满足以下条件方可判定签名者证书在签名时未被撤销：

- 清单具有有效的时间戳，且证明时间落在响应的(本次更新，下次更新)响应的(本次更新,下次更新)时间区间内，且
- 响应中的撤销时间晚于证明的时间戳。

若满足上述条件，则在签名时应视为证书未被撤销，验证器应返回签名凭证OCSP未撤销状态的成功代码。

否则：

- 若，则的certStatus字段在的响应中为unknown，则声明应被拒绝并返回失败代码signingCredential.ocsp.unknown失败代码。
- 否则，应视为证书在签名时已被吊销，并应以signingCredential.ocsp.revoked失败代码。

15.10. 验证断言

15.10.1. 验证清单类型的正确断言

15.10.1.1. 通用规则

根据清单类型，存在必需或禁止的断言。验证器应检查必需且未被允许的断言。

15.10.1.2. 标准清单断言

若为标准清单：

1. 验证是否存在唯一的内容硬绑定断言——即 `c2pa.hash.data`、`c2pa.hash.boxes`、`c2pa.hash.collection.data`、`c2pa.hash.bmff.v2`（已弃用）或 `c2pa.hash.bmff.v3` 之一。若未包含此类断言，清单应以失败代码 `claim.hardBindings.missing` 拒绝。若存在多个此类断言，清单应以失败代码 `assertion.multipleHardBindings` 拒绝。
2. 验证是否存在零个或一个关系为 `parentOf` 的 `c2pa.ingredient` 断言。若存在多个，则清单应以 `manifest.multipleParents` 失败代码被拒绝。
3. 验证`c2pa.`created或`c2pa.`opened操作仅包含于单个actions断言中。

15.10.1.3. 更新清单断言

若为更新清单：

1. 验证清单中仅存在一个成分断言且其关系为`parentOf`。若不满足此条件（即断言缺失、存在多个断言或关系值非`parentOf`），则应以失败代码`manifest.update.wrongParents`拒绝该清单。
2. 验证不存在 `c2pa.hash.data`、`c2pa.hash.boxes`、`c2pa.hash.collection.data`、`c2pa.hash.bmff.v2`（已弃用）、`c2pa.hash.bmff.v3` 或缩略图断言。若存在上述断言，则应以失败代码 `manifest.update.invalid` 拒绝该清单。
3. 验证是否存在 `c2pa.hash.multi-asset` 断言。若存在，则应以 `manifest.update.invalid` 失败代码拒绝该清单。
4. 若存在一个或多个 `c2pa.actions` 或 `c2pa.actions.v2` 断言，需验证此类断言中 `actions` 数组内每个操作的 `action` 字段是否属于《更新清单规范》中支持的值。若不符合要求，清单应被拒绝并返回失败代码 `manifest.update.invalid`。

15.10.2. 准备编辑后的断言列表

验证器处理声明时，应根据其 `redacted_assertions` 字段列出的每个 JUMBF URI，为每个成分清单（若存在）收集其被屏蔽的断言集合。声明的 `redacted_assertions` 字段绝不应包含指向其自身任何断言的 JUMBF URI。

注	在最终资产的来源链中，任何时候均可从成分资产中删除声明 历史中进行编辑，且不一定由首次将原料资产作为原料使用的声明生成者执行。

更多详情请参阅第 15.11.3.2 节“执行显式验证”。

15.10.3. 断言验证

15.10.3.1. 通用

声明中 created_assertions 和 gathered_assertions 字段（以及 v1 声明的 assertions 字段）中的每个断言都是一个 hashed_uri 结构。对于每个断言，验证器应首先确定 url 字段中的 URI 引用是否在编辑后的断言列表中。

NOTE

尽管 gathered_assertions 字段中列出的断言并非由
声明生成器创建，但它们仍是声明的一部分，因此也需根据此验证算法进行验证。

若该断言存在于被屏蔽断言列表中，且其标签为 c2pa.actions 或 c2pa.actions.v2，则声明应被拒绝并返回失败代码 assertion.action.redacted，因为c2pa.actions和c2pa.actions.v2断言不得被屏蔽。若该断言属于被屏蔽断言列表，且其标签为与内容硬绑定的断言（即 c2pa.hash.data、c2pa.hash.boxes、c2pa.hash.collection.data、c2pa.hash.bmff.v2（已弃用）或 c2pa.hash.bmff.v3），则应以断言失败代码 assertion.dataHash.redacted 拒绝该声明，因这类断言不得被编辑。否则，被编辑的断言视为有效，验证将根据断言类型继续进行。

对于所有其他断言（未出现在已编辑断言列表中），需解析url字段中的URI引用以获取其数据。若该URI未指向同一C2PA清单内的位置（即self#jumbf位置），则应以断言超出清单范围（assertion.outsideManifest）的失败代码拒绝该声明。若无法解析URI且获取数据失败，则应以失败代码 assertion.missing拒绝该声明。

遵循第15.4节“确定哈希算法”中的流程确定哈希算法及可能的失败代码。使用该算法及第8.4.2.3节"JUMBF框哈希处理"所述流程计算断言哈希值，并与哈希字段值比对。若不匹配，则以失败代码 assertion.hashedURI.mismatch 拒绝该声明；否则记录成功代码 assertion.hashedURI.match。

如果标准断言的内容不符合规范的CBOR格式或非标准JSON格式，则应以断言无效（assertion.cbor.invalid）或断言JSON无效（assertion.json.invalid）的失败代码拒绝该声明。

注注

格式正确的CBOR定义详见RFC 8949附录C。

RFC 8259第2节定义了JSON数据遵循的语法规范。

若断言存储库中存在的断言未被声明中的 created_assertions 或 gathered_assertions 数组（或 v1 声明中的 assertions 数组）的任何元素引用，则应以失败代码 assertion.undeclared 拒绝该声明。

对于声明中 redacted_assertions 数组中的每个 URI，若该 URI 指向声明自身的清单，则声明应被拒绝，失败代码为 assertion.selfRedacted。声明不允许屏蔽其自身的

断言。

15.10.3.2. 特定断言验证

对于每个断言，验证器应检查该断言的标签。若其标签在下列列表中存在，则验证器应执行该断言类型的特定验证步骤。若断言标签未在下列列表中出现，则该类型断言无需执行除已描述步骤之外的额外验证操作。

- c2pa.cloud-data，第15.10.3.2.1节，“c2pa.cloud-data验证”
- c2pa.actions 或 c2pa.actions.v2，第 15.10.3.2.2 节，“c2pa.actions 验证”
- c2pa.metadata，第15.10.3.2.3节，“c2pa.metadata验证”

注释	成分 assertions (c2pa.ingredient 或 c2pa.成分.v2 或 c2pa.ingredient.v3) 在验证流程的其他环节需接受额外验证（参见第15.11节“验证成分”）。

若标准断言中任意字段的值为hashed_uri或hashed_ext_uri，验证器应执行第15.10.3.3节"引用验证"所述步骤，但c2pa.ingredient.v3中的activeManifest字段除外——该字段的特殊验证行为在第15.11.3节“成分断言验证”中规定特殊验证行为。

15.10.3.2.1. c2pa.cloud-data 验证

若断言标签为 c2pa.cloud-data：

1. 检查断言是否包含以下字段：label、size、location 和 content_type。若存在任一字段缺失，则应以断言错误代码 assertion.cloud-data.malformed拒绝该声明。
2. 若外部声明的 label 字段为 c2pa.hash.data、c2pa.hash.boxes、c2pa.hash.collection.data、c2pa.hash.bmff.v2（已弃用）或 c2pa.hash.bmff.v3，则声明应被拒绝，并返回失败代码 assertion.cloud-data.hardBinding。
3. 若清单为更新清单且外部断言的标签字段为 c2pa.actions 或 c2pa.actions.v2，该声明应被拒绝，并返回失败代码 assertion.cloud-data.actions。
4. 位置字段应根据第15.10.4.2节“外部引用验证”进行验证。

15.10.3.2.2. c2pa.actions 验证

若断言标签为 c2pa.actions 或 c2pa.actions.v2：

1. 确保该声明包含一个动作字段。若不包含，则该声明应被拒绝并返回一个失败代码为断言.操作.格式错误.
2. 对于动作列表中的每个动作：
   a. 若操作字段为 c2pa.created 或 c2pa.opened，则应以

断言失败代码为 assertion.action.malformed，除非同时满足以下所有条件：

i. 该断言是创建断言数组或收集断言数组中首个动作断言

数组（v2声明），或v1声明中assertions数组的首个动作断言，且

ii. 且该操作是本断言中 actions 数组的首个元素。

b. 若操作字段为 c2pa.opened、c2pa.placed 或 c2pa.removed：

i. 若该操作无参数字段，或该字段值为空，则声明应被拒绝，并返回失败代码 assertion.action.ingredientMismatch。

ii. 如果操作的参数字段中不包含成分字段（或 c2pa.actions 的成分字段），则应以失败代码 assertion.action.ingredientMismatch拒绝该声明。

iii. 若成分字段的值不是包含至少一个元素的数组，则声明应被拒绝，并返回失败代码 assertion.action.ingredientMismatch。

iv. 检查对成分断言的引用：

A. 对于 c2pa.opened：检查 ingredients 字段（或 c2pa.actions 的 ingredient 字段）是否包含且仅包含一个有效的哈希 URI，该 URI 可解析为当前清单中关系字段为 parentOf 的成分断言。若不满足，则应以失败代码 assertion.action.ingredientMismatch 拒绝该声明。

B. 对于 c2pa.placed：检查 ingredients 字段（或 c2pa.actions 的 ingredient 字段）是否包含一个或多个有效的哈希 URI，且每个 URI 均可解析为当前清单中 relationship 字段为 componentOf 的 ingredient 断言。否则，应以 assertion.action.ingredientMismatch 失败代码拒绝该声明。

C. 对于 c2pa.removed：检查 ingredients 字段（或 c2pa.actions 的 ingredient 字段）是否包含一个或多个有效的哈希 URI，且每个 URI 均可解析为另一个清单中的 ingredient 断言，且其 relationship 字段为 componentOf。若不满足，则声明应被拒绝，并返回失败代码 assertion.action.ingredientMismatch。

c. 若操作字段为 c2pa.transcoded 或 c2pa.repackaged：

i. 若存在 ingredients 字段（或 c2pa.actions 的 ingredient 字段），则需验证该字段的每个元素是否为有效哈希 URI，且能解析为当前清单中具有 parentOf 关系的成分断言。若不满足，则应以 assertion.action.ingredientMismatch 失败代码拒绝该声明。

d. 如果操作字段为 c2pa.redacted：

i. 检查参数对象成员中的编辑字段是否包含JUMBF URI。若未包含JUMBF URI，或无法解析为断言，则应以断言失败代码 assertion.action.redactionMismatch拒绝该声明。

e. 若action-common-map-v2中存在softwareAgent字段，或actions-map-v2的softwareAgents字段中列出一个或多个软件代理，则：列于 actions-map-v2 的 softwareAgents 字段中：

i. 若 `generator-info-map` 中存在 icon 字段，则应按第 15.10.3.3 节“引用验证”所述进行验证。

f. 对于模板列表中的每个模板：

i. 若`action-template-map-v2`中存在icon字段，则应按第15.10.3.3节"引用验证"所述进行验证。

15.10.3.2.3. `c2pa.metadata` 验证

若断言标签为 `c2pa.metadata`，验证器应确保该断言不包含允许列表外的字段。若断言中存在任何未在允许列表内的字段，则应以 `assertion.metadata.disallowed` 失败代码拒绝该声明。

注 此验证要求需要验证器解析断言中包含的 JSON-LD 数据。

15.10.3.2.4. `c2pa.time-stamp` 验证

若断言标签为 `c2pa.time-stamp`，验证器应确保该断言为结构良好的 CBOR 格式，包含单个映射（主类型 `5`）且至少含一对键值。若不符合此要求，则应以失败代码 `assertion.timestamp.malformed` 拒绝该声明。

由于时间戳令牌的验证遵循第15.8.2节"验证TimeStampToken"所述流程，验证器需存储该时间戳令牌（及其关联的C2PA清单标识符）以备后续使用。

15.10.3.3. 引用验证

某些C2PA标准断言支持通过`hashed_uri`和`hashed_ext_uri`引用C2PA清单中的其他框。例如，在动作、成分和缩略图断言中可能存在各种引用。

对于标准断言中的所有hashed_uri和hashed_ext_uri字段（`c2pa.ingredient.v3`中的`activeManifest`字段除外——其特殊验证行为详见第15.11.3节"成分断言验证"），验证器应执行以下验证：对于验证器选择检索其资源的`hashed_ext_uri`，验证器应执行第15.10.4.2节"外部引用验证"所述步骤。对于`hashed_uri`，验证器应执行以下步骤。

`hashed_uri` 的目标地址位于其 `url` 字段中。若该字段缺失或目标地址无法定位（即数据未存放于预期位置），则应视为验证失败，并返回代码 `hashedURI.missing`。

若能定位目标，则按以下步骤操作：. 参照第15.4节“确定哈希算法”中的流程确定哈希算法及可能的失败代码。. 确保哈希字段存在于hashed_uri结构中。若不存在，则应以`hashedURI.mismatch`失败代码拒绝该声明。. 使用确定的哈希算法及第8.4.2.3节“哈希JUMBF盒子”所述流程计算断言的哈希值。. 将计算出的哈希值与哈希字段中的值进行比对。若不匹配，则应以`hashedURI.mismatch`失败代码拒绝该声明。

第8.4.2.3节"JUMBF数据框哈希处理"所述方法计算断言的哈希值。. 将计算结果与哈希字段值比对。若不匹配，则以`hashedURI.mismatch`失败代码拒绝该声明。

15.10.4. 外部数据验证

15.10.4.1. 通用规范

云数据断言的内容包含指向外部数据的URI引用及其哈希值，其验证方式与其他断言相同，但这些引用不会作为标准验证流程的一部分进行检索和验证。验证器必须先成功验证断言，再尝试检索所引用的外部数据。验证器不得尝试从被拒绝的声明中检索外部数据。由于外部数据检索属于可选操作，无法检索或验证外部数据不应导致声明被拒绝。

若验证器选择检索云数据断言中的外部数据，则应执行第15.10.4.2节"外部引用的验证"所述步骤。

15.10.4.2. 外部引用的验证

验证云数据断言中引用的外部数据时，应采用以下流程：

1. 解析url字段中的URI引用以获取其数据。若`url`字段不存在或URI无法解析且数据无法获取，验证器应终止外部数据的检索尝试。
2. 若获取的数据大小与size字段值不符，验证器应向应用程序返回断言失败代码`assertion.hashedURI.mismatch`，且不得提供所获取的数据。
3. 验证HTTP响应的`Content-Type`头部返回的内容类型是否与声明的内容类型一致。若不匹配，验证器应向应用程序返回失败代码`assertion.hashedURI.mismatch`，且不提供检索到的数据。声明的内容类型由以下因素决定：
    a. 对于外部数据，内容类型由 `hashed_ext_uri` 结构的 `dc:format` 字段决定
    结构的`dc:format`字段确定。若该字段缺失，内容类型验证始终成功。
    b. 对于云数据断言，若其位置字段中存在dc:format字段，则该字段决定内容类型，此时云数据断言的`content_type`字段值将被忽略。若位置字段不包含`dc:format`字段，则断言的`content_type`字段决定内容类型。
4. 根据第15.4.2节"针对哈希扩展URI"或可能的失败代码，确定应使用的哈希算法。
5. 使用确定的哈希算法及第8.4.2.3节"JUMBF数据框哈希处理"所述流程，对检索到的内容计算数据哈希值。对于外部数据，需将哈希算法与精确检索内容作为哈希函数输入。
    a. 将计算出的哈希值与哈希字段中的值进行比较。若哈希字段不存在或两者不匹配，验证器应向应用程序返回`assertion.hashedURI.mismatch`失败代码，且不提供检索到的数据。

b. 否则，验证器应记录成功代码 `assertion.hashedURI.match` 并将检索到的数据提供给应用程序。

15.11. 验证成分

15.11.1. 说明

验证器应执行对所提交资产及其有效清单的验证步骤。若任何步骤判定有效清单无效，则该清单应被拒绝并返回指定的失败代码。

资产的活动清单可能通过成分声明列出一种或多种成分。其中某些成分可能关联其自身的清单，而这些清单本身也可能包含成分及其清单。

15.11.2. 处理成分清单

15.11.2.1. 组件中的标准清单

处理标准清单时，验证器应验证每个组件（无论其
关系字段值），具体如下所述。

15.11.2.2. 更新成分中的清单

对于更新清单，应按以下所述验证更新清单的`parentOf`成分。

15.11.2.3. 组件中的时间戳清单

重要提示	此功能已弃用，现推荐使用时间戳断言。以下信息保留供历史参考。

组件中发现的任何时间戳清单均应忽略。

15.11.3. 组件断言验证

15.11.3.1. 验证概述

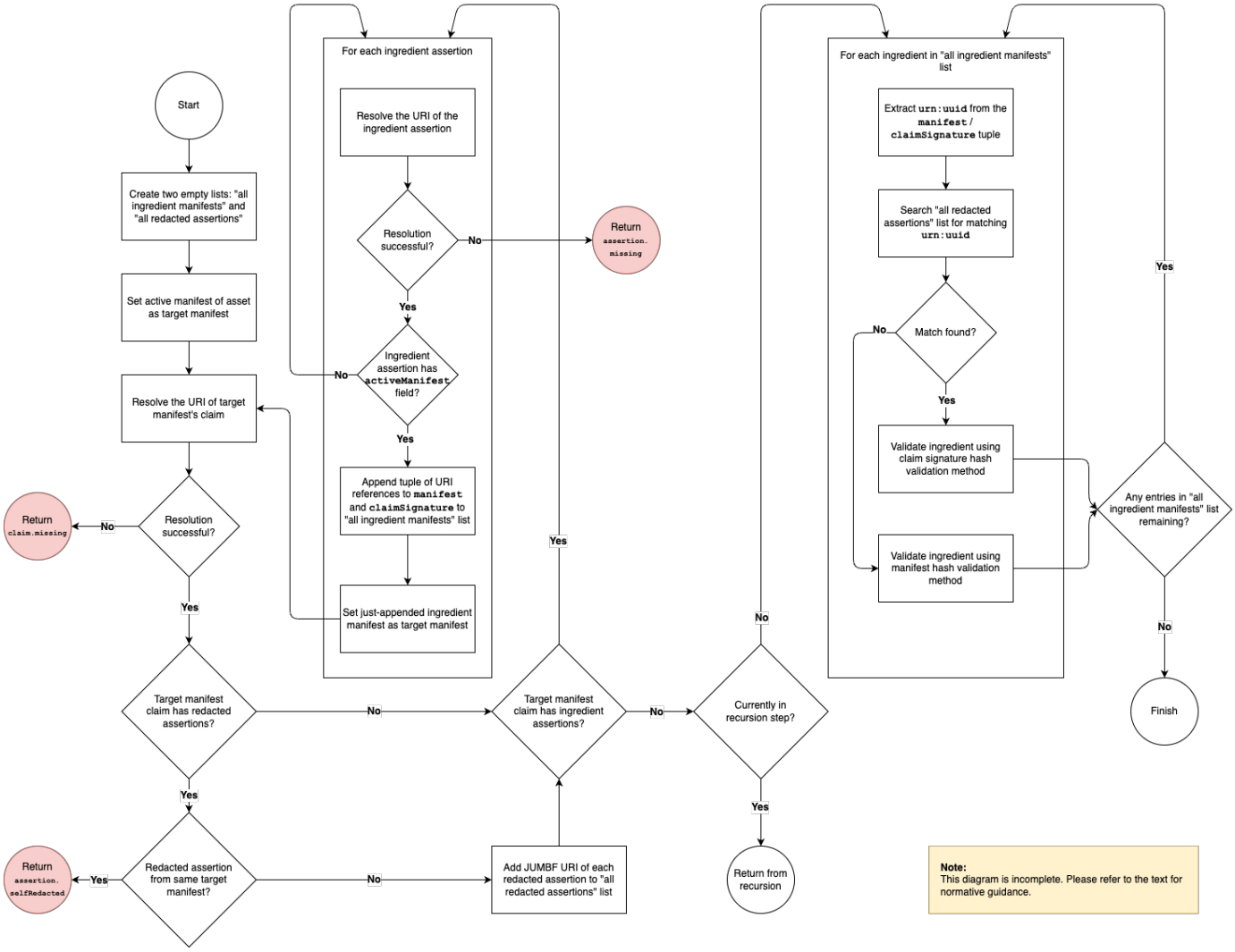

图14. 成分验证流程

图14“成分验证”流程图描述了验证特定C2PA清单中所有成分声明的过程。

注 若视觉呈现与文本存在任何差异，以文本为准。

15.11.3.2. 执行显式验证

若成分声明中未包含关系字段，则该声明应被拒绝，并返回失败代码 `assertion.ingredient.malformed`。

关系字段值应为以下之一：`parentOf`、`inputTo` 或 `componentOf`。若关系字段值不符合上述要求，则声明应被拒绝，并返回失败代码 `assertion.ingredient.malformed`。

15.11.3.3. 执行递归验证

验证器应递归验证资产中的所有成分清单，例如采用深度优先搜索方式：

如下所述。验证器不必完全按照所述方式实现该算法，但验证结果应等同于该算法的结果。

1. 创建两个空列表：
    a. 一个用于存储资产使用过的所有成分清单的哈希URI值的列表，无论这些清单出现在资产血统的哪个位置。
    b. 另一个列表用于存储该资产血统中所有被编辑断言的JUMBF URI。
2. 将待验证资产的活动清单设为目标清单
3. 开始递归。
4. 定位声明，具体操作参见第15.6节“定位和验证声明”。若定位失败，则以失败代码claim.missing拒绝该声明。
5. 若目标清单的声明包含 redacted_assertions 字段，则需检查每个被编辑断言的 JUMBF URI。
    a. 若该编辑断言源自目标清单，则以 assertion.selfRedacted 失败代码拒绝该声明。
    失败代码拒绝该声明。
    b. 否则，将被编辑断言的JUMBF URI追加到所有被编辑断言的列表中。
6. 若目标清单的声明包含成分声明：
    a. 针对每个成分断言：
        i. 尝试解析成分声明的哈希URI。若URI无法解析、哈希值不匹配，或声明的JUMBF内容框仅含零值，则跳至下一个成分声明。
        ii. 若成分声明包含 activeManifest 字段（或 v1/v2 版本中的 c2pa_manifest 字段）：
            A. 将包含以下值的元组追加至所有成分清单列表：
                - 组件断言中 activeManifest（或 c2pa_manifest）字段的 hashed_uri 值
                - 成分断言中 claimSignature 字段的 hashed_uri 值
            B. 将刚附加的组件清单设为目标清单，并从"开始递归"步骤重复上述流程。
        iii. 若成分断言不包含 activeManifest（或 c2pa_manifest）字段，则记录 ingredient.unknownProvenance 信息代码（除非 relationship 字段值为 inputTo），随后跳转至下一个成分断言直至遍历完毕。此时从当前递归层级返回。
7. 若目标清单的声明不包含成分断言，则从当前递归层级返回。
8. 结束递归。

在汇编了所有成分清单和所有被删节的声明清单后，验证者应执行

以下验证算法：

1. 针对所有成分清单中的每份清单：
    a. 从每组元组中提取成分清单JUMBF URI对应的清单标签
    b. 在所有被删减的断言列表中搜索与清单标签匹配的断言
    c. 若发现一个或多个匹配的编辑断言：
        i. 使用第15.11.3.3.1节“声明签名哈希验证方法”中描述的方法验证该成分。
    d. 若未找到匹配的编辑后断言：
        i. 使用清单哈希验证方法（详见第15.11.3.3.2节“清单哈希验证方法”）或声明签名哈希验证方法（详见第15.11.3.3.1节“声明签名哈希验证方法”）验证该成分。
    e. 若组件断言包含验证结果字段：
        i. 对于验证结果字段值中的每项条目，若验证过程中未返回等效条目，则将其作为验证结果的一部分返回。
        ii. 若验证过程中返回的条目中存在未出现在 `validationResults`字段中，则将其作为验证结果的一部分返回。
    f. 如果未存在validationResults字段，且成分断言是v3版本的成分断言且包含 包含 `activeManifest` 字段，则返回失败代码 `assertion.ingredient.malformed`。

验证器应忽略C2PA清单存储中出现的任何额外C2PA清单，只要这些清单未包含在成分清单列表中。

注意 忽略额外C2PA清单可确保兼容自定义断言及未来可能以验证器无法识别的形式引用C2PA清单的构造。

15.11.3.3.1. 声明签名哈希验证方法

该方法包含对声明组件声明的完整验证（与对活动清单的验证类似），但不评估内容绑定：

1. 解析声明签名字段中`url`值的URI引用以获取成分声明签名框。若URI引用无法解析或声明签名字段缺失，则以失败代码`ingredient.claimSignature.missing`拒绝该成分声明。
2. 通过遵循第15.4节“确定哈希算法”中的流程，确定哈希算法标识符（或可能的失败代码）。
3. 使用该算法及第8.4.2.3节“哈希JUMBF框”中描述的步骤，计算成分声明签名框的哈希值。
4. 将计算出的哈希值与哈希字段中的值进行比对。

a. 若哈希值不匹配或哈希字段缺失：

   i. 以失败代码 `ingredient.claimSignature.mismatch` 拒绝该声明。

b. 若哈希值相等，则返回验证成功码 `ingredient.claimSignature.validated`。

   i. 根据第15.7节"验证签名"、第15.8节"验证时间戳"及第15.9节"验证凭证撤销信息"验证声明签名、时间戳及凭证撤销信息。

   ii. 对于清单中每个具有匹配清单标签的编辑断言URI，若所引用的断言存在，且其中任何JUMBF内容框或填充框包含除零个或多个`0x00`字节以外的内容，则应以断言未编辑（`assertion.notRedacted`）失败代码拒绝该声明。

   iii. 除硬绑定断言（其无法用于验证成分）外，应按第15.10节"验证断言"对每个未编辑断言进行验证。

采用声明签名哈希验证方法时，验证器不得为 `activeManifest` 字段记录哈希不匹配失败代码。

注 这是因为如果编辑操作影响了引用的清单文件，该字段的哈希值可能无法匹配。

15.11.3.3.2. 清单哈希验证方法

若当前验证器信任先前声明生成器的验证结果，则可更快验证因编辑而未变更的成分清单：

1. 解析 `activeManifest` 字段 `url` 值中的 URI 引用以获取成分清单框。若 `url` 字段缺失或 URI 引用无法解析，则以 `ingredient.manifest.missing` 失败代码拒绝该成分声明。
2. 参照第15.4节"确定哈希算法"中的流程确定哈希算法标识符（或可能的失败代码）。
3. 使用该算法及第8.4.2.3节"JUMBF框哈希处理"所述流程，计算成分清单框的哈希值。
4. 将计算出的哈希值与哈希字段中的值进行比较。
   a. 若哈希值不匹配或哈希字段缺失：
      i. 以 `ingredient.manifest.mismatch` 失败代码拒绝该声明。
   b. 若哈希值相等，则该成分完全通过验证，并返回 `ingredient.manifest.validated` 成功代码。

15.12. 验证资源内容

若活动清单为标准清单，则应使用其中硬性绑定进行验证。若活动清单为更新清单，则需在parentOf组件的更新清单中查找硬性绑定

清单中查找硬绑定；若该清单同样是更新清单，则需沿parentOf组件链追溯至首个标准清单。若未找到标准清单，或标准清单不存在硬绑定，则应以失败代码`claim.hardBindings.missing`拒绝活动清单的声明。

资产也可由多个部分组成，每个部分均关联独立哈希值（参见第18.9节"多资产哈希"），可分别进行验证。例如，资产可包含独立的静态图像与视频部分，两者均可单独验证。

15.12.1. 数据哈希验证

15.12.1.1. 通用规范

一旦定位到标准清单（及其绑定），则应从
`c2pa.hash.`data断言中提取排除范围。

若某个排除区间的结束字节偏移量（起始偏移量 + 长度）大于数组中下一个排除区间的起始字节偏移量，或起始值/长度值为负值，则清单应被拒绝并返回断言失败代码 `assertion.dataHash.malformed`。

若遇到更新清单，则需将起始值为整个C2PA清单存储开头偏移量的排除区段长度值，视为当前整个C2PA清单存储的长度（含文件格式特有的附加信息）。

该 `c2pa.hash.data` 中指定的哈希算法 (`alg`) 应作用于资产字节序列计算，排除范围内的字节除外。若排除范围的结束位置超出资产末尾，则清单应被拒绝，并返回 `assertion.dataHash.mismatch` 失败代码。

如果alg字段中指定的哈希算法未出现在第13.1节"哈希"中的允许或弃用列表中，则清单应以算法不支持（`algorithm.unsupported`）失败代码被拒绝。如果缺少hash字段，则清单应以数据哈希断言不匹配（`assertion.dataHash.mismatch`）失败代码被拒绝。

排除范围与填充值的组合（特别是支持多遍处理工作流所需的填充）可能使攻击者得以用任意数据替换部分填充内容，从而影响资产使用而不导致哈希值失效。因此验证者应确保排除范围内的数据（包括C2PA清单存储）仅包含C2PA清单存储本身及明确标记的填充字段或自由/跳过框中的适当填充数据（如零填充数据）。在上述C2PA清单存储之外的其他排除范围内，资产元数据的全部或部分内容亦可按第9.2.5节"资产元数据绑定"所述纳入。若验证器发现任何超出上述许可范围的数据，则应以`assertion.dataHash.mismatch`失败代码拒绝该清单。若验证器发现除C2PA清单存储外存在其他排除范围，且在明确标记的填充字段或自由/跳过框中存在不适当的填充（如零值数据），则应设置信息代码`assertion.dataHash.additionalExclusionsPresent`。

若未发现错误情况，验证器应添加成功代码 `assertion.dataHash.match`。
最终返回的列表中。

若对资产全部数据（扣除任何排除范围）计算的哈希值与 `c2pa.hash.data` 字段中的哈希值不匹配，则验证者应检查是否存在多资产哈希断言。若存在该断言，则按第15.12.4节"验证多资产哈希"所述进行验证；若不存在，则应以断言数据哈希不匹配（`assertion.dataHash.mismatch`）失败代码拒绝清单。

15.12.1.2. JPEG 1文件的哈希处理

在JPEG 1文件中，上述文件格式附加信息将包含所有APP11标记及其对应的APP11分段长度字节。由于分段长度位于排除范围内，验证器应将排除范围的总长度与代表C2PA清单的所有APP11分段总长度进行比对，以确保长度未被篡改。

注 JPEG 1文件可能因C2PA以外的原因包含APP11分段（例如JPEG 360或JPEG隐私与安全功能），此类分段不计入上述计算。

15.12.2. 验证BMFF哈希值

对于呈现给用户的资产任何部分（包括但不限于音频、视频或文本），应根据第9.2节“硬绑定”对渲染内容对应的硬绑定进行验证。若标准硬绑定验证失败且存在多资产哈希断言，则应按[validating_a_multi_asset_hash]所述进行验证。若内容在任何验证环节失败，验证器应明确向用户提示部分内容与声明不符，并在可能时标明具体未通过验证的内容部分。若存在内容绑定关系却有内容缺失，该缺失情况同样构成验证失败。即使后续内容部分验证成功，验证器仍应持续报告验证失败。

对于在渲染开始前尚未完全可用的内容（例如自适应比特率流媒体传输（ABR）和渐进式下载过程中），尚未可用的内容部分缺失不视为验证失败。随着内容逐步可用，验证器应在渲染前对每个内容片段进行验证，具体流程如前所述。此外，验证器应验证内容序列与清单生成时完全一致。除非播放器已明确向验证器发出预期中断信号（例如用户通过界面手动跳转操作），否则当序列不匹配时，验证器应向用户清晰提示意外中断已发生。这包括验证给定默克尔树的位置值是否从零开始，且后续每个数据块递增一位；等效而言，位置值始终指示当前渲染的数据块。

对于需通过渐进式下载在播放过程中验证的内容，默克尔树的叶节点可与`'mdat'`中视频轨道的同步点（如RAP点随机访问点）对齐。当设置'variableBlockSizes'以实现此类对齐时，线性播放期间的验证或跳转至目标播放时间的操作均可通过同一序列实现。目标区块需被获取、验证，并从中选取轨道进行渲染。

对于故意偏离声明生成器原始渲染意图的内容（如快进、倒带或变速播放），验证器可能无法完成内容验证。此时，

验证器应向用户明确提示，在相应操作期间无法对内容进行验证。

对于包含C2PA内容溯源框且box_purpose属性设置为更新状态的内容，系统将首先在box_purpose属性为更新的C2PA内容溯源框中搜索活动清单，随后在box_purpose属性为原始的C2PA内容溯源框中进行搜索。若活动清单位于box_purpose设为更新的C2PA内容溯源框中，则追溯组件父链（根据需要在box_purpose设为更新或原始的C2PA内容溯源框中查找），直至找到首个非更新清单。该清单内容的BMFF哈希值应依据第9.2节"硬绑定"进行验证。添加box_purpose设为update的C2PA内容来源框不应影响哈希计算，因其添加于文件末尾且未改变任何偏移量。

如果 bmff-hash-map 不包含 exclusions 字段，或者该字段的值不是包含至少一个条目的数组类型，则清单应被拒绝，并返回断言失败代码 assertion.bmffHash.malformed。

通过遵循第15.4节"确定哈希算法"中的流程，确定哈希算法标识符（或可能的失败代码）。

若某子集区间的结束字节偏移量（偏移量 + 长度）大于数组中下一个区间的偏移量值，或偏移量/长度值为负值，则清单应被拒绝并返回失败代码 assertion.bmffHash.malformed。本节所述其他所有失败情况均使用失败代码 assertion.bmffHash.mismatch。否则，验证器应将成功代码 assertion.bmffHash.match 添加至最终返回的列表中。

如果 BMFF 哈希过程产生 assertion.bmffHash.mismatch 失败代码，则验证器应检查是否存在多资产哈希断言。若存在此类声明，则不应发出 assertion.bmffHash.mismatch 失败代码，而应按第 15.12.4 节 " 验证多资产哈希 " 所述验证多资产哈希声明；否则应以 assertion.bmffHash.mismatch失败代码拒绝清单。

15.12.2.1. 使用默克尔树的非碎片化资产

如果bmff-hash-map中存在merkle字段，验证者应验证Merkle树。若bmff-merkle-map中既未包含fixedBlockSize也未包含variableBlockSizes，则将mdat的整个有效负载视为单个叶节点进行哈希计算。若存在fixedBlockSize且不存在variableBlockSizes，则将mdat有效负载划分为固定长度区块，每个区块作为叶节点处理。若最终区块超出mdat有效负载末尾，则应将末尾区块大小调整为仅延伸至mdat有效负载末尾。若存在 variableBlockSize 且固定块大小未定义，则将 mdat 有效负载按 variableBlockSizes 数组定义的大小划分。若元素个数不等于 count 或数值总和不等于 mdat 有效负载大小，则清单应被拒绝并返回 assertion.bmffHash.malformed 失败代码。若bmff-merkle-map中同时存在fixedBlockSize和variableBlockSizes，则清单应被拒绝并返回断言失败代码assertion.bmffHash.malformed。

若 bmff-merkle-map 中计数等于该映射中哈希值元素数量
如果叶节点的哈希值与bmff-merkle-map中哈希值的元素不匹配，则清单

应以断言失败代码 assertion.bmffHash.mismatch 予以拒绝。若 bmff-merkle-map 中计数值小于该映射中哈希值元素数量，且辅助 uuid C2PA 盒子不存在（如 A.5.4 节所述），则清单应被拒绝，失败代码为 assertion.bmffHash.malformed。若辅助uuid C2PA 框与叶节点计算的哈希值与 bmff-merkle-map 中哈希元素不匹配，则清单应被拒绝，失败代码为 assertion.bmffHash.mismatch。若 bmff-merkle-map 中计数大于哈希值元素数量，则清单应被拒绝并返回失败代码 assertion.bmffHash.malformed。

15.12.2.2. 使用默克尔树的碎片化资产

如果bmff-hash-map中存在Merkle字段，验证者应验证Merkle树。若辅助uuid C2PA框不存在（如A.5.4节"适用于大型及碎片化文件的辅助'c2pa'框"所述），则应以断言失败代码assertion.bmffHash.malformed拒绝该清单。若辅助uuid C2PA 框与叶节点计算的哈希值与bmff-merkle-map中哈希元素 不匹配（而非 等于），则 此时 该 清单 应 被 拒绝 并 返回 错误代码 assertion.bmffHash.mismatch

15.12.3. 验证通用盒子哈希值

一旦定位到标准清单（及其绑定），则应从存储在 c2pa.hash.boxes 断言中的箱映射结构的 boxes 字段中提取待验证的箱列表。若不存在此字段，则应以断言失败代码 assertion.boxesHash.malformed 拒绝该清单。

资产中的盒子应按其在boxes数组中的顺序出现，包括包含C2PA清单的盒子。若资产中存在其他盒子，则清单应被拒绝并返回失败代码 assertion.boxesHash.unknownBox。若盒子出现顺序错误，则清单应被拒绝并返回失败代码assertion.boxesHash.mismatch。

若任何箱体的哈希值不匹配，且该箱体未包含值为true的excluded字段，则清单应以失败代码assertion.boxesHash.mismatch被拒绝。否则，验证器应将成功代码assertion.boxesHash.match添加至最终返回列表。

若任何alg字段指定的哈希算法未出现在第13.1节"哈希"中的允许或弃用列表中，或alg字段未出现在箱映射或任何特定箱哈希映射中，则清单应被拒绝并返回失败代码algorithm.unsupported。

如果 boxes 数组中的任何箱子哈希映射未包含 names 字段，则清单应被拒绝，并返回断言失败代码 assertion.boxesHash.malformed。

对于名称数组和箱子数组中列出的每个箱子，应针对箱子字节（连同任何关联头部）计算指定的哈希算法。若名称数组包含多个条目，则该范围内的箱子哈希值应从首个箱子起始位置计算至最后一个箱子结束位置，期间包含箱子间可能存在的任意字节。

若哈希字段缺失，或任何计算结果与该批次哈希字段值不符，则清单应以assertion.boxesHash.mismatch失败代码被拒绝。当箱体哈希处理产生assertion.boxesHash.mismatch失败代码时，验证器应检查是否存在多资产哈希断言。若存在该断言，则按第15.12.4节"多资产哈希验证"所述进行验证；若不存在，则清单应被拒绝并返回断言失败代码assertion.boxesHash.mismatch。

15.12.3.1. JPEG特殊处理

在验证JPEG文件时，验证器应检查每个标识为特殊C2PA框标识符的框是否确实包含部分或全部C2PA清单存储的APP11。C2PA清单存储通过以下特征识别：其为JUMBF超级框，标签为c2pa，且JUMBF类型UUID为63327061-0011-0010-8000-00AA00389B71（详见第11.1.4.2节"清单存储"）。

若存在未包含在哈希盒列表中的APP11且该APP11不属于C2PA清单存储库，则清单应被拒绝并返回失败代码assertion.boxesHash.unknownBox。

15.12.3.2. 字体特殊处理

在验证字体时，验证器应检查字体的C2PA表对应的框是否存在，并确定该框是否包含嵌入式清单、远程清单URI或两者兼有。

若存在未被任何框覆盖的字体表，则应以断言失败代码 assertion.boxesHash.unknownBox 拒绝该清单。

15.12.4. 多资产哈希验证

若资产硬链接的标准验证失败，且该资产包含多资产哈希断言，验证器应继续验证该断言。若存在多个多资产哈希断言，则清单应被拒绝，失败代码为 assertion.multiAssetHash.malformed。

多资产哈希断言（c2pa.hash.multi-asset）的验证应通过遍历多资产哈希映射中的部件数组来执行。若部件字段缺失，或存在但值为空数组，则清单应被拒绝，并返回断言错误代码 assertion.multiAssetHash.malformed。

对于每个部件，验证器应确保其同时包含有效的定位器和有效的哈希断言字段。若任一缺失，则清单应被拒绝，并返回断言错误代码assertion.multiAssetHash.malformed。

若定位器为字节偏移定位器，验证器应确保字节偏移和长度字段存在、非负且不超过资源总长度。若任一缺失、为负值或过大，则清单应被拒绝并返回失败代码 assertion.multiAssetHash.malformed。

若定位器由bmffBox表示，则验证器应确保指定的盒子存在于资产中

若 盒 不 存在， 则 清单 应 被 拒绝 并返回 失败代码
断言.多资产哈希.格式错误.

给定有效的定位器和哈希值后，验证器应尝试使用定位器信息定位该部件。若该部件不存在，且可选字段不存在或其值为`false`，则应以断言失败代码`assertion.multiAssetHash.missingPart`拒绝该清单。若可选字段存在且值为`true`，则验证器应跳过该部件并继续处理下一个部件。

注意	丢弃某些部件可能导致验证器无法明确识别 。多数情况下，仅能有效舍弃文件末尾的一个或多个部件，而非中间任意部件。

若定位到的部分存在重叠，或其总和未能覆盖资产的每个字节，则清单应被拒绝，并返回失败代码 `assertion.multiAssetHash.malformed`。

对于每个定位到的部分，验证器应使用指定算法及方法（即数据哈希、通用盒哈希或BMFF哈希）对该部分的字节计算哈希值。若计算结果与`hashAssertion`字段引用的硬绑定断言值不匹配，则清单应被拒绝并返回失败代码`assertion.multiAssetHash.mismatch`。

若所有定位部件的哈希断言均验证成功，验证器应记录成功代码 `assertion.multiAssetHash.`match，且不得记录与该资产硬绑定相关的任何失败代码。

15.12.5. 集合数据哈希验证

15.12.5.1. 通用规范

对标准清单中定位到的集合数据哈希断言（`c2pa.hash.collection.data`）的验证，应通过遍历集合数据哈希映射中的`URI`数组来执行。若未包含uris字段，则清单应以断言失败代码`assertion.collectionHash.malformed`被拒绝。

具体使用的哈希算法应根据alg字段的值确定，并按第15.4.3节"算法验证"规定进行处理。若`alg`字段缺失，则清单应被拒绝，并返回错误代码`assertion.collectionHash.malformed`。

对于 uris 数组中的每个 `uri-hashed-data-map`，验证器应确保其同时包含 `uri` 和 `hash` 字段。若任一字段缺失，则应以`assertion.collectionHash.malformed` 失败代码拒绝该清单。

为规避潜在安全风险，验证器应在使用前验证URI（即uri字段值），确保URI中不包含"."或".."字符。若发现URI中存在上述任一字符，则应以`assertion.collectionHash.invalidURI`失败代码拒绝该清单。

对于从URI检索到的资源，其哈希值应使用指定算法基于数据的所有字节进行计算。若计算结果与哈希字段值不匹配，则清单应被拒绝并返回失败代码assertion.collectionHash.mismatch。否则，验证器应将成功代码assertion.collectionHash.match添加至最终返回的列表中。

若验证器未能找到清单数据哈希断言中列出的任何文件，则清单应被拒绝，并返回失败代码assertion.collectionHash.incorrectFileCount。

15.12.5.2. ZIP文件扩展规则

在关联C2PA清单的ZIP文件中，集合数据哈希包含额外的zip_central_directory_hash字段。如前所述，该字段包含ZIP中央目录中每个"中央目录头"的哈希值，以及"中央目录结束记录"（即ZIP文件的最后部分）。该字段使用的哈希算法与c2pa.hash.collection.data断言中的hash字段相同。

验证ZIP文件时，验证器应检查zip_central_directory_hash字段是否存在，并确认ZIP中央目录与"中央目录结束记录"的哈希值与其声明值匹配。若哈希值不匹配，则应以assertion.collectionHash.mismatch失败代码拒绝该清单。

第16章 用户体验

16.1. 方法论

C2PA旨在为实现溯源功能的用户体验（UX）实施者提供明确的建议与指导。这些建议的制定是一个持续进行的过程，涉及多元利益相关方，其成果在统一性与熟悉度之间取得平衡，同时兼顾用户在不同场景、平台和设备中的实用性与灵活性。相关建议详见《用户体验指导文件》。

16.2. 原则

用户体验建议旨在为向消费者呈现C2PA溯源信息制定最佳实践。这些建议致力于描述标准且易于识别的体验模式，具体包括：

- 为资产创建者提供捕获内容信息与历史的途径，并
- 向资产消费者提供其正在体验的内容信息及历史背景，从而使其能够理解内容来源并决定信任程度。

用于消费C2PA溯源信息的用户界面设计应基于资产的上下文环境。我们研究了4类主要用户群体及其在各类场景中接触C2PA资产的情况。这些用户群体在《C2PA指导原则》中被定义为消费者、创作者、发布者和验证者（或调查者）。为满足各群体在常见场景中的需求，我们针对多种典型案例提供了示范性用户界面方案。这些方案属于建议而非强制要求，我们期待最佳实践能持续演进。

16.3. 披露层级

鉴于完整C2PA数据集可能令用户难以消化，我们提出四级渐进式披露机制作为设计指引：

- 第一级：提示存在C2PA数据及其加密验证状态。
- 第二层级：针对特定资产的C2PA数据摘要。该层级应提供足够的信息，使消费者能够充分理解该资产如何形成当前状态，这些信息需针对特定内容、用户及使用场景进行定制。
- 三级：所有相关溯源数据的详细展示。需注意某些项目的相关性具有情境性，由用户体验实施者决定。
- 第4级：针对复杂取证调查场景，建议采用能揭示所有签名与信任信号细微细节的专用工具。

16.4. 公开评审、反馈与迭代

制定用户体验建议的团队深知其局限性与潜在偏见，并认识到反馈、评审、用户测试及持续演进是成功的关键要素。因此本建议将作为动态文档，持续吸纳C2PA用户体验在各类应用场景中的实际部署经验。

第17章 信息安全

17.1. 威胁与安全考量

本节概述了C2PA核心规范中所述技术的信息安全考量与流程。更详细的内容将在未来发布的C2PA材料（包括指导文件）中提供。

17.1.1. 背景

信息安全是C2PA的核心关注点。C2PA为规范制定了威胁模型与安全考量框架，该工作与C2PA内部其他安全相关项目形成互补。相关文档正在开发中，可查阅《安全考量》获取最新进展。

C2PA正在制定的安全考量文档包含：

- C2PA技术相关安全功能概要
- C2PA技术实际应用的安全考量
- C2PA技术面临的威胁及其应对措施，包括反制手段

17.1.2. 威胁建模流程概述

C2PA在系统开发过程中将安全性融入设计，同时预期随着系统、生态系统及威胁态势的演进，安全设计与威胁建模工作将持续推进。

为此，C2PA采用聚焦式威胁建模流程，以支持构建强健的安全与隐私设计。该流程不仅直接产出明确的威胁清单及安全考量文档，更在整个设计过程中持续强化安全思维。

威胁建模流程融合了同步（实时）威胁建模会议与异步内容开发。每次同步会议的参与人数控制在较小规模以促进高效讨论，但C2PA所有成员均可通过任一模式参与其中。

与其他安全活动类似，我们期望威胁建模流程能随C2PA生态系统共同演进。流程文档仅作为指导性文件，而非对C2PA内部威胁建模操作的严格指令。

17.1.2.1. 参考文献

为C2PA的威胁建模及相关安全活动提供参考依据时，采用了多种参考资料和实践经验。本节提供部分公开文档供参考。

- IETF关于安全考虑事项

- IETF隐私考量指南
- W3C安全与隐私自我评估问卷
- OAuth2威胁模型（示例）
- 威胁建模：安全设计
- OWASP 威胁建模
- 微软威胁建模

17.2. 危害、误用与滥用

17.2.1. 引言

C2PA指导原则规定，C2PA规范在制定过程中须以审慎态度评估框架被滥用或误用的风险，以防止造成意外危害、侵犯人权或对全球弱势群体造成不成比例的风险。

为确保C2PA符合该原则要求，危害、误用与滥用评估旨在识别并解决规范制定阶段及后续实施过程中可能出现的潜在问题。

此外，正在对规范进行审查以：

- 预判并缓解潜在滥用风险；
- 解决用户普遍关切的隐私问题；以及
- 兼顾全球用户及利益相关方的需求。

17.2.2. 考量因素

危害、滥用及误用评估是一个持续进行的过程。危害建模文档中呈现的信息不应被视为全面评估的最终结果，而应作为持续讨论的基础——这些讨论应以受影响社区为中心，旨在减轻潜在滥用风险、防范误用行为并保护人权。

该方法有两个关键方面：

持续性

危害、误用和滥用评估必须贯穿C2PA的设计开发、实施应用全过程，通过持续指导规范制定流程、实施与用户体验指南、意识提升工作、联盟治理机制，以及推动多元化C2PA生态系统的多边合作，以满足全球多样化应用场景需求。

多学科与多元性

危害、滥用及误用评估应作为协作性工作，汇聚跨学科专家及广泛利益相关方——这些参与者需具备来自不同地理区域、文化背景及个人身份的实践经验、技术专长及亲身经历。

17.2.3. 评估

危害建模的核心在于分析社会技术系统如何可能对用户、其他利益相关方或更广泛的社会群体产生负面影响，或如何制造、强化不公正结构，威胁人权，以及给全球弱势群体带来不成比例的风险。危害建模过程需系统性地整合三方面要素：系统架构及其用户可操作性的知识体系；现有类似系统对不同社会群体影响的历史性与情境性证据；以及与可能受系统影响的多元社区开展的参与式协商。这些综合信息构成了预判危害并主动制定应对策略的能力框架。

危害建模文档阐述了框架及迄今实施的流程，随后介绍了方法论、评估概述、公众审查与反馈纲要，以及为配合本规范1.0版及其实施与演进而制定的尽职调查措施。

17.2.4. 尽职调查行动

危害、误用及滥用评估已为C2PA技术规范及其配套文档的制定提供依据，并将持续发挥指导作用：

- 实施者指南
- 用户体验指南
- 安全考量
- 说明指南

此外，危害、滥用及误用评估应指导联盟治理机制，并为推动多元化C2PA生态系统提供多边合作方向。该生态系统旨在优化C2PA规范制定初衷所追求的核心效益：提升媒体可信度、强化用户控制权及增强透明度。

第18章. C2PA标准断言

18.1. 引言

本节列出了C2PA实现所用的标准断言集，描述其语法、用法等。为简化说明，所有示例JUMBF URI均已缩短——实际数据中必须使用完整URI。

所有断言均须遵循第6.2节"标签"所述规范设置标签，并按第5章*版本控制*。

所有C2PA标准化断言均采用ISO 19566-5:2023标准中的JSON JUMBF内容类型、CBOR JUMBF内容类型或嵌入文件内容类型。实体特定声明可采用上述任一类型，也可采用ISO 19566-5:2023附录B中的其他JUMBF内容类型（如XML），或根据ISO 19566-5:2023表B.1中的说明创建自有类型。Codestream内容类型不得用于C2PA声明。

除非另有说明，本标准断言集中记录的所有断言均应采用CBOR格式序列化。所有采用CBOR格式序列化的断言均须符合CBOR核心确定性编码要求（参见RFC 8949第4.2.1节），其模式定义须使用CDDL定义进行定义。

注	所有CDDL均视为非规范性内容。

对于使用JSON定义的区域，其模式应采用最新版本的JSON Schema进行定义。

18.2. 感兴趣区域

18.2.1. 描述

在某些使用场景中，特定断言（如动作断言）可能仅与资产的特定区域相关，而非整个资产。此时需要一种方法来描述该区域——无论是时间、空间、文本还是它们的组合。区域定义正是为此目的而存在。

18.2.2. 常用

区域定义中最关键的部分是范围字段，用于描述区域的时间范围、空间范围、帧范围、文本范围或这些范围的组合。

注释	虽然规范允许指定范围组合，但未定义清单文件如何处理此类组合。 消费者将使用这些功能。预计C2PA用户体验工作组将在未来对此进行探讨。

区域还可能包含一个或多个通用字段：

名称

用于表示区域名称的自由文本字符串，该名称可供用户界面使用。

标识符

表示区域专属机器可读标识符的自由文本字符串。

type

来自受控词汇表（如https://cv.iptc.org/newscodes/imageregiontype/）的值，或实体特定值（如`com.litware.newType`），用于表示区域所描绘的事物类型。

描述

自由文本字符串。

本规范旧版本包含角色字段。该字段现已弃用，生成感兴趣区域时不再包含此字段。

18.2.2.1. 范围

所有范围均包含类型字段，其值为"空间"、"时间"、"帧"、"文本"或"已识别"。此外，必须包含以下字段之一，其数据为包含该范围具体数据的对象：

- 形状（空间范围）；
- `time`（时间范围）；
- 帧（用于时间或文本）；
- `text`（文本范围）；
- `item`（用于特定标识项）。

18.2.2.2. 空间

空间范围通过形状对象描述。形状可用于表示矩形、圆形或多边形，其模型基于IPTC的区域边界结构。

18.2.2.3. 时间

时间范围通过时间对象描述，该对象表示从起始时间到结束时间的区间。时间描述可采用RFC 2326定义的标准播放时间（`npt`）（如W3C媒体片段规范所推荐），或采用RFC 3339定义的ISO 8601互联网配置文件中的"墙钟时间"。

注 "墙钟时间"适用于媒体资源记录特定日期时段内活动的情境，例如新闻播报或现场活动。

若未提供类型字段，则默认范围采用npt格式。若未提供起始字段，则范围应从资产开头开始。若未提供结束字段，则范围应至资产结尾结束。若两者均未提供，则该范围应代表整个资产。

，则该范围代表整个资产。

18.2.2.4. 帧

帧对象通过起始帧和结束帧（或页面，含首尾帧）定义范围。若未指定起始帧，则范围从资源开头开始；若未指定结束帧，则范围至资源结尾结束；若两者均未指定，则范围代表整个资源。

帧以单个序数表示，其中0表示第一帧。

虽然框架通常用于表示文档（如PDF）的页码，但在动画、视频和音频等其他媒体类型中也可能有其用途。建议在处理随时间变化的感兴趣区域时，尽可能使用时间范围替代框架。

18.2.2.5. 文本型

文本对象通过一个或多个URL片段标识符定义范围，其定义遵循W3C网页注释片段选择器规范。也可通过起始字符和结束字符（含边界）的偏移量精确限定范围。若未指定起始点，范围从片段开头开始；若未指定结束点，范围至片段结尾结束；若两者均未指定，则范围代表整个片段。

当单独使用时，文本选择器映射的片段条目代表指定文本范围的全部内容。然而，文本选择器范围映射支持一对文本选择器映射对象。`selector`的值表示范围的起始点（若未指定结束条目则代表整个范围），而`end`的值（若存在）则表示连续范围的终点。此外，可使用多个配对条目表示非连续范围。

18.2.2.6. 标识项

项目对象定义资产中的媒体轨道、媒体项目或其他离散内容项，使声明生成器能够标注仅适用于资产文件容器所承载内容子集的断言。例如，可用于标注视频文件中仅音频轨道相关。

媒体或内容项通过标识符字符串进行识别，其值应符合特定容器格式中典型的项目标识命名方案。例如，MP4文件的标识符值应为`track_id`，HEIF文件则应为`item_ID`。该标识符值随后需填入`value`字段。例如，在MP4视频文件容器中，当标识符为`track_id`且值为2时，表示该断言与文件中的第二个媒体轨道相关（可能是音频轨道）。

标识范围的另一用途是通过已知语义值标注特定区域。例如，基础解剖模型可用于标识人体特定区域。此类情况下，标识符应为定位该模式的URL或URI（但不必直接指向机器可读版本）。

18.2.3. 模式

此类型的模式由以下CDDL定义中的区域映射规则定义：

```
区域映射 = {
      "区域":               [1* $范围映射],                  ; 范围定义，一个或多个范围
      ? "name": tstr .size (1..max-tstr-length), ; 表示区域人机可读名称的自由文本字符串，可在用户界面中使用。
      ? "标识符": tstr .size (1..max-tstr-length), ; 表示该区域机器可读标识符的自由文本字符串，在本断言中唯一。
      ? "type":       tstr .size (1..max-tstr-length),              ; 取自受控列表
      ? "角色":        $role-choice,        ; 已弃用
      ? "描述": tstr .size (1..max-tstr-length), ; 区域的人类可读描述
      ? "元数据": $断言元数据映射, ; 断言的附加信息
}

$range-choice /= "spatial"                ; 通过物理区域标识的范围
$range-choice /= "temporal"               ; 由时间段标识的范围
$范围选择 /= "帧"                           ; 由一系列帧或页面标识的范围
$range-choice /= "textual"                ; 通过文本范围标识的范围
$range-choice /= "标识范围"                 ; 由特定标识符和值标识的范围

range-map = {
      "type":         $range-choice,            ; 取值为"空间"、"时间"、"帧"、"文本"或"已识别"
        ? "shape":    $shape-map,               ; 空间范围形状的描述
        ? "时间":      $时间映射,                 ; 描述时间范围的时间边界
          ? "帧":      $frame-map,               ; 描述时间范围的帧边界
范围
      ? "文本":        $text-map,                ; 描述文本范围的边界
      ? "item":       $item-map,                ; 标识范围边界的描述
}

坐标映射 = {
      "x": 浮点数,          ; x轴坐标"y": float,  ; y轴坐标
}

$shape-choice /= "rectangle"              ; 矩形形状
$shape-choice /= "circle"                 ; 圆形形状
$shape-choice /= "polygon"                ; 多边形形状

$单位选择 /= "像素"                     ; 单位为像素
$unit-choice /= "percent"           ; 单位为总尺寸的百分比


shape-map = {
      "类型":          $形状选择,                       ; 取值为"矩形"、"圆形"或"多边形" "单位":   $单位选择,; 取值为
      "pixel" 或 "percent"
      "原点":          $坐标映射,                       ; 形状的起始/原点坐标。
      ? "width": 浮点数,                          ; 矩形宽度，圆形直径（多边形忽略此参数）
      ? "height": 浮点数                          ; 矩形的高度
      ? "inside" : 布尔值,                        ; 形状内部或外部，默认为 `true`
      ? "顶点数":           [1* $坐标映射]                  ; 多边形剩余点/顶点
}

; npt与utc起止时间采用不同正则表达式格式 time-map = npt-time-map / wall-clock-time-
map
```

```
npt-time-map = {
      ? "type":       "npt"; 若不存在，则默认采用"npt"时间映射
      ? "start": tstr .regexp "^(?:(?:([01]?\d|2[0-3]):)?([0-5]?\d):)?([0-5]?\d)(\.(\d{1,9}))?$",  ; 开
始时间（若不存在则为资产起始点）。
      ? "end":                        ; 结束时间（若不存在则为资源结尾）。
}

wall-clock-time-map = {
      "type":         "wallClock"; 用于"wall-clock"时间映射时必须存在
      ? "start": tstr .regexp "^(\d{4})-(\d{2})-(\d{2})T(\d{2}):(\d{2}):(\d{2})(\.\d+)?(([+-
]\d{2}:\d{2})|Z)$", ; 开始时间（若未提供则为资产起始时间）。
      ? "end":
]\d{2}:\d{2})|Z)$", ; 结束时间（或资产/直播边界结束位置，若不存在则省略）。
}

; 此结构可用于视频帧或文档页面 frame-map = {
      ? "start": int,           ; 起始帧（若不存在则为资源开头）。
      ? "结束":       整数        ; 结束帧（若不存在则为资源末尾）。
}

; 此模型参照W3C网页注释选择器模型 text-selector-map = {
      "片段":               tstr,      ; 片段标识符，遵循RFC3023或ISO 32000-2附录O规范
      ? "start":           整数,       ; 起始字符偏移量（若不存在则为片段开头位置）。
      ? "end":             整数        ; 结束字符偏移量（若不存在则为片段末尾）。
}

; 用于标识范围的一个或两个文本选择器映射文本选择器范围映射 = {
      "选择器":             $文本选择器映射,                    ; 起始（或唯一）文本选择器
      ? "结束":             $文本选择器映射                     ; 若存在，表示文本范围的结束
}

text-map = {
      "选择器": [1* $文本选择器范围映射] ; 文本范围数组（可能包含不连续的范围）
}

item-map = {
      "标识符":             tstr .size (1..max-tstr-length),              ; 用于标识项目的容器特定术语，例如MP4的"track_id"
或HEIF的"item_ID"
      "value":             tstr .size (1..max-tstr-length),              ; 标识符的值，例如标识符为"track_id"时值为"2"表示该
资源的第2条音轨
}

; 以下值已弃用
$role-choice /= "c2pa.areaOfInterest" ; 需标识的任意区域
$role-choice /= "c2pa.cropped"                    ; 此区域为裁剪操作后剩余部分
$role-choice /= "c2pa.edited"                     ; 此区域已应用编辑操作
角色选择 /= "c2pa.placed"                         ; 放置/添加配料的区域
$role-choice /= "c2pa.redacted"                   ; 此区域内容已被编辑
$role-choice /= "c2pa.subjectArea"                ; 特定于主体（人类或其他）的区域
$role-choice /= "c2pa.deleted"                    ; 部分信息已被移除/删除
$role-choice /= "c2pa.styled"                     ; 此区域已应用样式
$role-choice /= "c2pa.watermarked"                ; 为实现软绑定目的，在此区域添加了水印
```

18.2.4. 示例

以下是一系列采用CBOR诊断标记法（RFC 8949第8节）的示例：

```
// 视频中时空范围组合示例 //
{
   "区域": [
      {
         "type": "temporal","time":
         {
            "type": "npt",
            "start": "0",
            "end": "5.2"
         }
      },
      {
         "类型": "空间","形状": {
            "type": "矩形",
            "单位": "像素","原点": {
               "x": 10.0,
               "y": 10.0
            },
            "width": 200.0,
            "height": 112.0
         }
      }
   ],
   "name": "动画徽标","identifier": "logo-clip",
   "描述": "5.2秒的开场动画徽标，位于矩形区域内（距顶部和左侧10像素），尺寸为200像素×112像素"
}

// Word/DOCX文件中的文本范围示例 //
{
   "区域": [
      {
         "type": "textual", "text" :
         {
            "selectors" : [ [
                  {
                     "片段" : "xpointer(/w:document/w:body/w:p/)"
                  }
               ]
            ]
         }
      },
   ],
   "描述": "AI助手编辑的内容"
}

// 带标签PDF文件中的文本范围示例 //
{
   "区域": [
      {
         "type": "textual","text" :
         {
            "selectors" : [
```

```
            [
               {
                  "选择器" : {
                     "片段" : "路径=/文档/节[0]/页[3]","起始位置" : 10,
                     "end" : 20
                  }
               }
            ]
         ]
      }
   },
],
"描述": "根据《信息自由法》请求进行的删节处理"
}

// 未标记PDF文件中的文本范围示例 //
// 在这种情况下，我们只能指定页面和矩形区域 //
{
   "region": [
      {
         "type": "textual","text" :
         {
            "selectors" : [ [
                  {
                     "选择器" : {
                        "fragment" : "page=1,rect=10,10,450,500","start" : 10,
                        "end" : 20
                     }
                  }
            ]
         ]
      }
   },
],
"描述": "根据《信息自由法》请求进行的删节处理"
}

// 从PDF中删除某些页面的示例 //
{
   "区域": [
      {
         "type": "frame","frame" : {
            "start" : 27,
            "end" : 30
         }
      },
   ],
   "描述": "分发前已移除多余页面"
}

// Excel 中单元格区域示例 //
{
   "区域": [
      {
         "type": "textual","text" :
         {
            "选择器" : [ [
                  {
```

```
                  "选择器" : {
                     "fragment" : "xpointer(Sheet1!A5:A10)",
                  }
               }
            ],
            [
               {
                  "选择器" : {
                     "fragment" : "xpointer(Sheet1!B5:B10)",
                  }
               }
            ]
         ]
      }
   },
],
"描述": "对Excel中的一组单元格应用了样式"
}

// 连续表格单元格范围示例 //
{
"区域": [
   {
      "type": "文本型", "text" : {
         "选择器" : [ [
               {
                  "选择器" : {
                     "fragment" : "xpointer(//table[1]/tr[1]/td[2])",
                  },
                  "end" : {
                     "fragment" : "xpointer(//table[1]/tr[1]/td[4])",
                  }
               }
            ]
               ]
      }
   },
],
"描述": "清空了部分表格单元格"
}

// 视频中特定轨道的范围示例 //
{
"区域": [
   {
      "type": "temporal", "time":
      {
         "type": "npt",
         "start": "0",
         "end": "5.2"
      }
   },
   {
      "type": "已识别","item": {
         "标识符": "track_id", "值": "2"
      }
   }
],
"描述": "增强了部分音频轨道"
```

```
}


// 示例说明在整个资源中眼睛均被替换 //
{
   "区域" : [
      {
         "type" : "temporal",
         "time" : {},
      },
      {
         "type" : "identified","item" : {
            "标识符" : "https://bioportal.bioontology.org/ontologies/FMA","值" : "一双眼睛"
         },
      }
   ]
   "描述" : "确保他在整个会议期间看起来像是在睡觉"
}
```

18.3. 断言元数据

18.3.1. 描述

在许多情况下，提供断言的附加信息（如生成日期和时间或其他有助于清单消费者判断断言数据来源或真实性的数据）是很有用甚至必要的。

注	清单消费者无需读取断言元数据的任何部分，可自主选择读取哪些内容。 是否消费特定字段，甚至可根据所应用的断言类型动态调整消费策略。

下文展示了其他断言内部使用的基础模式。

CDDL中关于断言元数据映射规则的定义见CDDL断言元数据规范：

断言元数据的CDDL

```
;描述断言的附加信息，包括对其的哈希URI引用。此处采用套接字/插件机制，使哈希URI映射可在不同文件中使用，而无需在同一文件中定义映射
$assertion-metadata-map /= {
   ? "dateTime": tdate, ; 断言创建/生成时的RFC 3339日期时间字符串
   ? "reviewRatings": [1* rating-map], ; 该断言获得的评分（可能为空）
   ? "reference": $hashed-uri-map, ; 指向本评审所涉及的另一项断言的哈希URI引用
   ? "dataSource": source-map, ; 断言数据来源的描述，从预定义列表中选择
   ? "localizations" : [1* localization-data-entry] ; 断言中字符串的本地化版本
   ? "regionOfInterest" : $区域映射 ; 描述该断言相关的资产区域
```

```
}

$source-type /= "signer"
$source-type /= "claimGenerator.REE"
$source-type /= "claimGenerator.TEE"
$source-type /= "localProvider.REE"
$source-type /= "localProvider.TEE"
$source-type /= "remoteProvider.1stParty"
$source-type /= "remoteProvider.3rdParty"
$source-type /= "humanEntry"
; 以下两个源类型的值在2.0版本中已弃用
$source-type /= "humanEntry.anonymous"
$source-type /= "humanEntry.identified"


; 注意：本规范早期版本还包含"actors"字段，但在2.0版本中已移除。
source-map = {
   "type": $source-type,; 枚举列表中的值，用于标识断言来源：运行于丰富执行环境（REE）的声明生成器、运行于可信执行环境（TEE）的声明生成器、REE中的本地数据提供者（如移动操作系统的定位API）、TEE中的本地数据（如芯片供应商的可信定位可信应用）、服务器等远程数据提供者（如谷歌地理位置API服务），或人工输入。
   ? "details": tstr .size (1..max-tstr-length), ; 可读字符串，提供断言数据来源的详细信息，例如提供数据的远程服务器的URL
}

int-range = 1..5

$review-code /= "actions.unknownActionsPerformed"
$review-code /= "actions.missing"
$review-code /= "actions.possiblyMissing"
$review-code /= "深度图场景不匹配"
$review-code /= "ingredient.modified"
$review-code /= "配料.可能被修改"
$review-code /= "缩略图主图不匹配"

; 以下三个 review-code 值在 2.0 版本中已弃用
$review-code /= "stds.iptc.location.inaccurate"
$review-code /= "stds.schema-org.CreativeWork.misattributed"
$review-code /= "stds.schema-org.CreativeWork.missingAttribution"

rating-map = {
   "value": 整数范围, ; "该项目的评分值，范围为1（最差）至5（最佳）"
   ? "code": $review-code, ; 描述评分原因的标签格式字符串
   ? "explanation": tstr .size (1..max-tstr-length), ; 解释评分依据的人类可读字符串
}

; 用于存储本地化词典的数据结构
$localization-data-entry /= {
   * $$语言字符串
}

语言字符串 /= tstr .size (1..最大字符串长度)
```

CBOR诊断标记法示例（RFC 8949第8节）：

```
{
   "reference": {
         "url": "self#jumbf=c2pa.assertions/c2pa.metadata", "alg":
         "sha256",

      },
   "dataSource": {
      "type": "localProvider.REE",
      "details": "由操作系统地理位置API提供的EXIF GPS数据"
   }
}
```

在大多数情况下，此类断言特定元数据会直接出现在其他断言（例如成分）中，作为其元数据字段的值。但有时需要或更适合将断言元数据存储在独立的元数据断言中，例如当断言不采用JSON或CBOR格式时（如缩略图）。

该断言元数据断言的标签为 `c2pa.assertion.metadata`。

18.3.2. 数据源

此dataSource字段为可选字段，允许声明生成器向下游清单消费者告知断言内容的来源。若未为特定断言提供`dataSource`，则该断言的来源视为签名者。

注意 默认情况下，所有创建的断言均归属于签名者，因为信任模型根植于对签名者的信任——签名者通常也是声明生成者。

该字段的值为dataSource对象，包含两个字段：`type`和`details`。

数据源类型字段定义了数据源的类型。该字段采用第6.2节“标签”中描述的格式进行标签组合。其值可选取表5“数据源类型”中定义的规范值，或通过实体特定命名空间作为扩展机制实现。

表5. 数据源类型

类型值	含义
签名者	断言内容源自签名者
声明生成器.REE	断言内容来自在丰富执行环境（REE）中运行的声明生成器，例如桌面或移动操作系统
声明生成器.TEE	断言内容来自在可信执行环境（TEE）中运行的声明生成器，例如可信操作系统
本地提供者.REE	断言内容源自运行于同一物理计算设备上的REE中的数据源，该设备与声明生成器共用
类型值	含义

localProvider.TEE	断言内容来自与声明生成器位于同一物理计算设备上的 TEE 中运行的数据源
remoteProvider	断言内容来自由签名者或声明生成器供应商控制的远程数据源
remoteProvider.external	断言内容来自外部远程数据源，该数据源既非签名者也非声明生成器供应商
humanEntry	断言内容由人工输入

details 字段为可读字符串，提供关于数据源的补充信息，例如提供断言内容所使用的 API 名称，或内 容 来源服务器的 URL。例如，一个广泛的位置断言来源可能具有类型值 remoteProvider.3rdParty，其 details 值设置为www.googleapis.com/geolocation/v1/geolocate 。

18.3.3. 审核评分

当存在时，reviewRatings数组为声明生成器提供了一个位置，用于提供一个或多个关于断言质量（或质量不足）的评级对象。如果存在类型字段值为humanEntry.anonymous或humanEntry.credentialed的数据源对象，则不得存在reviewRatings。

评级对象的值字段必须存在，其整数值范围为*1*（最差）至*5*（最佳）。若存在说明字段，则应包含可供人类理解的评级类型描述字符串。此外，可选的机器可读代码字段用于定义断言特定的评估结果代码。代码字段的值采用第6.2节"标签"所述格式定义。其值可选取表6"代码字段值"中规范定义的值，或通过实体特定命名空间作为扩展机制实现。

表6. 代码 字段值

代码值	适用断言	含义
actions.unknownActionsPerformed	c2pa.actions	操作断言不包含在创作工具中执行的所有操作的完整列表（例如，由于使用了创作工具未知其效果的第三方过滤器）。
actions.placedIngredientNotFound	c2pa.actions	正在审查的操作断言包含一个未解析的成分URI的放置操作。值应为1。
ingredient.action 缺失	c2pa.ingredient	正在审查的成分断言在其声明中至少缺少一个引用它的操作。值应为 1。
ingredient.notVisible	c2pa.ingredient	正在审查的成分声明在绑定到该清单的数字内容中不可见。值应为 1。
代码值	**适用声明**	**含义**
深度图场景不匹配	c2pa.深度图.GDe pth	深度贴图断言的内容与资源中主呈现所描绘的场景不匹配（例如，由于画中画攻击）。

缩略图主图不匹配	c2pa.缩略图.内容	缩略图内容与资源中主呈现内容不匹配。

18.3.4. 引用

由于断言元数据断言的引用字段是标准的哈希URI，因此断言元数据断言也可引用活动清单之外的其他清单中的断言。例如，活动清单可能包含一个断言元数据断言，用于验证某个组件清单中存在的c2pa.metadata断言。

注 由于声明是断言的特殊类型，此方法同样适用于引用其他清单中的声明。

18.3.5. 日期时间

若存在dateTime字段，其值应为符合CBOR日期时间规范（RFC 8949, 3.4.1）的日期时间字符串。

18.3.6. 感兴趣区域

该声明可能仅针对资产的一部分——例如视频中的特定帧段或图像上的特定区域。此类区域可通过 regionOfInterest 字段进行标识，其值为区域映射对象（详见第 18.2 节"感兴趣区域"）。

18.3.7. 本地化

18.3.7.1. 通用要求

C2PA清单的消费者应尽可能以母语理解其中信息。为此，可在断言元数据中添加包含词典的本地化信息。

18.3.7.2. 本地化词典

本地化词典应由单一对象构成，其中每个键均采用JSON-LD的语言索引技术表示对应译文。若需翻译的值未关联顶级键，则应使用"点表示法"(.)引用对象内嵌的键。当需遍历数组中特定元素时，应采用数组索引表示法([n], n≥0)。当待译值本身为数组时，可引用特定元素。示例4"本地化字典示例"中展示了若干应用场景。

词典”中列举了若干示例：

示例4. 本地化词典示例

```
{
   "dc:title": {
      "en-US": "凯文的五只猫",
      "en-GB": "凯文勋爵的五只猫", "es-MX": "凯文的五只猫
      ", "es-ES": "凯文的五只猫", "fr": "凯文的五只猫",
      "jp": "ケヴィンの５匹の猫"
   }
}
```

```
{
   "actions[0].softwareAgent": { "en-US":
      "Joe's Photo Editor", "en-GB": "Joe's
      Photo Editor",
      "es": "Joe的照片编辑器", "fr": "乔的照片编辑器",
      "jp": "乔的照片编辑器"
   }
}
```

任何此类第三方键值均需遵循第6.2.1节"命名空间"中的命名规范进行命名空间标识，

例如 `com.litware`。为使清单消费者能显示这些键值的人类可读信息，声明生成器应通过此本地化方法提供字符串。

本地化操作通过在断言元数据中应用该机制，展示了其在定制操作本地化中的应用场景。

`c2pa.actions` 断言。

本地化操作

```
{
   "com.litware.blur": { "en-
      US": "模糊",
      "fr-FR": "模糊处理",
   },
   "com.litware.filter": {"en-
      US": "滤镜",
      "es-ES": "过滤",
      "jp-JP": "滤镜"
   }
}
```

18.4. 标准C2PA断言摘要

标准C2PA断言列于表7“标准C2PA断言”：

表 7. 标准 C2PA 声明

类型	断言	模式	序列化
操作	c2pa.actions, c2pa.actions.v2	C2PA	CBOR
断言元数据	c2pa.assertion.metadata	C2PA	CBOR
资产参考	c2pa.asset-ref	C2PA	CBOR
资产类型	c2pa.asset-type（已弃用）、c2pa.asset-type.v2	C2PA	CBOR
基于 BMFF 的哈希	c2pa.hash.bmff（已移除）、c2pa.hash.bmff.v2（已弃用）、c2pa.hash.bmff.v3	C2PA	CBOR
证书状态	c2pa.certificate-status	C2PA	CBOR
云数据	c2pa.cloud-data	C2PA	CBOR
集合数据哈希	c2pa.hash.收集.数据	C2PA	CBOR
数据哈希	c2pa.hash.data	C2PA	CBOR
深度图	c2pa.深度图.G深度	https://developers.google.com/深度图元数据/参考	CBOR
嵌入数据	c2pa.嵌入式数据	C2PA	JUMBF嵌入文件
字体信息	font.info	C2PA	CBOR
通用框哈希	c2pa.hash.boxes	C2PA	CBOR
成分	c2pa.ingredient、c2pa.ingredient.v2、c2pa.ingredient.v3	C2PA	JUMBF 嵌入文件
元数据	c2pa.metadata	C2PA	JSON-LD
多资产哈希	c2pa.hash.multi-asset	C2PA	CBOR
软绑定	c2pa.软绑定	C2PA	CBOR
缩略图	c2pa.thumbnail.claim（声明创建时间）、c2pa.thumbnail.ingredient（导入成分）	C2PA	嵌入文件
时间戳	c2pa.时间戳	C2PA	CBOR

18.5. 数据哈希

18.5.1. 描述

验证非BMFF资产部分完整性的最常见方式是通过数据哈希断言中的硬绑定（即加密哈希）。然而，对于那些"盒式"但与BMFF不兼容的格式，建议使用通用盒式哈希断言。

数据哈希断言支持根据第13.1节"哈希处理"所述创建并存储哈希值，该值应存储于哈希字段中。

每个数据哈希断言定义了哈希计算所覆盖的特定字节范围。若仅需对资产的部分内容进行哈希处理，则需在排除项字段的数组值中列出需排除的范围。这些排除范围应按起始位置升序排列，且不得存在重叠。

对于数据哈希排除范围，该范围的起止点应位于同一逻辑单元（如盒、段、对象）内，且不得与该单元关联的任何头部或长度字段重叠（自由盒或填充数据除外）。声明生成器有责任以确保攻击者在这些范围中放置的任何数据均不会实质性影响资产解释的方式定义排除范围。此外，声明生成器应确保排除范围仅包含来自C2PA清单存储库的内容，或资产元数据（如EXIF、IPTC元数据）。可跳过的元数据示例包括未经验证的用户名或图像旋转信息。

本规范的先前版本曾提供一个 `url` 字段，用于指向哈希数据的位置，但该字段从未被使用。现已弃用该字段，转而采用资产引用断言。声明生成器不得在数据哈希断言中添加此字段，消费者在遇到该字段时应予以忽略——但根据第15.10.3节"断言验证"所述，此规定不影响该字段作为待验证内容组成部分的包含。

数据哈希断言应采用标签 `c2pa.hash.data`。云数据断言中不得包含数据哈希

断言。

数据哈希断言不得与压缩清单配合使用。

> **注** 此限制旨在解决两者之间的技术不兼容问题。

18.5.2. 模式与示例

该类型的模式由以下CDDL定义中的`data-hash-map`规则定义：

```
; 同时检查哈希映射内的可选性
; 用于存储资产部分或全部数据的加密哈希值的数据结构

; 以及计算哈希所需的附加信息。data-hash-map = {
  ? "exclusions": [1* EXCLUSION_RANGE-map], ; 范围的`start`值呈单调递增，且任意两个范围不得重叠。
  ? "alg":tstr .size (1..max-tstr-length), ; 用于计算本断言哈希值的密码哈希算法标识符字符串，取自C2PA哈希算法标识符列表。若该字段缺失，则采用
```

```
若两者均存在，则使用本结构中的字段。若上述位置均无值，则该结构无效；无默认值。
   "hash": bstr, ; 哈希值的字节字符串
   "pad": bstr, ; 用于填充空间的零填充字节字符串
   ? "pad2": bstr, ; 可选的零填充字节串，用于填充空间
   ? "name": tstr .size (1..max-tstr-length), ; （可选） 此哈希表的可读性描述
   ? "url": uri, ; 未使用且已弃用。
}

EXCLUSION_RANGE-map = {
   "start": uint, ; 范围起始字节位置 "length": uint, ; 需排除的数据字节数
}
```

以下为 CBOR 诊断标记法（RFC 8949 第 8 节）示例：

```
{
   "alg" : "sha256",
   "pad" : '0000',
   "hash": 'Auxjtmax46cC2N3Y9aFmBO9Jfay8LEwJWzBUtZ0sUM8gA=',"name": "JUMBF清单"
   "exclusions": [
      {
         "start": 9960,
         "length": 4213
      }
   ],
}
```

通常，排除项的起始值和长度值应采用其首选序列化格式（即"尽可能短"）。但当需要创建数据哈希断言而起始值和长度值尚未确定时，应采用"尽可能大"的格式创建，即32位整数。

填充值必须始终存在，但除非用于多遍处理中的字节替换（即"填充"），否则应为长度为0的零填充字节串。pad2是可选的零填充字节串，当pad无法实现所需填充时使用。

注 第10.4节“多步处理”详述了如何填入正确值并调整填充。

18.5.3. JPEG 1的特殊考虑

在对将嵌入C2PA清单的JPEG 1（.jpg）文件进行哈希处理时，APP11标记（FFEB）以及所有包含JUMBF数据的APP11分段的长度（Lp）均应纳入排除范围。

注 所有包含C2PA清单JUMBF的APP11分段均连续排列，因此仅需设置单一排除范围。

18.5.4. PNG格式的特殊处理

对将嵌入C2PA清单的PNG（.png）文件进行哈希处理时，必须将包含JUMBF数据的分块的长度及其'caBX'字段（代表分块类型）纳入排除范围。

18.6. 基于BMFF的哈希

18.6.1. 描述

基于BMFF的资产中，索赔生成器希望通过硬绑定（即加密哈希）唯一标识的部分，应使用基于BMFF的哈希断言进行描述。

基于BMFF的哈希断言应采用标签c2pa.hash.bmff.v3。

注	本标准早期版本亦记录了c2pa.hash.bmff和c2pa.hash.bmff.v2 声明。
重要	验证器应忽略任何 c2pa.hash.bmff 断言，处理清单时应视该断言不存在。

基于 BMFF 的哈希断言不得出现在云数据断言中。

本规范的早期版本曾提供一个 url 字段，用于指向哈希数据的位置，但该字段从未被使用。现已弃用该字段，转而采用资产引用断言。声明生成方不得在BMFF哈希断言中添加此字段，接收方若发现该字段应予以忽略——但根据第15.10.3节"断言验证"所述，此规定不影响该字段作为待验证内容组成部分的包含。

18.6.2. 哈希计算

要计算BMFF哈希值字段中指定的哈希值，需将文件的所有字节（除那些与排除数组中任何条目匹配的BMFF框或其子集外）加入哈希计算。

完整包含的框体及其框头均纳入哈希计算的输入数据。同理，完整排除的框体及其框头均不纳入哈希计算的输入数据。当通过排除规范中的子集字段将框部分排除在哈希输入数据之外时，子集字段中定义的相对字节偏移量所指的待排除框部分，是以包含框头信息的框起始位置为基准的偏移量，而非以框内容起始位置为基准的偏移量。这些子集范围应按偏移量递增顺序排列，且不得重叠。

在 c2pa.hash.bmff.v2（已弃用）和 c2pa.对于未被整体排除的根盒，其哈希输入数据由二进制字符串偏移量 || 数据拼接而成。其中偏移量定义为盒子在大端字节序格式下的绝对文件偏移量（8字节整数），数据定义为盒子内容（含头部）减去所有排除项。此处"||"

表示二进制字符串的连接运算。当bmff-hash-map同时包含哈希值和默克尔字段时，默克尔树哈希计算中不得包含偏移量。

此外，`c2pa.hash.bmff.v2`（已弃用）和`c2pa.hash.bmff.`v3断言包含以下特性：

- 对于任何根盒，其哈希计算所用的输入数据开头都包含绝对文件字节偏移量。这确保了哈希中包含的根盒在文件中的位置无法改变。
- 当bmff-hash-map同时包含哈希值和默克尔字段时，`mdat`字段将不再被整体排除

  字段时，不再完全排除mdat框。取而代之的是，排除列表中的强制性条目将排除该框的大部分内容。

注：	这两项特性既确保了`m`dat在文件中的位置无法改变，同时又消除了 当bmff-hash-map同时包含哈希值和默克尔字段时，无需为每个默克尔树哈希单独设置偏移量。

当且仅当满足以下所有条件时，某个框才与排除数组中的排除条目匹配：

- 该框在文件中的位置与排除映射条目的`xpath`字段相匹配。例如，排除xpath

  `/foo/bar[2]`将匹配匹配位置`/foo[3]/bar[2]`和`/foo[2]/bar[2]`，但不`/foo[3]/bar[1]`或`/foo[3]/bar[2]/baz[1]`。
- 若排除映射条目中指定了长度，则框的长度必须与排除映射条目

  条目中的长度字段完全匹配。注意：长度包含框头信息。
- 若排除映射条目中指定了版本，则该框为完整框（FullBox），且框的版本与排除映射条目的版本字段完全匹配。
- 如果在排除映射条目中指定了标志位（精确为3字节的字节数组），且该框为FullBox类型。当`ex`act设置为true或未指定时，该框的标志位（bit(24)，即3字节）也需与排除映射条目的标志位字段完全匹配。若`exact`设置为 false，则盒子的标志位（bit(24)，即 3 字节）与排除映射条目标志字段的按位与运算结果完全匹配（即盒子至少包含这些位，但可能还包含其他位）。
- 若排除映射条目中指定了数据（对象数组），则数组中每个项对应的二进制数据在该项相对字节偏移字段处的值，必须与该项的字节字段值完全匹配。

`xpath`字段的字符串语法应严格限制于下列子集。

- 仅允许使用缩写语法。
- 仅允许使用完整路径。
- 仅应使用通过节点或节点`[整数]`进行节点选择。
- 禁止使用后代语法（即`//`）。
- 所有节点均应采用BMFF `4cc`编码。

示例5. `xpath` 字段完整语法

```
xpath = '/' nodes nodes =
node
        | node '/' nodes node =
box4cc
       | box4cc '[' 整数 ']'
```

其中：

`box4cc` 是符合 `ISO/IEC 14496-12` 标准的 `BMFF` 盒子允许的任意 `4cc`。整数是任意非零正整数，不带前导零。

任何给定的排除条目可匹配零个或多个框。排除条目不必精确匹配一个框。

非叶子XPath节点仅可指向不包含自身字段（即仅包含子框而无数据）且未继承自FullBox的容器框。此设计确保C2PA验证器无需关注包含其他框的特殊框的语法与语义。若需完全或部分排除此类特殊框的子框，则排除映射条目的xpath字段应指向特殊框本身，子集映射字段应排除包含被排除子框数据的字节范围。例如，'sgpd'框虽包含其他框，但因继承自FullBox而具有特殊性；因此若需从`'sgpd'`中全部或部分排除子框，断言应使用指向'sgpd'本身的xpath字段（ 例如：
`/moof/traf/sgpd`) 并应使用子集映射字段排除所需字节。

若C2PA清单嵌入文件中，则包含该清单的框体应作为排除数组的条目之一
数组中的条目之一。更多信息请参阅第A.5节“将清单嵌入基于BMFF的资产”。

若在C2PA清单创建后移除了非根排除框，则应使用相同大小的"空闲"框进行替换，以确保其他框的哈希值所依据的输入数据不失效。若在C2PA清单创建后通过压缩清单缩减存储空间，则应在原位置插入"空闲"框以保持偏移量不变。若预计在C2PA清单创建后可能添加非根排除区块，则在清单创建时应预留足够空间的"空闲区块"，该区块需通过完整XPath路径的排除条目进行标记。当排除框被添加或C2PA清单存储空间增大时，应缩小（或移除）"空闲"框以容纳新增数据。但若"空闲"框空间不足，则应使用标准清单。

通过MP4盒将C2PA数据嵌入基于BMFF的资产时，会改变其他MP4盒的文件偏移量，以及任何根盒哈希计算中包含的输入数据中的绝对文件字节偏移量。这些盒和偏移量应以其嵌入后的值（而非嵌入前的值）纳入哈希计算的输入数据，否则基于BMFF的哈希断言将无法通过验证。

实现方案可通过以下三种方式确保所有文件字节偏移量的嵌入后值被哈希处理：

1. 使用"自由"框。

a. 确定将嵌入的C2PA框的合理最大尺寸。所有支持C2PA的MP4框均允许末尾存在未使用的填充字节，因此可适当高估"空闲"框的尺寸——任何多余字节均会被忽略。

b. 将上述尺寸的"自由"框插入资产文件，并相应更新所有偏移量。

c. 对资产文件执行哈希处理，并将"/free"加入排除列表。

d. 创建并签名清单文件。生成C2PA盒区。

e. 用C2PA盒覆盖"自由"盒。

2. 采用两步处理方案。

a. 计算基于BMFF的哈希断言及任何默克尔盒的精确大小。后者需解析资产文件以确定默克尔树的大小。

b. 计算最终清单的确切大小。

c. 对资产文件执行哈希处理。在将包含文件偏移量的数据块纳入哈希计算的输入数据前，需将其偏移量更新为正确值。使用上述更新后的绝对文件偏移量，通过(偏移量 || 数据)计算哈希输入数据。如前所述，当bmff-hash-map同时包含哈希字段和默克尔字段时，偏移量不计入默克尔树哈希的计算数据。

d. 创建并签名清单文件。创建C2PA数据框。

e. 插入C2PA框。

3. 将更新后的清单存储放在BMFF文件末尾。

a. 将原始清单存储的`box_purpose`字段从`manifest`改为`original`。

b. 创建并签名清单文件。

c. 创建C2PA内容溯源框，将`box_purpose`设置为更新。

d. 将更新后的清单插入C2PA内容溯源框。

e. 将C2PA内容溯源框插入BMFF文件末尾。

f. 若在存在更新清单存储时添加标准清单，则更新清单存储内容将移至"原始"清单。

g. 更新后的清单存储随后从文件末尾移除，从而通过单一清单实现对常见使用场景的向后兼容性。

h. 将"原始"清单存储的`box_purpose`字段改回`manifest`，并按常规方式添加标准清单。

注	box_purpose字段不包含在哈希值中，可自由变更而不会导致任何数据失效 现有哈希值。同样地，追加新的C2PA内容来源框也不会使现有哈希失效。

虽然两遍处理方法的复杂度显著更高，但它能在无需预先知晓最大显式大小的情况下实现正确哈希，同时最大限度地缩小最终资源的体积。常见的包含文件偏移量的盒子（非穷举）包括'iloc'、'stco'、'co64'、'tfhd'等。

预先知晓最大清单大小。同时它能最小化最终资源的体积。常见的包含文件偏移量的盒子（非穷举）包括'iloc'、'stco'、'co64'、'tfhd'、'sidx'和'saio'。

将更新后的清单置于BMFF文件末尾的选项，可在缺乏足够"空闲"数据框或不希望采用双遍法复杂方案时实现更新。该选项还支持在原子'stco'数据框中使用包含部分数据偏移信息的分块偏移量。

18.6.3. 架构与示例

c2pa.hash.bmff.v2（已弃用）和c2pa.hash.bmff.v3断言的模式由bmff-hash-map规则定义，具体见以下CDDL定义：

```
bmff-哈希映射 = {
   "exclusions": [1* exclusions-map],
   ? "alg": tstr .size (1..max-tstr-length), ; 用于计算此哈希值的密码哈希算法标识符字符串，取自C2PA哈希算法标识符列表。若该字段缺失，则从外围结构中按其定义获取哈希算法。若两者均存在，则使用本结构中的字段。若所有位置均无值，则该结构无效；无默认值。
   ? "hash": bstr, ; 对于非分片MP4，此字段为整个BMFF文件（排除exclusions数组中列出的盒子）的哈希值。对于分片MP4，此字段必须缺失。
   ? "merkle": [1* merkle-map], ; 包含一组默克尔树行及其关联数据，用于验证资产内单个'mdat'框、多个'mdat'框及/或独立分片文件。
   ? "name": tstr .size (1..max-tstr-length), ; 可选） 该哈希所涵盖内容的人类可读描述。
   ? "url": uri, ; 未使用且已弃用。
}

;（可选） 恰为3字节的CBOR字节串。 flag-type = bytes

flag-t = flag-type .eq 3

exclusions-map = {
   "xpath": tstr, ; 需从哈希中排除的盒子位置，以https://www.w3.org/TR/xpath-10/版本的xpath格式字符串表示，采用高度约束的语法。
   ? "length": uint, ; （可选） 需从哈希中排除的最叶层框必须具备的长度。
   ? "data": [1* data-map], ; （可选） 指定相对字节偏移量处叶节点框中的数据必须与指定数据完全一致，该框才会被排除在哈希之外。
   ? "子集":[1* 子集映射], ; （可选） 仅排除框的此部分不参与哈希计算。数组中每个条目的相对字节偏移量必须单调递增。数组内子集不得重叠。最后一项可设为零长度，表示从该相对字节偏移量起剩余区域均被排除。允许设置超出框体长度的相对偏移量或偏移量加长度组合，但框体末尾后的字节永远不会被哈希。
   ? "version": int, ; （可选） 需在叶节点框中设置的版本号，用于将该框排除在哈希计算之外。仅适用于继承自FullBox的框。
   ? "flags": flag-t, ; （可选） 字节字符串，精确为3字节。必须在叶节点框中设置的24位标志位，用于将该框排除在哈希之外。仅适用于继承自FullBox的框。
   ? "exact": bool, ; （可选） 表示标志必须完全匹配。若未指定，默认为 true。仅适用于继承自 FullBox 的盒子且
```

```
同时指定了flags时才生效。
}

data-map = { "offset":
   uint, "value" : bstr,
}
子集映射 = { "偏移量": 无符
   号整数, "长度": 无符号整数
   ,
}

; 映射中的每个条目包含一条默克尔树行及其关联数据，用于验证单条记录
; 资产内的'mdat'框或多个'mdat'框。", merkle-map = {
   "uniqueId": int, ; 基于1的唯一标识符，用于在不同文件间区分，确定验证特定'mdat'框时应采用哪棵默克尔树。
   "localId": int, ; 标识默克尔树的本地标识符。
   "count": int, ; 默克尔树中叶节点的数量。空节点不计入此统计。
   ? "alg": tstr .size (1..max-tstr-length), ; 用于计算此默克尔树哈希值的密码哈希算法标识符字符串，取自C2PA哈希算法标识符列表。若该字段缺失，则采用包含结构定义的`alg`值作为哈希算法。若两者均存在，则使用本结构中的字段值。若所有位置均无值，则该结构无效；无默认值。
   ? "initHash": bstr, ; 对于拆分至多个文件的分段MP4资源，此字段必须存在，且为整个初始化段文件的哈希值（该文件包含由本默克尔树哈希的片段，但排除排除数组中列出的框）。对于存储为单个扁平MP4文件的分段MP4资源，该字段必须存在，且为首个'moof'盒之前所有字节的哈希值（排除排除数组中列出的盒）。对于非分段MP4，该字段必须缺失。
   "哈希值": [1* bstr], ; 表示默克尔树单行（可能是叶子行、根行或任意中间行）的有序数组。该行的深度由数组项数隐含（计算得出）。
   ? "固定块大小": uint, ; 对于非分段MP4资源（其mdat框需分段验证），此字段可存在。该字段表示默克尔树中指定叶节点的非负字节大小。分段MP4不包含此字段。
   ? "variableBlockSizes": [1* int], ; 对于非分片MP4资源（其mdat框需分段验证），此字段可存在。数组中每个条目对应默克尔树中某个叶节点的非负字节大小。元素数量等于`count`，数值总和等于mdat有效负载大小。对于分段MP4，此字段不存在。
}
```

以下是一个采用CBOR诊断标记法（RFC 8949第8节）的示例，用于描述整体式MP4文件资源，其中mdat框作为整体进行验证：

```
{
   "hash": b64'EiAuxjtmax46cC2N3Y9aFmBO9Jfay8LEwJWzBUtZ0sUM8gA=',"name": "示例
   `c2pa.hash.bmff.v2` 断言",
   "exclusions": [
      {
         "data": [
            {
               "value": b64'2P7D1hsOSDySl1goh37EgQ==', "offset": 8
            }
```

```
        ],
        "xpath": "/uuid"
      },
      {
        "xpath": "/ftyp"
      },
      {
        "xpath": "/mfra"
      },
      {
        "xpath": "/moov[1]/pssh"
      },
      {
        "xpath":
        "/emsg","data": [
          {
            "value": b64'r3avWCpXHkmKHATFsV0Q5g==', "offset": 20
          }
        ]
      }
   ]
}
```

以下为由碎片化MP4文件组成的资产在CBOR诊断标记法（RFC 8949第8节）中的示例：

```
{
   "alg": "sha256",
   "name": "示例 `c2pa.hash.bmff.v3` 断片化MP4断言","merkle": [
      {
        "count": 23,
        "哈希值": [ b64'HvWZOxKMfkSatRAygs8DJfnEEcN/G1BNi359NdIDxbQ=', b64'HvWZOxKMfkSatRAygs8DJfnEEcN/G1BNi359NdIDxbQ='
],
        "localId": 19,
        "初始哈希": b64'Hf0IgeqbL0m+FTTLpUWwsDGR8pvhUR1AlwvaXjQ0qGY=',"唯一标识": 17
      },
      {
        "count": 69,
        "hashes": [ b64'9Zk7Eox+RJq1EDKCzwMl+cQRw38bUE2Lfn010gPFtB0=',
b64'9Zk7Eox+RJq1EDKCzwMl+cQRw38bUE2Lfn010gPFtB0=',b64'mTsSjH5EmrUQMoLPAyX5xBHDfxtQTYt+fTXSA8W0Hf0='b64'mTsSjH5EmrUQMoLPAyX5
xBHDfxtQTYt+fTXSA8W0Hf0='b64'OxKMfkSatRAygs8DJfnEEcN/G1BNi359NdIDxbQd/Qg=' ],
        "localId": 38,
        "初始哈希值": b64'Hf0IgeqbL0m+FTTLpUWwsDGR8pvhUR1AlwvaXjQ0qGY=',"唯一标识符": 34
      },
      {
        "count": 46,
        "哈希值": [ b64'OxKMfkSatRAygs8DJfnEEcN/G1BNi359NdIDxbQd/Qg=' ],"本地ID": 57,
        "初始哈希值": b64'Hf0IgeqbL0m+FTTLpUWwsDGR8pvhUR1AlwvaXjQ0qGY=',"唯一标识符": 51
      }
   ],
   "exclusions": [
      {
```

```
        "data": [
          {
            "value": b64'2P7D1hsOSDySl1goh37EgQ==', "offset": 8
          }
        ],
        "xpath": "/uuid"
      },
      {
        "xpath": "/ftyp"
      },
      {
        "xpath": "/mfra"
      },
      {
        "xpath": "/moov[1]/pssh"
      },
      {
        "data": [
          {
            "value": b64'9Q==', "offset": 5
          },
          {
            "value": b64'UAJXD79SlkG9rfnmcsqTUA==', "offset": 20
          },
          {
            "value": b64'OxKM',"offset": 70
          }
        ],
        "flags": b64'ZDNx',
        "xpath": "/emsg","length":
        200,
        "子集": [
          {
            "长度": 7,
            "offset": 5
          },
          {
            "length": 28,
            "偏移量": 20
          },
          {
            "length": 63,
            "偏移量": 45
          },
          {
            "length": 112,
            "偏移量": 80
          }
        ],
        "version": 1
      }
    ]
}


{
   "alg": "sha256",
   "name": "示例 `c2pa.hash.bmff.v3` 断言用于非分段 MP4",
  "默克尔树": [
```

```
    {
      "count": 3,
      "hashes": [ b64'HvWZOxKMfkSatRAygs8DJfnEEcN/G1BNi359NdIDxbQ=',
b64'HvWZOxKMfkSatRAygs8DJfnEEcN/G1BNi359NdIDxbQ=' ],"variableBlockSizes": [ 100, 30, 20
      ],
      "本地ID": 19,
      "初始哈希值": b64'Hf0IgeqbL0m+FTTLpUWwsDGR8pvhUR1AlwvaXjQ0qGY=',"唯一标识符": 17
    }
  ],
  "exclusions": [
    {
      "data": [
        {
          "value": b64'2P7D1hsOSDySl1goh37EgQ==', "offset": 8
        }
      ],
      "xpath": "/uuid"
    },
    {
      "xpath": "/ftyp"
    },
    {
      "xpath": "/mfra"
    },
    {
      "xpath": "/moov[1]/pssh"
    },
    {
      "data": [
        {
          "value": b64'9Q==', "offset": 5
        },
        {
          "value": b64'UAJXD79SlkG9rfnmcsqTUA==', "offset": 20
        },
        {
          "value": b64'OxKM',"offset": 70
        }
      ],
      "flags": b64'ZDNx',
      "xpath": "/emsg","length":
      200,
      "子集": [
        {
          "长度": 7,
          "offset": 5
        },
        {
          "length": 28,
          "偏移量": 20
        },
        {
          "length": 63,
          "offset": 45
        },
        {
          "length": 112,
          "偏移量": 80
```

```
          }
        ],
        "version": 1
      }
    ]
}
```

该算法的伪代码实现见示例6“基于BMFF的哈希断言伪代码”。

示例6. 基于BMFF 的哈希断言伪代码

```
offset = 0
当 (偏移量 < 文件长度)
    从偏移量开始定位首个与排除数组条目匹配的框首字节，称之为 first_excluded_byte
        若未找到此类框，则设置 first_excluded_byte = 文件长度确定该框的长度，称之为 excluded_byte_count
        若未找到此类区块，则设excluded_byte_count = 0
    向哈希值中添加偏移量与first_excluded_byte减1之间（含）的所有字节
    若 first_excluded_byte < 文件长度，且在确定 first_excluded_byte 值的排除区内存在子数组
        设置 next_included_begin = first_excluded_byte
        对于确定 first_excluded_byte 值的排除区内子集数组中的每个条目
            设置 next_excluded_begin = 此子集数组条目的偏移字段加上 first_excluded_byte
            若 next_excluded_begin > next_included_begin
                向哈希表添加 next_included_begin 与 next_excluded_begin 之间减去1的所有字节（含边界值
)
            设置 next_included_begin = 此子集数组条目的长度字段加上 next_excluded_begin
        若 next_included_begin < first_excluded_byte + excluded_byte_count 则将
            next_included_begin 至 first_excluded_byte 之间的所有字节（包含边界）添加至哈希表
first_excluded_byte + excluded_byte_count 减去 1（包含边界）的字节添加到哈希中设置偏移量 =
    first_excluded_byte + excluded_byte_count
```

生成默克尔映射哈希值的示例见例7“默克尔映射建议示例”。

示例7. 默克尔映射的建议示例

```
若字段`fixedBlockSize`和`variableBlockSizes`均不存在
    向哈希值添加 mdat 有效负载中 begin_address 与 last_address 之间的所有字节若存在 `fixedBlockSize` 字段且不存在
`variableBlockSizes` 字段
    当 (1)
        next_address = begin_address + fixedBlockSize
        若 next_address > mdat 有效负载末尾地址next_address = mdat 有效负载末尾地址 +
            1hash_complete = true
        向哈希值添加 begin_address 到 next_address 减 1（含）之间的所有字节
        若 hash_complete 为真 则 break
        初始地址 = 下一地址
```

```
如果存在 `variableBlockSizes` 字段且不存在 `fixedBlockSize` 字段
    对于 (blockSize in variableBlockSizes) next_address =
        begin_address + blockSize
        若 next_address > mdat 有效负载的最后地址 next_address = mdat 有效负载的最后地址 + 1
            hash_complete = true
        向哈希值添加 begin_address 与 next_address 之间减一的所有字节（含边界）
        若 hash_complete 为真 则 跳出循环
        初始地址 = 下一地址
```

18.6.4. 排除列表配置文件

18.6.4.1. 通用

本节描述一组预定义的、命名的扩展列表配置文件。

18.6.4.2. 基本配置文件

典型的非时序媒体（例如静态照片）和时序媒体（例如带或不带音频轨道的视频，无论是否碎片化）只需包含《排除列表要求》中列出的强制性排除项。

18.6.5. 分段式BMFF实体图

图15“分段BMFF实体图”展示了构成分段BMFF清单的C2PA对象间关系。

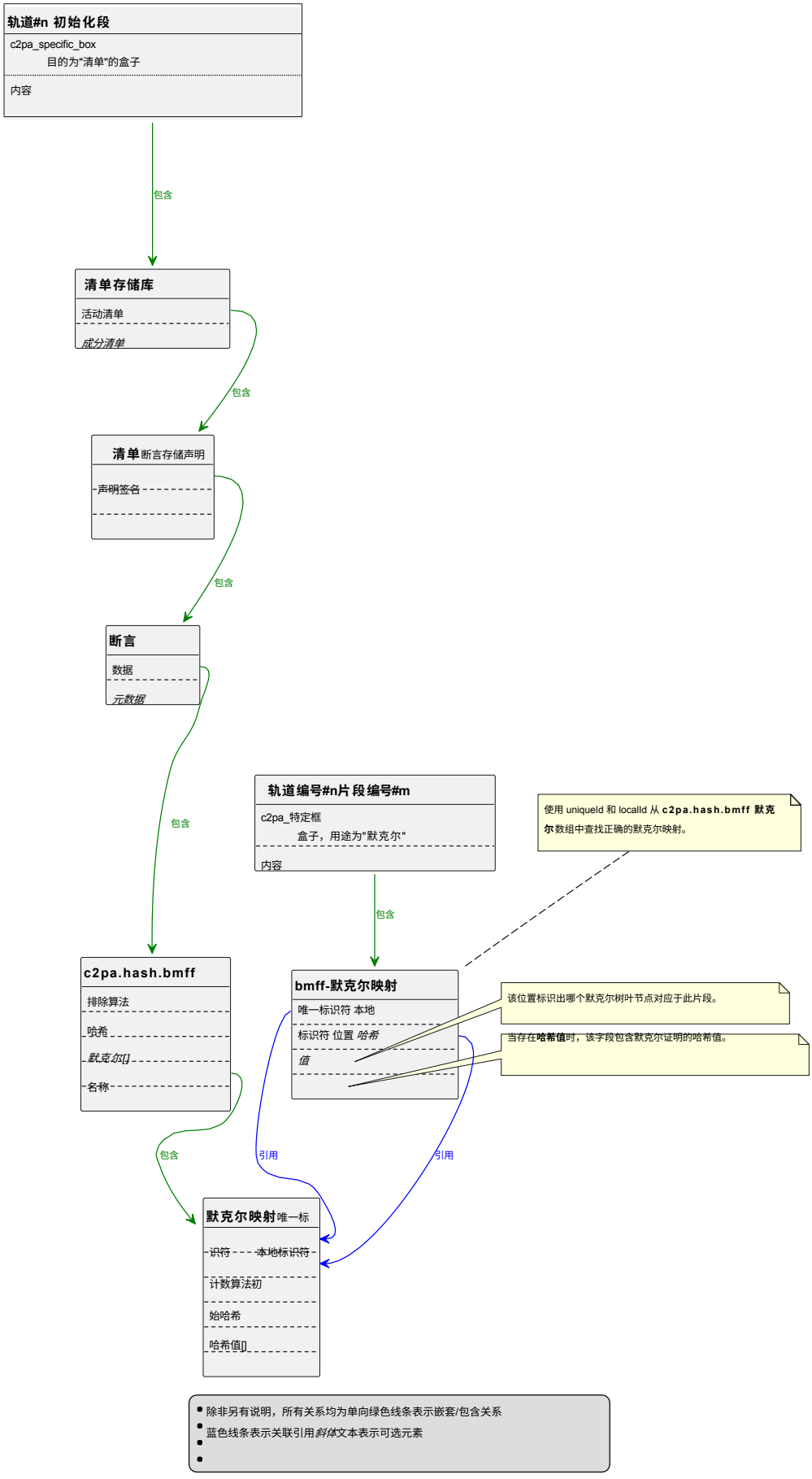

图15. 碎片化BMFF实体图

18.6.6. 验证

验证给定数据块需先验证默克尔映射字段的初始哈希值是否与对应初始化段匹配，随后在默克尔映射字段的哈希数组中定位正确条目，并将其与数据块数据的哈希值进行比对验证。若必要，可通过数据块中bmff-默克尔映射字段指定的哈希值，利用默克尔证明推导该哈希值。

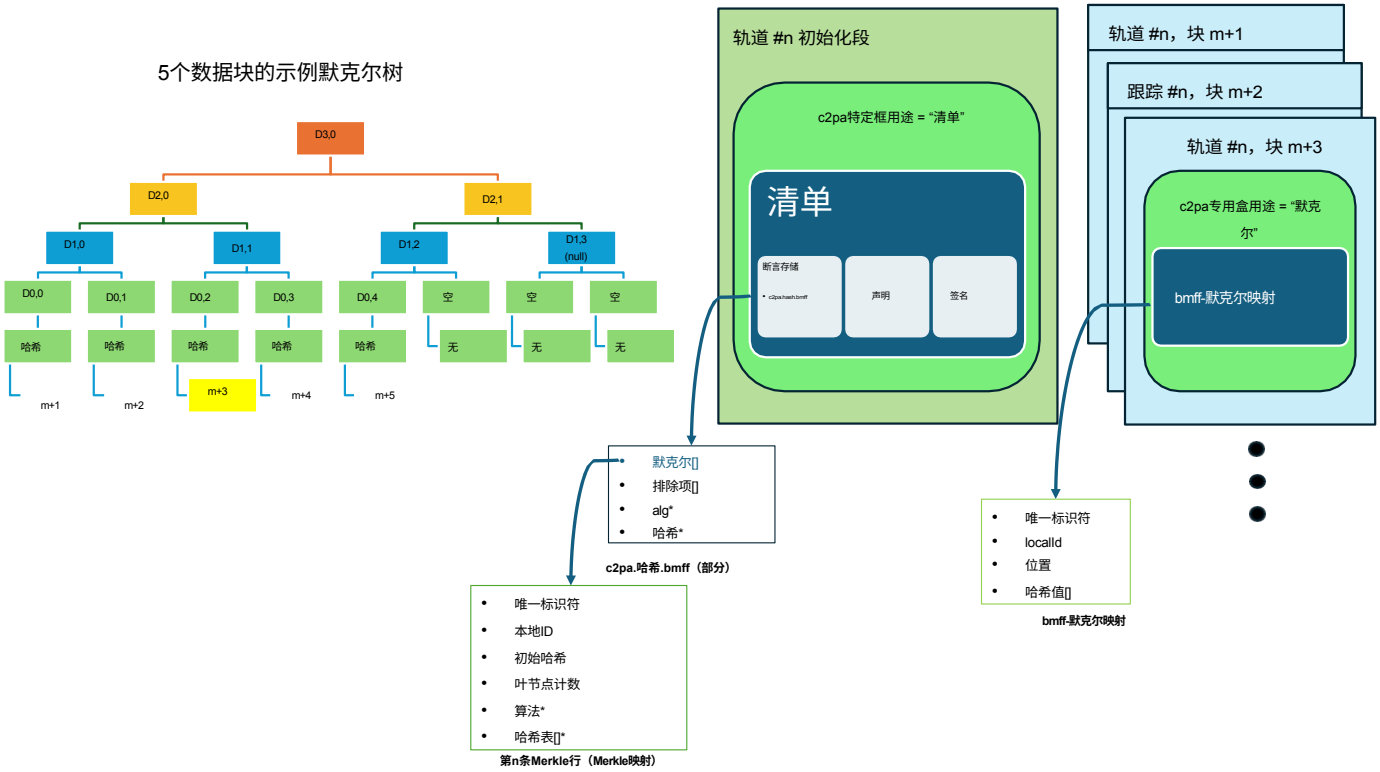

图16. 初始化段验证及数据块验证示例

要验证轨道块 **m+3**，必须先验证对应的初始化段。每个轨道初始化段中的 c2pa 专用清单框将包含清单存储库。若资产包含多个初始化段，则每个初始化段中的清单存储库必须完全一致。这使验证者能够确认轨道属于更大型的集合。活动清单的c2pa.bmff.hash断言将包含一个默克尔字段，该字段包含一个默克尔映射对象数组，每个轨道对应一个对象。

18.6.6.1. 步骤

1. 从数据块c2pa专用默克尔框中的bmff-默克尔映射获取uniqueId与localId。使用uniqueId和localId在初始化分区的c2pa.bmff.hash断言默克尔数组中查找匹配的默克尔映射。
2. 若使用c2pa.bmff.hash排除项与默克尔映射算法计算的初始化段哈希值等于您刚定位的默克尔映射中的initHash，则初始化段验证通过。

注意	bmff-hash-map 顶层的参数 alg & hash 用于整体式 MP4，而默克尔映射中的 alg & hashes 用于分段式 MP4。

要完成对块 **m+3** 的验证：我们正在查看步骤 1 中找到的轨道 #n 的默克尔映射，在本例中，它包含默克尔树的第 2 行——**D2,0** 和 **D2,1**。

3. 使用`c2pa.hash.bmff`排除数组和默克尔映射中的算法对哈希块**m+3**进行哈希运算，得到 **D0,2(衍生)**。
4. 块 m+3 的 `bmff-`默克尔映射哈希数组（默克尔证明）将包含块 m+4（**D0,3**）的哈希值及第一行哈希值 **D1,0**。
5. **对D0,2(衍生)与D0,**3进行哈希运算，**生成D1,1(衍生)**。**将D1,0与D1,1(衍生)**进行哈希运算，**生成D2,0(衍生)**。
6. 若 **D2,0(推导值)** = 断言默克尔映射哈希参数中存储的 **D2,**0，且对应的初始化段已在步骤 2 中验证通过，则分块 m+3 已完成验证。

18.7. 通用盒哈希

18.7.1. 描述

对于采用非BMFF盒格式（如JPEG、PNG或GIF）的资产，声明生成器应使用通用盒哈希断言进行完整性验证，并采用强绑定机制（即密码学哈希）。

通用箱哈希断言应具有标签 `c2pa.hash.boxes`。此类断言由结构数组构成，每个结构包含一个或多个箱（通过其名称/标识符标识），以及覆盖这些箱数据（及文件中可能存在的其他数据）的哈希值，同时注明所用哈希算法。盒子在断言中的排列顺序须与资产文件中的顺序一致，包括包含C2PA清单的盒子。若资产中存在未明确包含于本断言的其他盒子，或盒子排列顺序错误，则清单将在验证过程中被拒绝，具体详见第15.12.3节"通用盒子哈希验证"。

一个数据框可能包含一个排除字段，该字段为布尔值，用于指示验证器在验证过程中是否可忽略该数据框（及其关联哈希值）。若该字段缺失，或字段存在且值为`false`，则必须对数据框进行哈希处理并比较值。对于包含值为`true`的`excluded`字段的框，声明生成器应包含准确的哈希值，以兼容不识别该字段的旧版验证器。若声明生成器无需考虑向后兼容性，则应将哈希值写为二进制字符串`00` 单字节值为0）。

当存在多个相同框类型的实例时（例如JPEG 1文件中包含多个APP1段），每个实例均应在声明中单独列出。共享相同段标识符的片段式JPEG段也应作为独立框列出，但构成C2PA清单存储（如下所述）的段除外。

哈希值的生成方法详见第13.1节“哈希处理”，该值应存在于哈希字段中。对于一组数据框，其哈希值应从首框起始位置计算至末框结束位置，期间包含数据框间可能存在的任意字节。

注 使用一组框时，所有数据都包含在哈希中，范围从第一个框的起始点到最后一个框的终点。

包含在哈希值中。但当单独列出每个框时，额外数据不被包含，仅限于所列框内的数据。

包含C2PA清单存储的盒子（例如PNG的caBX或GIF的21FF）也应单独列出。为明确标识其为C2PA清单盒子，该盒子名称应为C2PA，其哈希值应为二进制字符串`00`（单字节值为0）。C2PA清单存储区应作为单一数据框呈现，即使在JPEG文件中该框被分割为多个`APP11`标记段的情况亦然。

注意

由于验证器常与文件解析器输出结合使用，安全最佳实践要求

对C2PA清单存储库外的所有文件内容进行哈希处理。这将确保媒体文件及其关联清单的完整性。

填充值必须始终存在，且应为零填充字节字符串，除非在多遍处理过程中被其他内容替换，此时则不应存在填充。

注

第10.4节"多步处理"描述了如何填入正确值并调整填充。

通用框哈希断言不得出现在云数据断言中。

18.7.2. 多部分资产的特殊处理

为支持由多个部分组成的文件格式（详见第18.9节"多资产哈希"），特定义一个额外的逻辑框，用于处理一个或多个部分的数据位于主部分基于框的数据之后的情况。该框应标记为`c2pa.after`（表示框结构末尾之后的任意数据）。若存在`c2pa.after`框，其应作为列表中的最后一个框，其哈希值应从最后一个框后的字节开始计算至物理文件末尾。

覆盖整个资产的硬绑定断言应是唯一可包含`c2pa.after`
盒。单个部分的哈希断言仅覆盖该部分自身内容，不得涉及其他部分。

18.7.3. 特定格式的处理

18.7.3.1. JPEG 特定处理

处理JPEG格式时，除C2PA标准（即JPEG 360）外，均使用APP11框。此类情况下，所有非C2PA的APP11框均应纳入哈希框列表。包含C2PA清单存储的APP11框应标识为`C2PA`。其余所有框均应采用ISO 10918-1:1994标准B.1表中的符号进行标识。

C2PA清单存储可通过以下特征识别：作为JUMBF超级框，其标签为`c2pa`，且JUMBF类型UUID为`63327061-0011-0010-8000-00AA00389B71`，具体参见第11.1.4.2节"清单存储"。

注

扫描起始框和重启框（标记为SOS和`RST[n]`）将包含各自标记后跟随的熵编码分段。

通过此方法也可支持JPEG的多图像格式（MPF）扩展，具体操作是按文件中出现的顺序列出所有数据框，前提是单个图像的结束标记（EOI）与下一个图像的开始标记（SOI）之间不存在其他数据。该数据框列表将按顺序枚举MPF中每个独立图像的分段（即从SOI开始）。

..., EOI, SOI, ..., EOI, ...)。然而，若索赔生成器计划将MPF文件视为多部分资产，则应使用c2pa.after框对首个独立图像（主部分）的EOI之后的附加部分进行哈希处理。

18.7.3.2. PNG格式特殊处理

PNG文件始终以8字节头部开头（89 50 4E 47 0D 0A 1A 0A）。要包含该头部，请使用特殊值PNGh作为盒子列表中的第一个盒子，并从图像的第一个字节开始进行哈希处理。

18.7.3.3. TIFF特有处理

TIFF文件始终以8字节头部开头。若需包含该头部，请在框列表中将特殊值TIFh设为首个框。

TIFF文件由一个或多个IFD（图像文件目录）构成，这些目录相当于"超级框"。每个IFD包含名为"IFD条目"或"TIFF字段"的条目数组，这些条目代表"框"。每个IFD条目的框名称应为Tag字段值转换为十进制字符串后的结果。

与其他盒式格式不同，IFD条目的数据可能不包含在条目内部（除非其长度为4字节或更小），而存在于文件的其他位置。

注： IFD条目的数据长度通过将数据值数量（如由IFD条目中的Count字段确定）与每个数据值的大小（由在IFD条目中输入字段）。

IFD条目的哈希值应基于该条目12字节的数据计算。若IFD条目长度超过4字节，则需将这12字节与条目所引用的文件数据进行拼接后计算哈希值——拼接起始位置为IFD条目"值偏移"字段指定的字节偏移量，并延续至该数据段的完整长度。

对于某些已知IFD条目——StripOffsets（273）、TileOffsets（324）和FreeOffsets（288）——其引用的数据本身即为指向实际数据的偏移量列表。此类情况下，哈希计算应基于按下列顺序连接的数据：

1. IFD的12字节，
2. 从值偏移量起始、长度为计数值乘以包含偏移量的类型大小的字节序列，以及
3. 按偏移量在列表中的顺序，分别提取该偏移量处的字节，其长度由类型关联的字节计数条目决定：StripByteCounts (279)、TileByteCounts (325) 和 FreeByteCounts (289)。

注 因此，TIFF中的图像数据将通过这种"偏移量"与"字节计数"的组合进行哈希处理。

TIFF还支持子IFD（SubIFD），这是一种通过引用指向并整合一个或多个IFD的IFD类型。此类不仅包含名为SubIFD（330）的类型，还包括EXIF（34665）、GPS（34853）和互操作性（40965）。对于所有此类IFD类型，以及任何以类似方式引用其他IFD的类型，哈希计算所依据的数据应按以下顺序拼接：

1. IFD的12字节标识符，
2. 或：
    a. 若N=1，则为从值偏移量起始、长度为包含所引IFD偏移量的类型大小的字节序列，或
    b. 若 N > 1，则包含数组偏移量长度的字节从值偏移量处开始，该数组偏移量包含指向 IFD 偏移量数组的偏移量，其长度为类型大小。随后连接从该偏移量处开始的字节，其长度为计数值乘以类型大小，这些字节包含指向每个"树状"IFD 的偏移量。
3. 对于每个引用的IFD，按本节规定递归计算该IFD在指定偏移处的哈希数据。

18.7.3.4. GIF特有的处理

包含'Packed Fields'属性的框的哈希值也将包含该属性所指示的可选数据。例如，若存在图像描述符将包含局部颜色表块，逻辑屏幕描述符将包含全局颜色表块。

所有包含块标签的框均应遵循以下命名规范："<块标签>"。

所有扩展块的命名规范如下："<扩展引入符><扩展标签>"。除上述命名规范外，唯一未描述的块为：

- 标题将标记为"GIF89a"。
- 基于表的图像数据将标记为"TBID"。
- 逻辑屏幕描述符将标记为"LSD"。

例如：

- 头部："GIF89a"。
- 尾部标记："3B"。
- 图像描述符："2C"。
- 注释扩展："21FE"。

18.7.3.5. RIFF特有的处理方式

RIFF文件块可嵌套在任意深度的树形结构中。该结构的根节点由一个或多个
L0 块组成，每个块的块标识符均为 RIFF。这些 RIFF 块的结构定义如下：

- 字节 0-3：块标识符，始终为 RIFF。
- 字节 4-7：块长度（减去 8 字节用于块标识符和块长度字段）。
- 字节 8-11：媒体类型标识符。
- 字节 12-n：数据块（所有 **L1** 数据块）。

在媒体类型标识符之后，RIFF块可能包含一个或多个**L1**子块，每个子块具有以下结构：

- 字节 0-3：块标识符。
- 字节 4-7：块长度（减去块标识符和块长度字段占用的 8 字节）。
- 字节 8-n：块数据。
- 字节 n+1：填充字节（如需）。

特殊块标识符LIST可用于**在**L1块内嵌套子块。此类LIST块采用**与L0** RIFF块相同的结构：

- 字节 0-3：数据块标识符，始终为 LIST。
- 字节 4-7：块长度（减去块标识符和块长度字段占用的 8 字节）。
- 字节 8-11：列表类型标识符。
- 字节 12-n：数据块（所有 **L2** 数据块）。
- 字节 n+1：填充字节（如有必要）。

为计算通用盒哈希值之目的，**每个L0**分块应视为单个盒子，其大小精确为12字节，盒名等于媒体类型标识符（第8-11字节）。每个**非**LIST类型的L1数据块应视为一个盒子，其名称等于数据块标识符（字节0-3），内容范围从数据块标识符起始处（字节0）延伸至填充字节（若存在则包含该字节）。每个LIST **L1**数据块应视为一个盒子，其名称等于列表类型标识符（第8-11字节），内容范围从数据块标识符起始位置（第0字节）延伸至填充字节（若存在则包含该字节）。所有嵌套在LIST **L1**分块内的分块（L2及更高层级）均应视为LIST **L1**分块内容的一部分，并作为单个数据框进行哈希处理。

在所有情况下，填充字节应被视为前一个数据块内容的一部分，并应包含在该框的哈希值中。

18.7.3.6. 字体特定处理

字体的表直接对应于哈希框，包括 C2PA 表。表始终按其在字体表目录中的出现顺序进行编号。

请注意，字表目录本身不属于哈希内容，因此不被任何哈希框覆盖。

计算包含头表的哈希值时，检查和调整值应视为零（0）。

头部表的哈希值时，应将校验和调整值视为零(0)。

字体表在通用框哈希断言中的分组（或不分组）取决于断言生成器。

注：面向广泛分发的字体可通过将每个字符表分配至独立数据框获益；如此当字体转换为其他格式时，其哈希值仍能保持有效验证。反之，自动生成大量字体的系统（如子集生成器）可能选择合并字符表至更少数据框以优化处理流程。此类情况下，由于跨表填充的存在，格式转换后数据框哈希值可能无法通过验证。

由于字体消费者不应对其无法识别的表格作出响应，现有的字体处理基础设施将要求头表格的检查和调整值包含C2PA表格本身最终确定的内容，包括其完整的本地清单。

18.7.4. 模式与示例

此类型的模式由CDDL定义中的box-map规则定义，详见CDDL for Box Hash：

CDDL 框哈希规范

```
box-map = {
   "boxes": [1* box-hash-map],
   ? "alg":tstr .size (1..max-tstr-length), ; 用于标识本断言中哈希计算所用密码哈希算法的字符串，取自C2PA哈希算法标识符列表。若该字段缺失，则采用包含结构的`alg`值作为哈希算法。若两者均存在，则使用本结构中的字段。若所有位置均无值，则该结构无效；无默认值。
}

box-hash-map = {
   "names": [1* box-name], ; 按出现顺序排列的盒标识符字符串数组（例如 `APP0`, `IHDR`）
   ? "alg": tstr .size (1..max-tstr-length), ; 用于计算本断言哈希值的密码哈希算法标识符字符串，取自C2PA哈希算法标识符列表。若该字段缺失，则采用外围结构的`alg`值作为哈希算法。若两者均存在，则使用本结构中的字段。若所有位置均无值，则该结构无效；无默认值。
   "hash": bstr, ; 哈希值的字节字符串
   ? "excluded": bool, ; 布尔值，用于指示验证者在验证过程中是否可忽略此框（及关联哈希值）。若该字段缺失，则对框进行哈希处理并比较值。
   "填充字符串": bstr, ; 用于填充空位的零填充字节串
}

box-name /= tstr .size (1..10)
```

示例框哈希中展示了五个采用CBOR诊断标记法（RFC 8949第8节）的示例：

1. JPEG;
2. PNG;
3. GIF;

4. DNG（TIFF），带子IFD；

5. TTF。

示例框哈希

```
// JPEG示例 //
{
     "alg" : "sha256", "boxes": [
          {
               "名称" : ["SOI", "APP0", "APP2"],
               "哈希" : b64'...', "填充"
               : b64'',
          },
          {
               "名称" : ["C2PA"],
               "hash" : b64'AA==',
               "填充" : b64''
          },
          {
               "名称" : ["DQT", "SOF0", "DHT", "SOS", "RST0", "RST1", "EOI"],
               "hash" : b64'',
          }
     ]
}

// PNG示例 //
// 排除XMP框 //
{
     "alg" : "sha256", "boxes": [
          {
               "names" : ["PNGh", "IHDR"],
               "hash" : b64'...', "pad"
               : b64'',
          },
          {
               "names" : ["C2PA"],
               "hash" : b64'AA==',
               "填充" : b64''
          },
          {
               "名称" : ["sBIT"],
               "哈希" : b64'',"填充" :
               b64'',
          },
          {
               "names" : ["iTXt"],
               "哈希值" : b64'...', "排除"
               : true, "填充" : b64'',
          },
          {
               "names" : ["IDAT", "IEND"],
               "hash" : b64'...', "pad"
               : b64'',
          }
     ]
}
```

```
// GIF示例 //
{
     "alg" : "sha256", "boxes": [
          {
               "names" : ["GIF89a", "LSD"]
               "hash" : b64'...', "pad"
               : b64'',
          },
          {
               "names" : ["2C", "TBID", "2C", "TBID"],
               "hash" : b64'...', "pad"
               : b64'',
          },
          {
               "names" : ["21FE"],
               "hash" : b64'...', "pad"
               : b64'',
          },
          {
               "names" : ["21F9"],
               "hash" : b64'...',"pad" :
               b64'',
          },
          {
               "names" : ["3B"],
               "hash" : b64'...', "pad"
               : b64'',
          },
     ]
}

// TIFF/DNG 示例 //
{
     "alg" : "sha256", "boxes": [
          {
               "名称" : ["TIFh", "254", "256", "257", "258", "259", "262"],
               "hash" : b64'...', "pad" :
               b64'',
          },
          {
               "名称" : ["273", "277", "278", "279", "284"],
               "hash" : b64'...', "pad" :
               b64'',
          },
          {
               // 此为包含次要图像的子IFD //
               "names" : ["330", "254", "256", "257", "258", "259", "262", "277", "278", "279",
"284"],
               "hash" : b64'...', "pad" :
               b64'',
          },
          {
               "names" : ["700", "34665"],
               "hash" : b64'...', "pad" :
               b64'',
          },
          {
               "names" : ["C2PA"],
               "hash" : b64'AA==',
               "填充" : b64''
```

```
          }
     ]
}

// TTF 示例 //
{
     "alg" : "sha256", "boxes": [
          {
               "names" : ["C2PA"],
               "哈希值" : b64'AA==',
               "填充" : b64''
          },
          {
               "名称" : ["PCLT"],
               "哈希" : b64'',"填充" :
               b64'',
          },
          {
               "names" : ["cmap"],
               "哈希" : b64'...', "填充"
               : b64'',
          },
          {
               "names" : ["cvt"],
               "hash" : b64'...', "pad"
               : b64'',
          },
          {
               "names" : ["fpgm"],
               "hash" : b64'...', "pad"
               : b64'',
          },
          {
               "names" : ["gasp"],
               "hash" : b64'...', "pad"
               : b64'',
          },
          {
               "names" : ["glyf"],
               "hash" : b64'...', "pad"
               : b64'',
          },
          {
               "names" : ["head"],
               "hash" : b64'...', "pad"
               : b64'',
          },
          {
               "names" : ["hhea"],
               "hash" : b64'...', "pad"
               : b64'',
          },
          {
               "names" : ["hmtx"],
               "hash" : b64'...', "pad"
               : b64'',
          },
          {
               "names" : ["loca"],
               "hash" : b64'...', "pad"
               : b64'',
          },
```

```
            {
                 "names" : ["maxp"],
                 "哈希" : b64'...', "填充" :
                 b64'',
            },
            {
                 "names" : ["name"],
                 "hash" : b64'...',"pad" :
                 b64'',
            },
            {
                 "names" : ["post"],
                 "hash" : b64'...', "pad" :
                 b64'',
            },
            {
                 "names" : ["prep"],
                 "hash" : b64'...',"pad" :
                 b64'',
            }
      ]
}
```

18.8. 集合数据哈希

18.8.1. 描述

在工作流中，若预先知晓C2PA清单将引用一组资产而非单个资产，则应使用集合数据哈希断言作为指定集合中资产硬绑定（即加密哈希值）的方法。

注 可通过将AI/ML模型训练数据集的每个文件夹作为完整训练数据集清单的独立组件来描述。

集合数据哈希断言应具有标签 `c2pa.hash.collection.data`。集合数据哈希断言不得出现在云数据断言中

。

18.8.2. 模式与示例

此类型的模式由以下CDDL定义中的集合数据哈希映射规则定义：

```
; URI及其关联哈希值的数组
$collection-data-hash-map /= { "uris": [1* uri-
   hashed-data-map],
   "alg": tstr .size (1..max-tstr-length), ; 用于计算`uris`数组中每个条目哈希值的密码哈希算法标识符字符串，取自C2PA哈希算法标识符列表。
      ? "zip_central_directory_hash" : bstr,
}

; 用于存储URI及其哈希值的引用数据结构。
$uri-hashed-data-map /= {
```

```
   "uri": relative-url-type, ; 相对URI引用"hash": bstr, ; 包含哈希值的字节字符
   串
   ? "size": size-type, ; 数据字节数
   ? "dc:format": 格式字符串, ; 数据的IANA媒体类型
   ? "data_types": [1* $asset-type-map], ; 数据类型的附加信息
}

; CBOR 头 (#) 和尾 ($) 已在正则表达式中引入，因此无需显式指定 relative-url-type /= tstr .regexp "[-a-zA-Z0-
9@:%._\\+~#=]{2,256}\\.[a-z]{2,6}\\b[-a-zA-Z0-9@:%_\\+.~#?&//=]*"
```

以下是一个采用CBOR诊断标记法（RFC 8949第8节）的示例：

```
// 远程URL列表示例 //
{
     "alg" : "sha256", "uris": [
          {
               "uri": "photos/id/870.jpg"
               "hash": b64'+ddHMTUUEpuSF6dNaHFa9uFc1sSnY+O3l3MMPFvX5Ws=', "dc:format": "image/jpeg"
          },
          {
               "url": "deepmind/bigbigan-resnet50/1","hash" :
               b64'...',
               "dc:format": "application/octet-stream","data_types": [
                    {
                         "type": "c2pa.types.generator",
                    },
                    {
                         "type": "c2pa.types.model.tensorflow", "version":
                         "1.0.0",
                    },
                    {
                         "type": "c2pa.types.tensorflow.hubmodule","version": "1.0.0",
                    }
               ]
          }
     ]
}

// (相对)文件URI列表示例 //
{
     "alg" : "sha256", "uris": [
          {
               "uri": "image1.png",
               "hash": b64'U9Gyz05tmpftkoEYP6XYNsMnUbnS/KcktAg2vv7n1n8='
          },
          {
               "uri": "document.pdf",
               "hash": b64'G5hfJwYeWTlflxOhmfCO9xDAK52aKQ+YbKNhRZeq92c='
          },
     ]
}

// EPUB（即 ZIP 格式）内部相对路径列表示例 //
{
```

```
    "alg" : "sha256", "uris":
    [
        {
            "uri": "mimetype"
            "hash": b64'+ZXhhbXBsZSBvZiBhIGxpc3Qgb2YgcmVsYXRpdmUgc8=','dc:format": "text/text"
        },
        {
            "uri": "META-INF/container.xml"
            "hash": b64'+ddHMTUUEpuSF6dNaHFa9uFc1sSnY+O3l3MMPFvX5Ws=','dc:format": "text/xml"
        },
        {
            "uri": "封面页.svg",
            "hash": b64'U9Gyz05tmpftkoEYP6XYNsMnUbnS/KcktAg2vv7n1n8='
        },
        {
            "uri": "chapter1.html",
            "hash": b64'G5hfJwYeWTlflxOhmfCO9xDAK52aKQ+YbKNhRZeq92c='
        },
    ]
}
```

18.8.3. 字段

uris 字段由 uri-hashed-data-map 值组成的数组构成，该数组代表一组资源集合。alg 字段的定义详见第 13.1 节“哈希算法”，将其置于此处可确保列表中所有内容项均采用相同算法进行哈希处理。

对于每个 uri-hashed-data-map，必须包含 uri 字段且该字段应为有效的相对 URI。所有 URI 均应视为相对于清单文件所在位置的相对路径，无论该位置是本地、容器（如 ZIP）内还是云端。由于相对URI可能包含导航元素（如../），存在引用与清单不在同一文件夹的内容项的风险——这将构成安全隐患。声明生成器应在使用前对URI进行验证或清理，确保URI中不包含"."或".."字符。

哈希字段是一个字节字符串，表示内容项的有效哈希值，该值由算法字段确定。
字段确定的有效哈希值。该哈希值必须覆盖内容项的所有字节（从0到n），不存在例外情况。其余字段与成

分声明中的字段完全一致。

18.8.4. 集合成员的哈希处理

集合中的每个文件均应使用alg字段中定义的特定哈希算法单独进行哈希处理。生成的哈希值应存储在与文件URI关联的uri-hashed-data-map的hash字段中。

特定层级中的文件不必全部纳入哈希集合。

注	虽然这在存在无需哈希处理的文件时很有用，但也为攻击者提供了添加文件而不破坏绑定有效性的机会。

18.9. 多资产哈希

18.9.1. 描述

存在多种由多个部分组成的文件格式，其中每个部分本身都是有效的文件格式，例如将多个独立图像聚合为单个文件的情况。具体示例包括：

- CIPA多画面格式（MPF）
- Android 超级HDR格式（采用MPF）
- ISO 21496 HDR（采用MPF）
- Android动态照片格式（不采用MPF，但可在同一文件中与MPF并存）

在某些情况下，可能需要甚至必须验证文件每个单独部分的完整性，而非仅验证文件整体。因此，当前这套硬性绑定断言不足以单独验证每个部分的完整性。此外，各个部分可能拥有各自的C2PA清单需要记录。多资产哈希断言正是为实现此功能而设计。

另一特殊情形是当某个组成部分为可选项时——即该部分可能在不涉及可信签名者的工作流中被移除——但仍需验证文件其余部分的完整性。

18.9.2. 细节说明

多资产哈希声明应采用标签 `c2pa.hash.multi-asset`。尽管其包含哈希值并修改硬性绑定处理机制，但不被视为硬性绑定。

多资产哈希声明不得出现在云数据声明中。

多资产哈希声明不应与压缩清单配合使用。

> **注** 目前尚不明确两者是否存在技术兼容性问题，因此建议在完成进一步评估前避免同时使用。

每个分段（包括主分段）均应以分段哈希映射对象形式存储于分段数组中。位置字段应包含定位器对象，用于描述分段在文件中的位置。该定位器对象应包含`bmffBox`字段或字节偏移量与长度字段。`byteOffset`字段应包含该部分在文件中的字节偏移量（从文件物理起始位置计算），`length`字段应包含该部分的字节长度。当部分作为特定BMFF框（如Motion Photo使用的mpvd）包含于主部分时，`bmffBox`字段应包含该部分的BMFF框。对于由`bmffBox`字段描述的片段，其内容仅视为该框的有效负载，不包含框头信息。

部件数组内的部件应按其在文件中出现的顺序列出，且部件应连续、不重叠，并覆盖资产的每个字节。

注	文件中的出现顺序指从字节0开始扫描至文件末尾时，各部分的连续排列顺序。

hashAssertion字段应包含指向该部件哈希断言的哈希URI。部件的哈希断言应采用标准硬绑定断言（如`c2pa.hash.data`），但标签需附加字符串`.part`及任何多重实例标识符。例如：`c2pa.hash.data.part 2`。

注	添加这些标签后缀可明确表明：部件的硬绑定断言不被视为标准硬绑定断言，因此可在C2PA清单中存在其多个实例。

可选字段应为布尔值，用于指示该部件是否为可选项——若未提供则默认为`false`。

若某个部件拥有独立的C2PA清单文件，且该文件未包含在该部件内部（例如多帧资产中的单帧），则建议将该C2PA清单存储至资产的清单存储库中，并创建`componentOf`组件进行引用。

18.9.3. 模式与示例

此类型的模式由以下CDDL定义中的`multi-asset-hash-map`规则定义：

```
multi-asset-hash-map = {
   "parts": [* part-hash-map] ; 多部分文件中各个部分对应的一个或多个哈希值组成的数组
}

byte-range-locator = (
   "byteOffset": uint           ; 该部分在文件中的字节偏移量"length": uint       ; 该部分的长度
)

; 此为CDDL特殊映射表（表示以下选项仅可存在一项）
定位器映射 = {
   byte-range-locator //             ; 文件中该部分的字节偏移量及长度"bmffBox": tstr      ; 指向该部分BMFF框的
   XPath
}

part-hash-map = {
   "location" : locator-map, ; 该部分在文件中的位置"hashAssertion": $hashed-uri-map, ; 该部分的哈希断言对应的哈希URI
   ? "optional": bool, ; 该部分是否可选且可被舍弃
}
```

以下是一个采用CBOR诊断标记法（RFC 8949第8节）的示例：

```
// 多资产哈希断言 //
// 该资产（33,333字节）包含一个字节范围[0,11111)的JPEG部分，以及另一个
// 部分（字节范围[11111,33333)）。
{
  "parts" : [
```

```
      {
        "location": { "byteOffset":
          0,
          "length": 11111
        },
        "哈希断言": "self#jumbf=c2pa.assertions/c2pa.hash.boxes.part"
      },
      {
        "位置": { "字节偏移量": 11111,
          "length": 22222
        },
        "hashAssertion": "self#jumbf=c2pa.assertions/c2pa.hash.data.part"
      }
   ]
}

// c2pa.hash.boxes.part - 资产第一部分的箱哈希值 //
{
   "alg" : "sha256",
   "boxes": [
      {
        "names" : ["SOI", "APP0", "APP2"],
        "hash" : b64'...', "pad" :
        b64'',
      },
      {
        "names" : ["C2PA"],
        "hash" : b64'AA==',
        "填充" : b64''
      },
      {
        "名称" : ["DQT", "SOF0", "DHT", "SOS", "RST0", "RST1", "EOI"],
        "hash" : b64'',
      }
   ]
}

// c2pa.hash.data.part - 资产第二部分的数据哈希值 //
{
   "alg" : "sha256",
   "填充" : '0000',
   "hash" : b64'...',
}

// c2pa.hash.boxes - 整体资产哈希值，覆盖整个两部分资产 //
{
   "alg" : "sha256",
   "boxes": [
      {
        "names" : ["SOI", "APP0", "APP2"],
        "hash" : b64'...', "pad" :
        b64''
      },
      {
        "names" : ["C2PA"],
        "hash" : b64'AA==',
        "填充" : b64''
      },
      {
        "名称" : ["DQT", "SOF0", "DHT", "SOS", "RST0", "RST1", "EOI"],
        "hash" : b64'...',
```

```
        "pad" : b64''
      },
      {
        "名称" : ["c2pa.after"],
        "哈希" : b64'...', "填充
        " : b64''
      }
    ]
}
```

此类多资产哈希断言示例可包含在图像中，如[_multi_asset_hdr_image]所示。

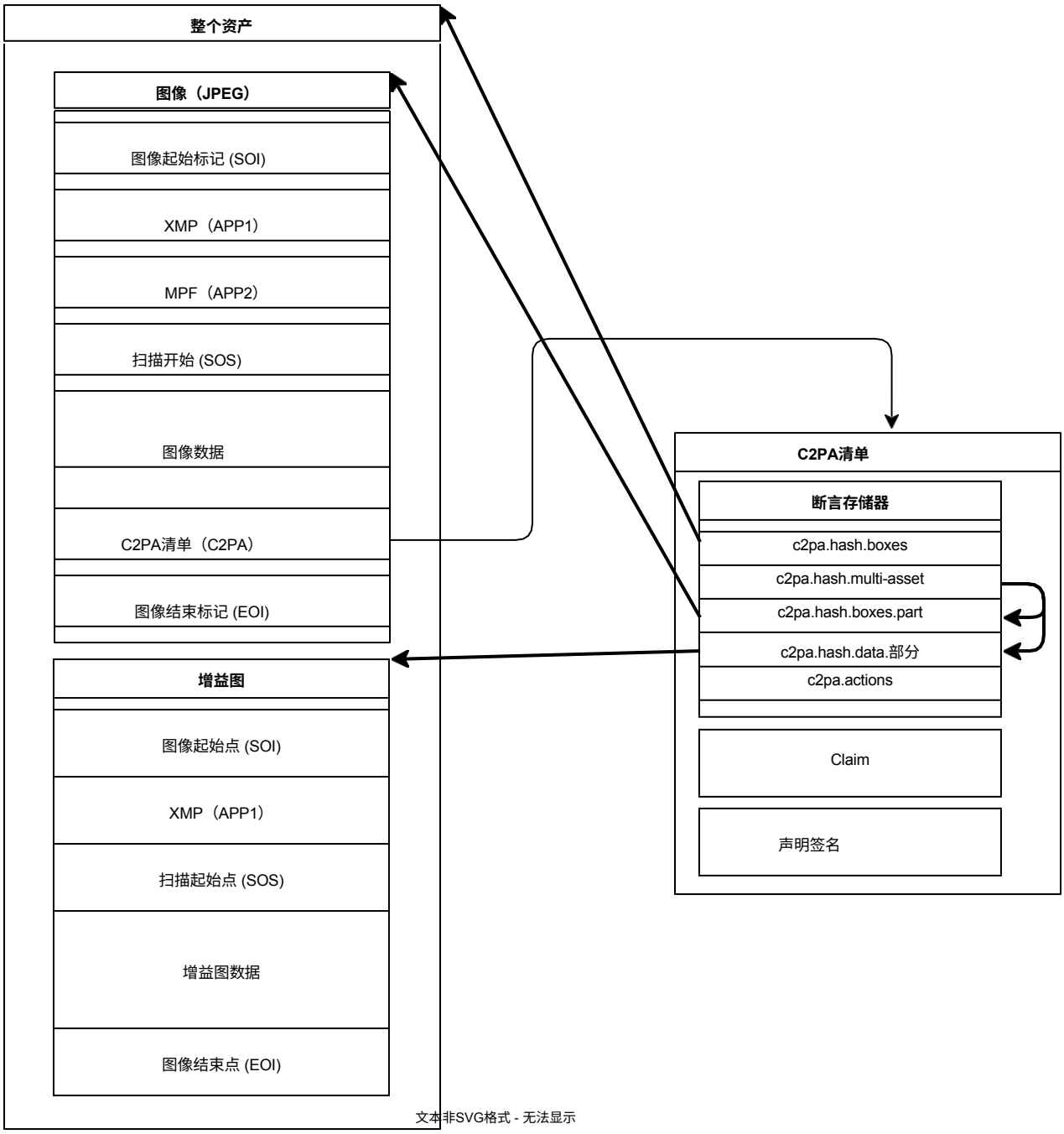

图17. 用于HDR增益图的多资产哈希声明示例

18.10. 软绑定

18.10.1. 描述

若索赔生成器将为资产内容提供软绑定，则应使用软绑定断言进行描述。此类断言中可创建并存储的软绑定类型详见第18.10节“软绑定”。

本规范的早期版本曾提供一个 `url` 字段，用于指向哈希数据的存储位置，但该字段从未被使用。现已弃用该字段，转而采用资产引用断言。声明生成器不得在软绑定断言中添加此字段，消费者在遇到该字段时应予以忽略——但根据第15.10.3节"断言验证"所述，此规定不影响该字段作为待验证内容组成部分的包含。

本规范的先前版本曾在范围字段内提供一个范围字段，用于以特定算法格式描述软绑定断言所涵盖的数字内容部分。该字段现已弃用，取而代之的是区域字段。声明生成器不得在软绑定断言中添加此字段，消费者若发现该字段应予以忽略。此变更不影响第15.10.3节"断言验证"所述内容验证过程中该字段的包含状态。

软绑定断言应采用标签 `c2pa.soft-binding`。

18.10.2. 模式与示例

该类型的模式由以下CDDL定义中的`soft-binding-map`规则定义：

```
在软绑定断言中，将感兴趣区域对象结构与用于其他目的的结构保持一致
;# 包含感兴趣区域

;用于存储资产内容部分或全部范围内一个或多个软绑定的数据结构
软绑定映射 = {
   "alg": tstr, ; 字符串标识符，用于标注计算值所采用的软绑定算法及其版本，取自C2PA软绑定算法列表。若该字段缺失，则从外围结构的`alg_soft`
值中获取算法。若两者均存在，则优先使用本结构中的字段。若所有位置均无值，则该结构无效；无默认值。
   "blocks": [1* soft-binding-block-map],
   "填充字节数组": 字节数组, ; 用于填充空间的零填充字节字符串
   ? "pad2": 字节数组, ; 可选的零填充字节字符串，用于填充空间
   ? "name": tstr .size (1..max-tstr-length), ; (可选) 描述此哈希所涵盖内容的人类可读说明
   ? "alg-params": bstr, ; (可选) 描述软绑定算法参数的CBOR字节字符串。
   ? "url": uri, ; 未使用且已弃用。
}

软绑定块映射 = { "范围": 软绑定范围映射,
   "value": bstr, ; 以算法特定格式描述的 CBOR 字节串，表示
```

```
"
}

软绑定作用域映射 = {
   ? "范围": bstr, ;已弃用, 以算法特定格式描述数字内容中软绑定值计算所依据部分的CBOR字节字符串"
   ? "时段":软绑定时段映射,
   ? "区域": region-map, ; CBOR对象定义于regions-of-interest.cddl
}

软绑定时间跨度映射 = {
   "start": uint, ; 计算软绑定值的时间范围起始点（以媒体开始时间为基准的毫秒数）
   "end": uint,        ; 软绑定值计算的时间范围结束点（以媒体开始时间为基准的毫秒数）。
}
```

以下为 CBOR 诊断标记法（RFC 8949 第 8 节）示例：

```
{
   "alg": "phash",
   "填充符": h'00',
   "url": 32("http://example.c2pa.org/media.mp4"),"blocks": [
      {
         "范围": {
            "时间跨度": {"结束时间
               ": 133016
               "start": 0,
            }
         },
         "value": b64'dmFsdWUxCg=='
      },
      {
         "scope": {
            "时间范围": {"结束时间
               ": 245009
               "start": 133017,
            }
         },

      }
   ]
}
```

18.10.3. 要求

所使用的软绑定算法应作为alg字段的值呈现，其应用范围应在blocks字段中列出。若所用算法需要额外参数，则应作为alg-params的值呈现。

范围字段可包含区域字段或时间跨度字段，用于描述软绑定计算所覆盖的数字内容片段。当存在区域字段时，其内容为区域映射对象（定义详见第18.2节"感兴趣区域"）。当存在时间跨度字段时，其描述软绑定计算所覆盖的时间区间，单位为内容起始点起算的毫秒数。

18.10.4. 软绑定算法列表

软绑定算法列表是alg字段允许取值的机器可读清单。alg字段必须与该列表中算法的alg字段相匹配。alg-params和value字段的格式因算法而异，具体描述详见列表中alg条目内通过informationalUrl引用的可读信息页。

该列表由C2PA以JSON文档形式维护，存放于：https://github.com/c2pa-org/softbinding-algorithm-list

软绑定算法列表中标记为 true 的弃用字段应视为已弃用，不得用于在清单中创建软绑定断言。标记为弃用的软绑定算法仍可用于解析软绑定，但不建议采用此行为。

软绑定算法列表条目的JSON模式如下所示：

```
{
     "$schema": "https://json-schema.org/draft/2020-12/schema", "type": "object",
     "properties":
          {"identifier": {
               "type":
               "integer","minimum": 0,
               "maximum": 65535,
               "描述": "此标识符将在软绑定算法被添加到列表时分配。"
          },
          "deprecated": { "type":
               "boolean", "default":
               false,
               "description": "指示该软绑定算法是否已弃用。
已弃用的算法不得用于创建软绑定。已弃用的算法可用于解析软绑定，但不建议采用此行为。"
          },
          "alg": {
               "type": "string",

               "描述": "实体专属命名空间，用于C2PA断言标签，需以实体的互联网域名开头，类似Java包的定义方式（例如
`com.example.algo1`、`net.example.algos.algo2`）"
          },
          "type": {
               "类型": "字符串","枚举": [
                    "水印", "指纹"
               ],
               "描述": "此算法实现的软绑定类型。"
          },
          "解码媒体类型": { "类型": "数组", "
               最小项数": 1,
               "items": {
                    "type": "string",
                    "enum": [
                         "应用程序",
```

```
                        "audio",
                        "image",
                        "模型",
                        "文本","视
                        频"
                    ],
                    "描述": "本软绑定算法适用的IANA顶级媒体类型（渲染后）。"
                }
            },
            "编码媒体类型": {"类型": "数组","
                最小项数": 1,
                "items": {
                    "type": "string",
                    "description": "本软绑定算法适用的IANA媒体类型，例如application/pdf",

\\+]+)*)$"
                }
            },
            "条目元数据": { "类型": "对象", "
                属性": {
                    "描述": { "类型": "字符串",
                        "description": "算法的人类可读描述。"
                    },
                    "dateEntered": { "type":
                        "string",
                        "format": "date-time",
                        "description": "该算法的录入日期。"
                    },
                    "联系信息": {
                        "类型": "字符串",
                        "格式": "电子邮件"
                    },
                    "informationalUrl": {
                        "type": "string",
                        "格式": "URI",
                        "description": "包含算法详细信息的网页。"
                    }
                },
                "required": [
                    "描述", "提交日期", "联系方式
                    ", "信息网址"
                ]
            },
            "软绑定解析API": {"类型": "数组",
                "items": {
                    "type": "string",
                    "格式": "URI"
                },
                "description": "支持此算法的软绑定解析API列表
算法的软绑定解析API列表。"
            }
        },
        "required": [
            "标识符", "算法",
```

```
            "类型", "条目元数据"
      ],
      "oneOf": [
            {
                  "required": [
                        "解码媒体类型"
                  ]
            },
            {
                  "required": [
                        "编码媒体类型"
                  ]
            }
      ]
}
```

以下是一个软绑定算法列表条目的 JSON 示例：

```
{
      "identifier": 1,
      "deprecated": false,
      "alg": "com.example.product", "type":
      "watermark", "decodedMediaTypes": [
            "audio",
            "video",
            "text","i
            mage"
      ],
      "条目元数据": {
            "描述": "Foo公司水印算法版本1.2","录入日期": "2024-04-23T18:25:43.511Z",
            "联系信息": "foo.bar@example.com ","信息网址":
            "https://example.com/wmdetails"
      },

      ]
}
```

算法的唯一名称存储于alg字段中，该名称对应于使用该算法的软绑定断言中alg字段所使用的字符串。名称应遵循命名空间要求，并代表算法的所有者。每个算法还被分配一个唯一的数字标识符。若提供算法的不同版本，则每个版本应在软绑定算法列表中拥有独立条目。

算法类型应为'watermark'或'fingerprint'，分别表示该算法属于不可见水印或指纹技术。

算法的弃用状态通过`deprecated`字段标注。验证器不应解析使用弃用算法的软绑定。C2PA清单不得采用弃用的软绑定进行编写。

软绑定算法列表条目应包含支持的媒体类型列表，可采用`encodedMediaTypes`或`decodedMediaTypes`格式。`decodedMediaTypes`支持的媒体类型应至少对应以下一种：

顶级IANA媒体类型包括："application"、"audio"、"image"、"model"、"text"、"video"。encodedMediaTypes支持的媒体类型应与前文所述decodedMediaType对应的至少一种已注册IANA子类型相匹配。这些IANA顶级类型及子类型详见https://www.iana.org/assignments/media-types/media-types.xhtml

软绑定算法列表中的每条条目均应在entryMetadata字段中附加补充信息。这些信息包括算法的人类可读描述（description）及其被提议纳入软绑定算法列表的日期（dateEntered）。

条目所有者的联系方式应以电子邮件地址形式提供（contact，必填）。需提供一个信息性URL（informationalUrl，必填），该URL指向描述软绑定算法特征的可读页面。该页面内容不受限制，但可包含如何解读软绑定注册库中值字段的说明（该字段采用算法特定格式编码）。

18.10.5. 软绑定解析API

软绑定解析API是一种Web API，它提供了一种标准方式，可通过软绑定值、清单标识符或资产从软绑定解析API端点检索C2PA清单存储。软绑定算法列表条目可在softBindingResolutionApis字段中包含软绑定解析API的URI列表。若提供多个URI，则可使用其中任意一个进行软绑定解析。

API规范及文档详见此处。

18.10.5.1. 验证软绑定匹配

软绑定常见用途是从清单存储库中为缺少或无效C2PA清单的资产发现有效清单。

C2PA清单的发现应通过C2PA软绑定算法列表中alg字段标识的一种或多种算法组合实现。该列表由C2PA以JSON文档形式维护，地址如下：https://github.com/c2pa-org/softbinding-algorithm-list

若在清单存储库中发现C2PA清单，且该清单包含一项或多项软绑定断言，则匹配器应确保定位清单中的所有软绑定断言与执行发现操作时使用的软绑定相匹配。

当断言中描述的算法标识符（alg）与值（value）均与执行匹配时使用的算法标识符（alg）和值（value）相符时，该软绑定断言应视为匹配。匹配操作应遵循指定算法规定的方式进行。

18.11. 云数据

18.11.1. 描述

在某些使用场景中，将断言数据存储在远程位置（如云端）比嵌入资产内部更优，尤其当数据体量较大时。对于此类场景，可采用特殊类型的断言作为数据引用的载体。出于隐私保护与可靠性考量，通过云端数据断言引用的内容应视为可选项：其内容不应作为清单验证环节的检索对象。验证者可后续调取这些内容以满足特定应用需求，例如深入追溯数据溯源历史。

若断言元数据作为另一断言的组成部分包含其中，则其同样属于云数据断言所引用的信息范畴。与其他断言类型相同，独立的断言元数据断言也可进行远程存储。

云数据断言应具有标签 `c2pa.cloud-data`。

云数据断言不得引用具有以下标签的断言：`c2pa.hash.data`、`c2pa.hash.boxes`、`c2pa.hash.collection.data`、`c2pa.hash.bmff.v2`（已弃用）或 `c2pa.hash.bmff.v3`。

18.11.2. 模式与示例

此类型的模式由以下CDDL定义中的`cloud-data-map`规则定义：

```
; 引用云端实际存储断言的断言 cloud-data-map = {
   "label": tstr, ; 云端断言的标签（例如c2pa.actions）"size": size-type, ; 数据字节数
   "location": $hashed-ext-uri-map, ; 存储云端断言的http(s)网址
   "content_type": tstr .regexp "^[-\\w.]+/[-+\\w.]+$", ; 数据的媒体/MIME类型
   ? "元数据": $断言元数据映射, ; 断言的附加信息
}

; 尺寸最小值为1，且必须为1.0的倍数 size-type = int .ge
1
```

以下是一个采用CBOR诊断标记法（RFC 8949第8节）的示例：

```
{
   "size": 98765,
   "label": "c2pa.thumbnail.claim","location": {
      "url": "https://some.storage.us/foo",
      "hash": b64'zP84FPSremIrAQHlhw+hRYQdZp/+KggnD0W8opXlIQQ='
   },
   "content_type": "application/jpeg"
}
```

18.12. 嵌入数据

18.12.1. 描述

在该规范的早期版本中，曾采用数据框（作为JUMBF框的特殊类型）的概念，以实现将数据任意嵌入C2PA清单的功能，例如用于缩略图、图标和inputTo组件。但经评估发现，通过引入新型框实现此功能会带来不必要的复杂性且存在功能缺失——例如无法对数据框进行数据遮蔽处理。因此，该概念已被弃用，转而采用标准断言机制：使用标准JUMBF嵌入文件内容类型框来承载数据。

嵌入式数据断言应具有以 `c2pa.embedded-data` 开头的标签，并遵循断言标签在多实例情况下的规则。此外，某些其他断言类型在技术上将等同于嵌入式数据断言，但会拥有各自独特的标签（例如 `c2pa.thumbnail.claim`）。

18.12.2. 技术细节

由于嵌入式数据断言基于JUMBF嵌入式文件内容类型框，其嵌入式文件描述框应包含IANA媒体类型（如`image/png`）作为*MEDIA TYPE*字段值，并可包含文件名作为*FILE NAME*字段值。该断言不得设置外部切换位。

注释	IANA 结构化 后缀 (https://www.iana.org/assignments/media-type-structured-suffix/media-类型结构化后缀（如+json和`+zip`）同样可作为*媒体类型*字段的取值。

嵌入数据声明的二进制数据框应包含文件（如光栅图像或文本提示）的二进制位，其格式由声明生成者自定义，但必须与嵌入文件描述框中指定的媒体类型相匹配。

18.13. 缩略图

18.13.1. 描述

缩略图断言提供资产生命周期中特定事件的近似视觉呈现。当前存在两个具体事件：

- 成分导入和声明创建；
- 每个事件均采用专属标签标识该断言。

18.13.1.1. 声明缩略图

对于 缩略图， 在 创建了 ，声明 创建于 时间， 该 缩略图 断言 应 具有 该 标签 `c2pa.thumbnail.claim`。C2PA清单中不得包含超过一个此标签的缩略图声明。本规范先前版本要求缩略图的IANA注册媒体类型需包含在

标签名称中包含IANA注册媒体类型（例如c2pa.thumbnail.claim.png）。该命名规范现已弃用。

18.13.1.2. 成分缩略图

导入成分时（参见第10.3.2.2节“添加成分”），应引用该成分自身清单中存储的缩略图。但某些成分可能未包含缩略图声明，甚至未包含清单。此时应为该成分生成新缩略图，并在活动清单中创建新的缩略图声明。

成分缩略图声明的标签应以c2pa.thumbnail.ingredient开头，并遵循声明标签在多重实例中的命名规则。例如，成分缩略图可能采用标签c2pa.thumbnail.ingredient 1。

本规范旧版要求缩略图标签名包含IANA注册媒体类型（如c2pa.thumbnail.claim.png）。此命名规范现已废弃。

本规范先前版本要求首个实例使用_1后缀且仅含单个下划线。当前规范为实现与所有断言的一致命名，采用c2pa.thumbnail.ingredient作为首个实例标识符，第二个实例为c2pa.thumbnail.ingredient 1，依此类推。旧命名规范现已弃用。

18.13.1.3. 技术细节

缩略图断言是一种嵌入式数据断言，但带有特殊标签以标识此特定用例。

18.14. 操作

18.14.1. 描述

操作声明提供有关编辑及其他影响资产内容的操作信息。该数组包含一系列操作记录——每项操作声明在资产上发生的内容（可选地声明发生时间），并可能包含其他信息，例如执行操作的软件。除第18.14.2节“至少包含一项操作断言的强制要求”另有说明外，该数组中操作的顺序未作规定，且不暗示操作的执行顺序。

动作断言有两个版本——原始的v1（应标记为c2pa.actions）和新改进的v2（应标记为c2pa.actions.v2）。动作基于XMP ResourceEvents进行建模，但进行了若干C2PA特有的调整。

v1动作在actions数组中完全定义。而在v2中，动作既可通过actions数组元素完整定义，也可从templates数组中同名动作元素派生而来。

对于出现在actions或templates数组中的每个动作，其action字段值应为：预定义动作（ ） 动作名称 （c2pa.resized, c2pa.edited, 等） 或 实体专属动作 动作名称

(如 `com.fabrikam.gaussianBlur` 等)。

表8"预定义操作列表"中列出了以`c2pa.`为前缀的预定义名称集：

表8. 预定义操作列表

操作	含义
c2pa.addedText	（可见）文本内容已插入资产，例如文本图层或作为标题。
c2pa.adjustedColor	调整色调、饱和度等参数。
c2pa.changedSpeed	降低或提高视频或音频轨道的播放速度
c2pa.color_adjustments	[已弃用] 调整色调、饱和度等参数
c2pa.converted	该资源的格式已被更改。
c2pa.created	该资源首次创建。
c2pa.cropped	该资产的数字内容部分区域被裁剪掉。
c2pa.deleted	该资产的数字内容部分区域已被删除。
c2pa.drawing	使用绘图工具（包括画笔或橡皮擦）进行的修改。
c2pa.dubbed	对音频进行了修改，通常涉及复合资产中的一条或多条音轨。
c2pa.edited	泛指对内容进行编辑性转换的操作。
c2pa.edited.metadata	修改资产元数据或元数据断言，但不涉及资产的数字内容。
c2pa.enhanced	应用增强处理（如降噪、多频段压缩或锐化），此类操作不属于内容编辑性转换。
c2pa.filtered	通过应用滤镜、样式等手段改变外观的操作。
c2pa.opened	现有素材已被打开，并被设置为父元素 组件。
c2pa.orientation	内容方向和位置的变更。
c2pa.placed	向资产中添加/放置一个或多个成分组件。
c2pa.published	资产已向更广泛的受众发布。
c2pa.redacted	一个或多个断言被编辑
c2pa.removed	成分中的某个组件已被移除。

c2pa.repackaged	将一种包装或容器格式转换为另一种格式。内容在不进行转码的情况下重新包装。此操作被视为对父级成分的非编辑性转换。
c2pa.resized	对内容尺寸、文件大小或两者同时进行调整
c2pa.transcoded	将一种编码转换为另一种编码，包括分辨率缩放、比特率调整和编码格式变更。此操作被视为parentOf组件的非编辑性转换。
c2pa.translated	内容语言的变更。
c2pa.trimmed	删除内容的特定时间段。
c2pa.unknown	发生了某些情况，但索赔生成器无法具体说明是什么。
c2pa.watermarked	为创建软绑定，已在数字内容中插入不可见水印。

此外，以下预定义名称集（见表9“字体操作列表”）专用于字体资源，其名称均以`font.`为前缀：

注 本规范早期版本曾将这些标记为`c2pa.font`，现已弃用该命名方式，转而采用更简洁的`font`前缀。

表9. 字体操作列表

操作	含义
font.charactersAdded	添加字符或字符集。
font.charactersDeleted	已删除的字符或字符集。
字符集修改	添加和删除的字符或字符集。
字体由可变字体创建	字体（全部或部分）从可变字体实例化。
font.edited	字体经历了未被更具体操作描述的编辑操作。
font.hinted	应用了提示处理。
字体.合并	字体是先前字体的组合。
字体.openType功能添加	字体已添加OpenType特性。
字体.OpenType特征修改	OpenType 功能已更改。
font.openTypeFeatureRemoved	从字体中移除了OpenType功能。
font.subset	字体已被精简为支持特定字符子集。

18.14.2. 必须至少包含一个动作断言

存在 必须 存在 至少 至少 一个 动作 断言 呈现 在 任一 断言 created_assertions 或 标准C2PA清单中Claim的gathered_assertions数组。此外：

- 若资产*为全新*创建（例如通过创意工具执行"文件→新建"操作、拍摄照片或视频，或由生成式AI模型生成），则声明中created_assertions或gathered_assertions数组的首个c2pa.assertions断言内，其actions数组的首个元素必须为c2pa.created操作。
    - 对于所有资产，应在c2pa.created操作中记录对应的digitalSourceType字段及其适当值，以表明资产创建时的性质。若资产创建时不包含数字内容，则digitalSourceType字段应取值http://c2pa.org/digitalsourcetype/empty。
- 若资产通过打开现有资产作为parentOf成分进行编辑而创建，则该声明中created_assertions或gathered_assertions数组的首个c2pa.actions断言中，其actions数组的首个元素应为c2pa.opened操作。配合c2pa.opened操作时无需设置digitalSourceType字段。

注意 此要求不适用于更新清单。

注意 在 gathered_assertions 中记录任何操作时，请注意这些断言不归属于签署者（参见第 10 *章"声明"*）。

C2PA清单中的完整操作断言集应包含不超过一个类型为c2pa.created或c2pa.opened的操作。若其中任一操作出现在created_assertions中，则两者均不得出现在gathered_assertions中；若其中任一操作出现在gathered_assertions中，则两者均不得出现在created_assertions中。

示例：生成式AI模型根据文本提示生成视频。生成的视频资产的活动清单将包含一个以c2pa.created动作开头的c2pa.actions断言，该动作本身具有http://cv.iptc.org/newscodes/digitalsourcetype/trainedAlgorithmicMedia值，其对应的digitalSourceType字段为 。

对应的digitalSourceType字段中。

示例：用户打开Emily移动海报制作工具创建社交媒体图片。用户选择模板后开始定制，过程中导入现有照片。生成的图像资源的活动清单将包含以下内容：- 以 c2pa.created 操作开头的 c2pa.actions 断言，且不包含 digitalSourceType 字段，表明该文件为全新创建- 针对用户导入的每张照片，均存在指向对应组件断言的 c2pa.placed 操作，其中标注了 componentOf 关系- 最后，还将记录用户执行其他操作时产生的额外操作记录

示例：某报社媒体编辑部需要编辑一张由摄影记者使用支持C2PA功能的相机拍摄的照片。媒体编辑打开照片后，对其进行了裁剪和晕影处理。最终编辑后的照片资产的活动清单中包含一个c2pa.actions断言，其中c2pa.opened操作指向原始照片的成分断言，并标注了parentOf关系。该清单还包含裁剪和晕影编辑的操作。

裁剪和晕影编辑操作。

18.14.3. 所有操作均包含在内

`actions-map-v2` 可包含一个名为 `allActionsIncluded` 的字段，该字段为布尔值。若存在 `allActionsIncluded` 且其值为 true，则声明生成器表明仅对资产执行了 actions 断言中列出的操作。若 `allActionsIncluded` 不存在或其值为 false，则清单消费者可假定存在其他未列出的操作。

18.14.4. 动作断言中的字段

18.14.4.1. 描述

操作可在描述字段中包含自由文本说明，阐述操作功能。此字段对非标准操作尤为有用，但也可用于补充标准操作的附加信息。例如，`c2pa.`edited操作可添加"画笔工具"的描述。

18.14.4.2. 原因

若存在，原因字段应包含以下标准值之一，或符合实体特定命名空间相同语法的自定义值，以说明操作依据：

- `c2pa.PII.present`;
- `c2pa.invalid.data`;
- `c2pa.`商业秘密存在;
- `c2pa.`政府机密。

注	尽管原因字段可用于任何操作，但目前仅定义了专注于编辑的 `c2pa` 值。

使用 `c2pa.redacted` 操作时，原因字段应包含编辑的依据。有关 `c2pa.redacted` 操作的其他要求，请参阅第 18.14.4.7 节“参数”。

18.14.4.3. 当

当字段中可能还包含操作发生的时间戳。若包含该字段，其值应符合CBOR日期/时间规范（RFC 8949, 3.4.1）。
字段的值应符合CBOR日期/时间规范（RFC 8949, 3.4.1）。

注	`when`字段作为简单非可信时间戳使用。建议采用基于UTC的时间格式。

18.14.4.4. 软件代理

执行操作所使用的软件或硬件可通过 `softwareAgent` 字段识别。在 v1 操作中，该字段为简单文本字符串。但在 v2 中，`softwareAgent` 使用更丰富的 `generator-info-map` 结构作为

如第10.2.3.2节“生成器信息映射”所述。当使用多个软件代理（参见第18.14.6.2节“软件代理”）时，应使用softwareAgentIndex字段通过软件代理数组中基于0的索引来引用软件代理。特定操作仅应包含一个softwareAgent或softwareAgentIndex字段。

注 当软件代理程序与声明生成程序不一致时，这些字段具有实用价值。

注 本规范早期版本曾包含actors字段，但在2.0版中已移除。

18.14.4.5. 数字来源类型

操作可包含 digitalSourceType 键，其值应为 IPTC 定义的术语之一，或下表中的 C2PA 特定值：

http://c2pa.org/digitalsourcetype/empty

数字内容实质为空的媒体，例如空白画布或零长度视频。

http://c2pa.org/digitalsourcetype/trainedAlgorithmicData

通过算法处理采样内容和数据生成的模型所得结果。区别于http://cv.iptc.org/newscodes/digitalsourcetype/trainedAlgorithmicMedia，其结果并非媒体类型（如图像或视频），而是数据格式（如CSV、pickle）。

注释 common use case for the digitalSourceType key is in conjunction with the c2pa.created 操作用于指定媒体项的生成方式——例如"数字拍摄"、"底片数字化"或"训练算法生成媒体"。

对于生成式AI等"训练算法"资产与数据，可在C2PA清单中添加一个或多个组件，以提供资产生成所需输入信息的说明。这些组件可通过c2pa.placed或c2pa.created操作进行引用，具体示例详见例8"生成式AI操作示例"。

18.14.4.6. 变更

该操作可能仅针对资产的特定部分生效——例如视频中的帧范围或图像上的特定区域。在v1版本中，该值仅为简单的文本字符串。而v2版本中，操作通过"changes"字段进行标识，其值为区域映射对象数组（具体定义参见第18.2节"感兴趣区域"）。

18.14.4.7. 参数

操作可能包含一个参数键，用于通过预定义字段以及开放式添加的任意自定义字段（及其关联值）来指定特定操作信息。自定义字段应遵循与实体命名空间相同的语法规则，例如：com.litware.someFieldName。

注 此功能可为特定工作流或C2PA清单消费者提供有价值的附加信息。

执行相同操作且参数设置恒定的声明生成器，可通过 `multipleInstances` 字段标记操作是否重复执行。若未包含该字段，则无法判断操作是否多次执行。

使用 `c2pa.opened` 或 `c2pa.placed` 操作时，参数对象中的 `ingredient` 字段（v1 版本）或 `ingredients` 字段（v2 版本）应包含指向一个或多个相关成分断言的哈希处理后的 JUMBF URI。在 `c2pa.removed` 操作中，该字段应包含指向其他清单中 `componentOf` 成分断言的哈希处理后的 JUMBF URI。在某些情况下，仅部分成分与操作相关。此时成分断言应包含断言元数据中的regionOfInterest字段，用于指定成分的相关区域（详见第18.15.13节"成分元数据"）。

注 在该规范的先前版本中，`c2pa.transcoded` 和 `c2pa.repackaged` 操作必须引用前置操作 `c2pa.opened` 操作所引用的 parentOf 元素断言；声明生成器可为此操作进行引用，以兼容旧版验证器。

使用 `c2pa.translated` 操作时，
参数对象中的 sourceLanguage 和 targetLanguage 字段必须包含符合 RFC 5646、BCP 47 标准的语言代码。

示例8. 生成式人工智能操作示例

生成式AI模型创建的图像对应的c2pa.created操作，在CBOR诊断标记法（RFC 8949第8节）中可能如下所示：

```
// 用于描述生成式AI输出的动作断言 //
{
   "actions": [
      {
         "action": "c2pa.created",

         "软件代理" : {
               "name": "Joe's Photo Editor","version": "2.0",
               "操作系统": "Windows 10"
         },
         "数字来源类型": "http://cv.iptc.org/newscodes/digitalsourcetype/trainedAlgorithmicMedia",
         "参数" : {"成分" : [
               {
                  "url": "self#jumbf=c2pa.assertions/c2pa.ingredient.v3","alg": "sha256",
                  "hash" : b64'...',
               },
               {
                  "url": "self#jumbf=c2pa.assertions/c2pa.ingredient.v3 1","alg": "sha256",
                  "hash" : b64'...',
               }
            ]
```

```
        }
      }
   ]
}
```

使用 `c2pa.redacted` 操作时，参数对象中的 `redacted` 字段应包含指向已编辑断言的 JUMBF URI。

18.14.5. 水印处理

使用 `c2pa.watermarked` 操作时，C2PA 清单中还应包含软绑定断言以描述插入的水印。

18.14.6. 操作模板

18.14.6.1. 模板

在v2操作中，模板数组的元素通过操作通用元素与特定模板值的组合进行描述。这些值由C2PA清单消费者通过

同名操作，或所有操作（当操作字段值为特殊值　时），以全面了解操作详情。若存在多个适用于同一操作的模板，则先合并模板（若存在）的值，再按模板数组中的顺序应用这些值。

操作，则值将按以下顺序合并：首先应用模板（若存在），随后按模板数组中的顺序应用其余模板。

示例9. 操作模板示例

采用CBOR诊断标记法（RFC 8949第8节）的操作与模板示例：

```
// 单一模板应用于多个动作的示例 //
{
     "actions": [
          {
               "操作": "com.joesphoto.filter", "时间点":
               0("2020-02-11T09:00:00Z")
          },
          {
               "操作": "c2pa.edited",

          },
          {
               "action": "com.joesphoto.filter","when":
               0("2020-02-11T09:20:00Z")
          },
          {
               "action": "c2pa.cropped",

          }
     ],
     "模板": [{
          "操作": "com.joesphoto.filter","描述": "魔术滤镜",
```

```
            "数字源类型": "http://cv.iptc.org/newscodes/digitalsourcetype/compositeSynthetic",
            "软件代理" : {
                  "name": "Joe's Photo Editor","version": "2.0",
                  "操作系统": "Windows 10"
            }
      }]
}

// 使用特殊全动作模板/`*` 的示例，适用于所有动作 //
{
      "actions": [
            {
                  "action": "c2pa.created", "when":
                  0("2024-03-09T20:04Z")
            },
            {
                  "action": "c2pa.edited",

            },
            {
                  "action": "c2pa.cropped",

            }
      ],
      "模板": [{
            "操作": "*","数字来源类型":
"http://cv.iptc.org/newscodes/digitalsourcetype/humanEdits","softwareAgent" :
            {
                  "name": "简的人类创作工具", "version": "1.0"
            }
      }]
}
```

C2PA清单消费者应从模板中获取值，并覆盖操作本身的值，这将导致同名项被替换。

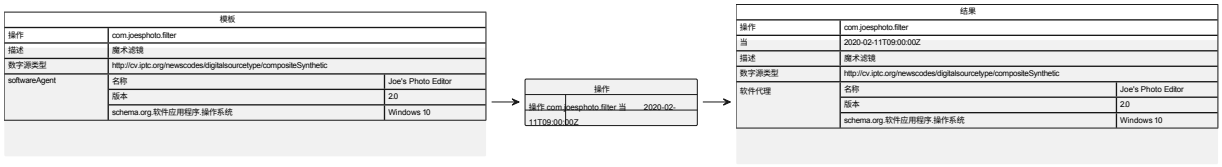

图18. 操作模板流程

模板可包含 `templatePar`ameters 键，用于添加其他任意键值对。此功能可为特定工作流或 C2PA 清单消费者提供额外有用信息。

18.14.6.2. 软件代理

若使用多个软件代理，可将其列于 `softwareAgents` 字段中。该字段为 `generator-info-map` 对象数组，每个对象描述不同的软件或硬件，可通过特定操作或模板的 `softwareAgentIndex` 字段以其索引进行引用。

示例10. 软件代理示例

跨多个操作指定多个代理的示例，采用CBOR诊断标记法（RFC 8949第8节）：

```
{
    "actions": [
        {
            "action": "com.joesphoto.magic-avatar","when": 0("2020-
            02-11T09:00:00Z"),
            "软件代理索引" : 0
        },
        {
            "action": "c2pa.edited",

            "软件代理索引" : 1
        },
        {
            "操作": "com.joesphoto.beauty-filter","时间点": 0("2020-
            02-11T09:20:00Z"),
            "软件代理索引" : 0
        },
        {
            "操作": "com.joesphoto.all-smiles", "时间点": 0("2020-02-
            11T09:40:00Z"),
            "软件代理索引" : 0
        },
        {
            "操作": "c2pa.裁剪",

            "软件代理索引" : 1
        },
        {
            "操作": "com.joesphoto.green-screen","时间点": 0("2020-
            02-11T09:50:00Z"),
            "软件代理索引" : 0
        }
    ],
    "软件代理": [
        {
            "name": "乔的人工智能过滤器",
            "version": "1.0",
            "操作系统": "Windows 10"
        }
        {
            "名称": "乔的图片编辑器","版本": "2.0",
            "操作系统": "Windows 10"
        }
    ]
}
```

18.14.6.3. 图标

模板还可包含图标——一种图像（光栅或矢量格式），可在C2PA清单消费者的用户界面中使用，以图形方式呈现操作内容。由于清单消费者将知晓所有

定义的操作中，此类图标仅应出现在实体特定操作的模板中。

若存在图标字段，其值应为哈希URI。该哈希URI应指向嵌入式数据断言或云端数据断言。若使用嵌入式数据断言，其标签应为`c2pa.icon`，并遵循断言标签关于多重实例的规则。

注	此图标字段的结构与声明生成器信息图中的图标字段完全一致。

Manifest Consumers 还应支持本规范早期版本推荐的数据框方法。

18.14.7. 本地化

若操作断言的元数据包含模板的本地化字典，则该本地化方案也应适用于基于该模板的任何操作。

18.14.8. 相关操作

当一系列操作相互关联且通常同时发生时，将它们关联起来会很有帮助。在v2操作中，关联字段提供了列出相关附加操作的位置。每个关联操作应作为主操作的子集，仅包含差异字段。与操作模板类似，C2PA清单消费者会将这些值与主操作的值合并，从而完整呈现每个关联操作的完整信息。

18.14.9. 资产呈现版本

在互联网上分发媒体时，资产版本化是常见现象。这些版本通常为满足不同网络连接、屏幕分辨率及其他环境的用户需求而创建。我们可通过动作断言帮助消费方理解特定声明创建者创建资产版本的意图。

当`c2pa.`actions声明中仅存在`c2pa.published`、`c2pa.`transcoded和`c2pa.`repackaged三项操作时，即向清单消费者传递信号：签名者声明该资产与原始素材之间未发生任何编辑性修改。

若同时存在单个"parentOf"组件，则向清单消费者进一步表明签名者声明该资产直接源自该父组件。

18.14.10. 软绑定查找

当对未包含C2PA清单的资产执行`c2pa.`opened或`c2pa.`placed操作时，声明生成器可通过软绑定查找获取该资产的C2PA清单。若查找成功，声明生成器应将定位到的C2PA清单作为ingredient断言中`activeManifest`字段的值添加。若执行此操作，则组件断言还应包含 `softBindingsMatched` 字段（值为 true）及 `softBindingAlgorithmsMatched` 字段（其数组值至少包含一项条目）。

注 添加这些字段向 C2PA 清单消费者表明使用了软绑定查找。

注 由于大多数通过软绑定恢复的清单将包含与被查询资产不匹配的硬绑定断言，因此预期将在组件断言中报告验证失败。

一个通过软绑定检索其清单的成分操作示例，采用CBOR诊断标记法（RFC 8949第8节）：

```
// 通过软绑定恢复清单的成分断言 //
{
   "dc:title": "image 1.jpg", "dc:format":
   "image/jpeg", "relationship": "parentOf",
   "softBindingsMatched": true,
   "softBindingAlgorithmsMatched": [
      "com.foo.watermark.1"
   ]
   "activeManifest": {
        "url": "self#jumbf=/c2pa/urn:c2pa:5E7B01FC-4932-4BAB-AB32-D4F12A8AA322","hash":
        b64'1kjJTO108b71cL95UxgfHD3eDgk9VrCedW8n3fYTRMk='
   },
}



// 指向成分的操作断言 //
{
   "actions": [
      {
         "action": "c2pa.opened",

         "软件代理": {
            "name": "Joe's Photo Editor","version": "2.0",
            "操作系统": "Windows 10"
         },
         "参数": {"组件": [
              {
                 "网址": "self#jumbf=c2pa.assertions/c2pa.ingredient.v3","算法": "sha256",
                 "hash": "b64'...'"
              }
           ]
         }
      }
   ]
}
```

18.14.11. 已弃用的操作

以下操作属于本规范的先前版本，现已弃用：

- c2pa.copied;

- c2pa.formatted;
- c2pa.version_updated;
- c2pa.printed;
- c2pa.managed;
- c2pa.生成;
- c2pa.保存状态。

这些属性将不再写入 c2pa.actions 或 c2pa.actions.v2 断言中，但可能出现在现有的 C2PA 清单中。

18.14.12. 模式与示例

c2pa.actions 的模式由 actions-map 规则定义，c2pa.actions.v2 的模式则由 actions-map-v2 规则定义，具体定义如下所示的 CDDL 定义：

动作的CDDL

```
actions-map = {
   "actions" : [1* action-items-map], ; 动作列表
   ? "元数据": $断言元数据映射, ; 断言的附加信息
}

$action-choice /= "c2pa.addedText"
$action-choice /= "c2pa.adjustedColor"
$action-choice /= "c2pa.changedSpeed"
$action-choice /= "c2pa.color_adjustments"
$action-choice /= "c2pa.converted"
$action-choice /= "c2pa.复制"
$action-choice /= "c2pa.创建"
$action-choice /= "c2pa.裁剪"
$action-choice /= "c2pa.deleted"
$action-choice /= "c2pa.绘图"
$action-choice /= "c2pa.配音"
$action-choice /= "c2pa.编辑"
$action-choice /= "c2pa.编辑.元数据"
$action-choice /= "c2pa.过滤"
$action-choice /= "c2pa.格式化"
$action-choice /= "c2pa.管理"
$action-choice /= "c2pa.opened"
$action-choice /= "c2pa.orientation"
$action-choice /= "c2pa.produced"
$action-choice /= "c2pa.放置"
$action-choice /= "c2pa.印刷"
$action-choice /= "c2pa.published"
$action-choice /= "c2pa.redacted"
$action-choice /= "c2pa.removed"
$action-choice /= "c2pa.重新打包"
$action-choice /= "c2pa.resized"
$action-choice /= "c2pa.保存"
$action-choice /= "c2pa.转码"
$action-choice /= "c2pa.翻译"
$action-choice /= "c2pa.trimmed"
$action-choice /= "c2pa.未知"
```

```
$action-choice /= "c2pa.版本更新"
$action-choice /= "c2pa.添加水印"
$action-choice /= "字体编辑"
$action-choice /= "字体子集"
$action-choice /= "字体.由可变字体创建"
$action-choice /= "字体字符添加"
$action-choice /= "字体字符删除"
$action-choice /= "字体字符修改"
$action-choice /= "字体提示"
$action-choice /= "字体.OpenType功能添加"
$action-choice /= "字体.OpenType特征修改"
$action-choice /= "字体.OpenType功能移除"
$action-choice /= "字体合并"
$action-choice /= tstr .regexp "([\\da-zA-Z_-]+\\.)+[\\da-zA-Z_-]+" buuid = #6.37(bstr)

; 注：本规范早期版本还包含一个"actors"字段，但在2.0版本中已移除。
action-items-map = { "action":
   $action-choice,
   ? "when": tdate, ; 操作发生的时间戳。
   ? "软件代理": tstr .size (1..max-tstr-length), ; 执行操作的软件代理。
   ? "changed": tstr .size (1..max-tstr-length), ; 以分号分隔的资源变更部分列表（相较于前次事件历史记录）。若未提供则视为未定义。当追踪变更且未知变更范围时，可假定所有内容均可能发生变更。
   ? "instanceID": buuid, ; 修改后（输出）资源的 xmpMM:InstanceID 属性值
   ? "parameters": parameters-map, ; 操作的附加参数。这些参数通常因操作类型而异
   ? "数字源类型": tstr .size (1..max-tstr-length), ; 源类型定义详见 https://cv.iptc.org/newscodes/digitalsourcetype/
}

parameters-map = {
   ? "ingredient": $hashed-uri-map, ; 该操作作用对象的组件断言哈希URI
   ? "description": tstr .size (1..max-tstr-length) ; 操作的附加描述
   * tstr => any
}


; 动作断言版本 2 (v2)

$action-reason /= "c2pa.PII.present"
$action-reason /= "c2pa.invalid.data"
$action-reason /= "c2pa.商业机密存在"
$action-reason /= "c2pa.government.confidential"
$action-reason /= tstr .regexp "([\\da-zA-Z_-]+\\.)+[\\da-zA-Z_-]+"

actions-map-v2 = {
   "actions" : [1* action-item-map-v2],                ; 操作列表
   ? "模板": [1* $action-template-map-v2], ; 动作模板列表
   ? "软件代理": [1* $生成器信息映射], ; 执行操作的软硬件列表
   ? "元数据": $断言元数据映射,                          ; 断言的附加信息
   ? "allActionsIncluded": bool                        ; 若存在且为真，表示未发生任何未包含在操作列表中的操作。
}

action-common-map-v2 = {
```

```
   ? "软件代理": $生成器信息映射, ; 执行操作的软硬件描述
   ? "软件代理索引": int, ; 动作映射2中软件代理数组的0基索引
   ? "description": tstr .size (1..max-tstr-length), ; 操作的附加描述，对自定义操作至关重要
   ? "数字源类型": tstr .size (1..max-tstr-length), ; 源类型定义详见 https://cv.iptc.org/newscodes/digitalsourcetype/ 或本
规范
}

; 注：本规范早期版本还包含一个"actors"字段，但在2.0版本中已移除。
action-item-map-v2 = {
   "action": $action-choice , ; 行动类型 action-common-map-v2, ; 新增
   通用项
   ? "when": tdate, ; 操作发生的时间戳
   ? "changes": [1* region-map], ; 资源中发生变更的关注区域列表。若未提供，则视为未定义。
   ? "related": [1* action-item-map-v2], ; 关联操作列表
   ? "reason": $action-reason, ; 执行该操作的原因，当操作为 `c2pa.redacted` 时必填
   ? "参数": 参数映射-v2 ; 操作的附加参数。这些参数通常会因操作类型而异
}

action-template-map-v2 = {
   "action": $action-choice / "*", ; 模板支持额外的特殊"*"选项 action-common-map-v2, ; 附加通用项
   ? "icon": $hashed-uri-map, ; 指向嵌入式数据断言的哈希URI引用
   ? "templateParameters": parameters-common-map-v2 ; 模板的附加参数。
}

parameters-common-map-v2 = (
   * tstr => any
)

parameters-map-v2 = {
   ? "redacted": $jumbf-uri-type, ; 指向编辑后断言的JUMBF URI，当操作为`c2pa.redacted`时必填
   ? "成分": [1* $hashed-uri-map], ; 指向该操作所作用的成分（v2 或 v3）断言的哈希化 JUMBF URI 列表
   ? "sourceLanguage": tstr .size (1..max-tstr-length), ; `c2pa.translated` 操作的源语言 BCP-47 编码
   ? "目标语言": tstr .size (1..max-tstr-length), ; `c2pa.translated`操作的目标语言BCP-47编码
   ? "multipleInstances": bool, ; 此操作是否多次执行parameters-common-map-v2, ; 来自通用参数的任何内
   容
}
```

字体资源专属的标准操作详见：

字体操作的CDDL规范

```
; 字体专属动作的映射、范围及参数

; 多重字体操作基于Unicode值范围生效。font-unicode-range-map = {
   "start": uint, ; 包含起始值 "stop": uint, ; 包
   含结束值
}
```

```
; 字体参数，用于 font.subset、font.charactersAdded、
; font.charactersDeleted 和 font.charactersModified 使用的字体参数。
font-parameter-unicode-ranges-map = {
   "ranges": [1* font-unicode-range-map] ; Unicode 范围数组
}

; 字体实例化参数的范围
字体粗细范围 = 1..1000 ; 字体的有效粗细值。400为常规值。字体宽度范围 = 0.0..1000.0 ; 相对于常规宽度的百分比，范围0%至1000%。
100%为常规宽度。
font-slant-range = -90.0..90.0 ; 倾斜角度，0度表示无倾斜。

; 创建可变字体实例时使用的字体参数。
; 此处详述字体的不同'变体轴'。各轴名称的标记
; 不同轴的名称在每个参数的注释中用括号标注
;
字体参数由可变字体映射创建 = {
   ? "weight": 字重范围, ; 待实例化字体的字重(wght)或粗细度。
   ? "width": 字体宽度范围, ; 待实例化字体的字符宽度(wdth)或窄度。
   ? "italic": bool, ; 获取字体的斜体(ital)版本。
   ? "倾斜度": 字体倾斜度范围, ; 字体的倾斜角度(slnt)。
   ? "optical-size": int / float, ; 字体的光学尺寸(opsz)，通常需匹配请求的字号。
   * tstr → any ; 自定义坐标轴的名称和类型。
}
```

示例 11. v2 操作示例

以下为 v2 动作示例，采用 CBOR 诊断标记法（RFC 8949 第 8 节）：

```
{
   "actions": [
      {
         "action": "c2pa.filtered",

         "参数": {

         },
         "软件代理" : {
               "名称": "乔的图片编辑器","版本": "2.0",
               "操作系统": "Windows 10"
         }
      },
      {
         "操作": "c2pa.裁剪",

      }
   ],
   "元数据": {

      "reviewRatings": [
         {
            "value": 1,
            "说明": "内容绑定未通过验证"
         }
      ]
   }
```

```
}
```

18.15. 成分

18.15.1. 描述

当资产组合时（例如在图像编辑工具中将图像置入图层，或在视频编辑工具中将音频片段置入视频），必须将原始资产的声明信息记录至新资产中，以便追溯组合资产的完整历史。此原则同样适用于利用现有资产创建衍生资产或资产呈现形式的情形。

成分的另一种常见用途是描述某些作为流程输入使用的资产或数据，例如与AI/ML模型相关的训练或推理请求。

成分声明共有三个版本：原始的v1（标签为`c2pa.ingredient`）、改进的v2（标签为`c2pa.ingredient.v2`）以及进一步优化的v3（标签为`c2pa.ingredient.v3`），后者解决了在编辑后验证成分的问题。

注： 由于通常存在多个成分声明，采用单调递增的版本标识符可确保唯一性。增加索引在标签将是用于（例如：`c2pa.ingredient.v3`、`c2pa.ingredient.v3__1`、`c2pa.ingredient.v3__2`）。

18.15.2. 建立唯一标识符

若添加的成分包含C2PA清单，则其唯一标识符应取自包含该成分有效C2PA清单的JUMBF超级包装箱清单标签，此时无需在成分声明中提供可选的`instanceID`字段。当声明生成器提供成分断言的可选`inst`anceID字段时，唯一标识符的值应按第8.3节"识别非C2PA资产"的规定确定。

注 即使成分具有 C2PA 清单，声明生成器仍可在成分声明中提供 `instanceID` 字段。

18.15.3. 关系

添加配料时，需描述其与当前产品的关联关系。关联关系字段的可能取值及其含义详见表10“配料关联关系”。

关系字段的可能值及其含义详见表10“配料关系”。

表10. 配料关系

值	含义
父级关系	当前资产是该组件的衍生资产或资产呈现形式。此关系值也用于更新清单。
componentOf	当前资产由多个部件组成，该组件是其中之一。
输入源	该组件作为计算过程（如AI/ML模型）的输入，导致了该资产的创建或修改。

添加成分断言时，声明生成器应在活动清单中添加c2pa.actions断言（参见第18.14节"操作"），若该断言尚未存在。根据成分类型，应向c2pa.actions断言的操作数组添加下列新增条目之一：

- 当添加具有parentOf关系的成分时，应向actions数组中添加一个c2pa.opened动作。
- 当添加具有 componentOf 关系的成分时，应向此要求仅适用于标准清单，因为仅该清单类型支持记录操作。

此要求仅适用于标准清单，因为仅该清单类型支持记录操作。

18.15.4. 标题

若存在，dc:title的值应为该成分的人类可读名称，该名称可取自资产的XMP属性，或取自本地/远程（如云端）文件系统中资产的名称。若成分未指定具体名称，则可使用成分描述替代。

18.15.5. 格式

若存在，dc:format的值应为该组件的IANA媒体类型。建议声明生成器提供此字段，且其值必须有效。描述多文件组件（如AI/ML模型的数据集）时，dc:format字段应设置为multipart/mixed。

18.15.6. 模式与示例

该类型的CDDL定义为：

```
; 描述资产所用组件的断言 ingredient-map = {
   "dc:title": tstr, ; 成分名称
   "dc:format": 格式字符串, ; 元素的媒体类型
   ? "documentID": tstr, ; 元素`xmpMM:DocumentID`的值 "instanceID": tstr, ; 唯一标识符，例如元素的
`xmpMM:InstanceID`的值
```

```
    "relationship": $relation-choice, ; 该成分与其所属资产之间的关系。
    ? "c2pa_manifest": $hashed-uri-map, ; 指向成分C2PA清单的哈希URI引用
    ? "thumbnail": $hashed-uri-map, ; 成分缩略图的哈希URI引用
    ? "validationStatus": [1* $status-map] ; 元素的验证状态
    ? "元数据": $断言元数据映射 ; 断言的附加信息
}

; 组件断言版本 2 (v2)
; 描述资产所用成分的断言 ingredient-map-v2 = {
    "dc:title": tstr, ; 成分名称
    "dc:format": format-string, ; 成分的媒体类型
    "relationship": $relation-choice, ; 该成分与其所属资产的关系
    ? "documentID": tstr, ; 元素的 `xmpMM:DocumentID` 值
    ? "instanceID": tstr, ; 唯一标识符，例如该组件的
`xmpMM:InstanceID` 的值
    ? "data" : $hashed-uri-map / $hashed-ext-uri-map, ; 指向嵌入数据断言的哈希URI或指向外部数据的哈希扩展URI
    ? "data_types": [1* $asset-type-map], ; 向组件V2结构补充数据类型的附加信息。
    ? "c2pa_manifest": $hashed-uri-map, ; 指向该组件C2PA清单的哈希URI引用
    ? "thumbnail": $hashed-uri-map, ; 嵌入数据断言中缩略图的哈希URI引用
    ? "validationStatus": [1* $status-map] ; 元素的验证状态
    ? "description": tstr .size (1..max-tstr-length) ; 成分的附加描述
    ? "informational_URI": tstr .size (1..max-tstr-length) ; 指向成分或其数据信息页面的URI
    ? "元数据": $断言元数据映射 ; 断言的附加信息
}

; 版本 3 (v3) 的成分声明
; 描述资产所用成分的声明 ingredient-map-v3 = {
    ? "dc:title": tstr, ; 成分名称
    ? "dc:format": format-string, ; 元素的媒体类型
    "relationship": $relation-choice, ; 该成分与其所属资产的关系
    ? "validationResults": $validation-results-map, ; 声明生成器对成分资产执行完整验证的结果
    ? "instanceID": tstr, ; 唯一标识符，例如该组件的
`xmpMM:InstanceID` 的值
    ? "data" : $hashed-uri-map / $hashed-ext-uri-map, ; 指向嵌入式数据断言的哈希URI或指向外部数据的哈希扩展URI
    ? "dataTypes": [1* $asset-type-map], ; 向成分V3结构补充数据类型的附加信息
    ? "activeManifest": $hashed-uri-map, ; 指向该组件活动清单对应框的哈希URI
    ? "声明签名": $哈希URI映射, ; 指向组件C2PA清单中声明签名框的哈希URI
    ? "thumbnail": $hashed-uri-map, ; 指向嵌入数据断言中缩略图的哈希URI引用
    ? "description": tstr .size (1..max-tstr-length), ; 成分的附加说明
    ? "informationalURI": tstr .size (1..max-tstr-length), ; 指向该成分或其数据信息页面的URI
    ? "softBindingsMatched": bool, ; 是否匹配软绑定
    ? "softBindingAlgorithmsMatched": [1* tstr] ; 用于发现活动清单的算法名称数组
? "softBindingsMatched": bool, ; 是否匹配软绑定
```

```
   ? "metadata": $assertion-metadata-map ; 断言的附加信息
}

format-string = tstr .regexp "^\\w+\/[-+.\\w]+$"

; 描述该成分与资产关联原因的选择项
$relation-choice /= "parentOf"
$relation-choice /= "componentOf"
$relation-choice /= "inputTo"
```

CBOR诊断标记法示例（RFC 8949第8节）：

```
{
   "dc:title": "image 1.jpg", "metadata": {

      "reviewRatings": [
         {
            "value": 5,
            "explanation": "内容绑定已验证"
         }
      ]
   }
   "dc:format": "image/jpeg","thumbnail": {
        "url": "self#jumbf=c2pa.assertions/c2pa.thumbnail.ingredient","hash":
        b64'UjRAYWiAq4lfCRDmksWAlDJN/XtHHFFwMWymsZsm3j8='
   },
   "relationship": "parentOf", "activeManifest":
   {
        "url": "self#jumbf=/c2pa/urn:c2pa:5E7B01FC-4932-4BAB-AB32-D4F12A8AA322","hash":
        b64'1kjJTO108b71cL95UxgfHD3eDgk9VrCedW8n3fYTRMk='
   },
   "claimSignature": {


   },
   "验证结果": { "活动清单": {
        "成功": [
        {
            "code": "claimSignature.validated",

        },{
            "code": "签名凭证受信任",

        },{
            "code": "timeStamp.validated",

        },{
            "code": "时间戳已验证",

        },{
            "code": "assertion.hashedURI.match",
```

```
D4F12A8AA322/c2pa.assertions/c2pa.ingredient.v3"
        }
        ],
        "informational": [{
           "code": "签名凭证OCSP查询跳过",

        }],
        "failure": []
      },
      "成分差异": [{
        "成分断言URI": "self#jumbf=/c2pa/urn:c2pa:5E7B01FC-4932-4BAB-AB32-D4F12A8AA322/c2pa.assertions/c2pa.ingredient.v3",
        "验证差异": { "成功": [],
           "informational": [],
           "failure": [{
              "code": "assertion.hashedURI.mismatch",

        }]
        }
      },{
        "成分断言URI": "self#jumbf=/c2pa/urn:c2pa:F095F30E-6CD5-4BF7-8C44-CE8420CA9FB7/c2pa.assertions/c2pa.ingredient.v3",
        "验证差异": { "成功": [],
           "informational": [],
           "failure": [{
              "code": "签名凭证不可信",

        }]
        }
      }
      ]
   }
}
```

18.15.7. 描述字段

描述字段可包含自由文本描述，说明该成分的性质或用途。当标题或格式描述不足以说明时，此字段尤为实用。

18.15.8. 成分数据

18.15.8.1. 标准用法

在某些使用场景（如生成式人工智能）中，提供成分数据至关重要——数据可嵌入C2PA清单或通过URL引用。这通过成分中的数据字段实现：使用哈希URI指向嵌入式数据断言，或使用哈希外部URI指向外部引用。

注	本规范旧版本允许哈希URI指向数据框。

使用嵌入式数据断言意味着其内容将被嵌入此C2PA清单及任何未来包含该资产作为成分的C2PA清单（除非被删除）。声明生成器在选择是否嵌入数据时应考虑该字段的大小。

示例12. 含数据的组件示例

部分含数据的组件示例，采用CBOR诊断标记法（RFC 8949第8节）：

```
// 提示框的数据区域 //
{
   "dc:format": "text/plain",
   "data" : '肩上坐着鸟的海盗'"dataTypes": [{
      "type": "c2pa.types.generator.prompt",
   }]
}

// 成分（提示） //
{
   "dc:title": "提示词", "dc:format":
   "text/plain", "relationship":
   "inputTo", "data": {
      "url": "self#jumbf=c2pa.assertions/c2pa.embedded-data", "alg" :
      "sha256",
      "哈希值" : b64'...',
   }
}


// 成分（模型） //
{
   "dc:title": "模型",
   "dc:format": "application/octet-stream","dataTypes": [
      {
         "type": "c2pa.types.generator",
      },
      {
         "type": "c2pa.types.model.tensorflow", "version":
         "1.0.0",
      },
      {
         "type": "c2pa.types.tensorflow.hubmodule","version":
         "1.0.0",
      }
   ],
   "关系": "输入到","数据": {
      "url": "https://tfhub.dev/deepmind/bigbigan-resnet50/1?tf-hub-format=compressed","alg" : "sha256",
      "hash" : b64'...',
   },
   "description": "基于ImageNet训练的无监督BigBiGAN图像生成与表示学习模型，采用更小型的编码器架构（ResNet-50）。",
   "informationalURI": "https://tfhub.dev/deepmind/bigbigan-resnet50/1",
}
```

在某些使用场景中，识别成分数据的信息至关重要，但既无法嵌入数据也无法提供有效URL——例如描述私有/内部AI模型的使用情况。对于此类情况，可通过data_types字段的值提供资产类型，以更清晰地说明该数据的格式与描述。

示例13. 带data_types字段的成分示例

一个未包含哈希URI的成分示例，采用CBOR诊断标记法（RFC 8949第8节）：

```
// 元素（私有模型） //
{
      "dc:title": "模型",
      "dc:format": "application/octet-stream","relationship": "inputTo",
      "dataTypes": [
            {
                  "type": "c2pa.types.generator",
            },
            {
                  "type": "c2pa.types.model.tensorflow","version":
                  "1.5.0",
            }
      ],
      "描述": "Joe的私有生成式AI模型","信息URI": "https://www.example.com/joes-model-info.html"
}
```

18.15.8.2. 多文件组件

某些情况下，成分可能由多个文件组成，例如AI/ML模型的训练数据集。在此类情形中，建议在成分声明中包含C2PA清单，且完整数据集的C2PA清单需包含资产引用声明，以指明这些文件的存储位置。

注意 此方法特别适用于处理资产集合的情境，其中所有文件并不位于相同的层级结构中。

18.15.9. 信息性URI

当需要提供包含成分信息的网页URL（例如AI/ML模型的详细信息）时，应将其设置为成分断言中informationalURI字段的值。

注 信息URI并非指向成分内容本身的认证链接，而是更普遍地供人类用户参考的信息。

注 旧版（已弃用）版本的成分断言命名为此字段信息性URI。

18.15.10. 缩略图

添加成分时，附带该成分的缩略图有助于记录导入时的状态。为此，应将缩略图作为缩略图断言添加，并通过哈希URI引用在本文件中进行引用。

清单消费者还应支持本规范早期版本推荐的数据框方法。

18.15.11. 现有清单

18.15.11.1. 通用条款

若组件已存在C2PA清单存储库，则该存储库中所有经过验证且尚未存在于资产C2PA清单存储库的C2PA清单，均应由声明生成器复制至资产C2PA清单存储库，但第18.15.12节所述情况除外，“复制现有清单”条款所述情况，或存在明确禁止操作的指令（例如通过用户输入或配置实现）。

声明生成器还应将任何未通过验证的附加C2PA清单，以及出现在C2PA清单存储中但未被识别为C2PA清单的任何附加JUMBF框和超级框，复制到资产的C2PA清单存储中。

注	复制这些附加元素可支持自定义断言及未来功能的兼容性。 构造的兼容性，这些构造可能以声明生成器无法识别的形式引用C2PA清单存储中的元素。

18.15.12. 复制现有清单

18.15.12.1. 确定需求

为确定是否需要将现有清单从成分的C2PA清单存储库复制到资产的C2PA清单存储库，声明生成器应：

1. 根据第18.15.12.4节"成分验证"所述流程验证该成分。若验证失败，声明生成器可根据指令（例如通过用户输入或配置）跳过后续步骤。
2. 针对原料C2PA清单存储库中的每份清单，将其URN标识符与资产C2PA清单存储库中已存在的每份C2PA清单的URN标识符进行比对。
    a. 若发现匹配项，则计算并比较来自成分C2PA清单存储库的清单框哈希值与来自资产C2PA清单存储库的匹配清单框哈希值。
        i. 若哈希值匹配，则声明生成器不得将清单从组件的C2PA清单存储复制至资产的C2PA清单存储。
        ii. 若哈希值不匹配：

A. 声明生成器应检查两个清单中的任何断言是否被编辑（可选地使用在执行显式验证过程中编译的编辑列表）。

I. 若验证器能确定哈希值差异仅源于编辑操作，则：

1. 若所有编辑操作均已应用于资产C2PA清单存储库中现有的清单，则声明生成器不得将组件C2PA清单存储库中的清单复制至资产C2PA清单存储库。
2. 若所有编辑操作均应用于组件清单存储中的清单，则声明生成器应将资产C2PA清单存储中的清单替换为组件C2PA清单存储中的清单。
3. 若对来自成分C2PA清单存储库的C2PA清单与资产C2PA清单存储库的清单分别应用了不同的编辑规则，则声明生成器应从资产C2PA清单存储库中的现有清单中编辑尽可能多的断言，以使两组编辑结果形成并集。

II. 在所有其他情况下，声明生成器应从成分的C2PA清单存储库复制清单，根据《唯一标识符》所述流程使用更新后的URN重新标记该清单，并将重新标记的版本插入资产的C2PA清单存储库。

> 判定哈希值差异是否仅因编辑所致的过程由验证者自行决定。

注

18.15.12.2. 示例

示例：假设索赔生成器D正在导入原料。它首先导入原料B，该原料本身包含原料清单A。在验证两份清单后，索赔生成器D将清单B和A复制到资产D的C2PA清单存储库。随后它导入原料C，该原料同样包含经过编辑的清单A版本。在验证清单C及编辑版清单A后，它对比两个清单A版本的哈希值。鉴于原料C中的清单A版本经过编辑，声明生成器D将资产D的C2PA清单存储库中原有的清单A版本覆盖为原料C提供的编辑版清单A。

示例：假设上述场景中，组件C中的清单A因某项断言的哈希值比对失败而验证失败。此时声明生成器D将从组件C复制清单A，为其重新标记新的统一资源名称（URN），并将重新标记的副本存入资产D的C2PA清单存储库。

注

> C2PA清单存储库可包含非C2PA清单的JUMBF盒或超级盒，此类对象无需在该流程中复制。

18.15.12.3. 向组件断言添加清单引用

若组件的活动清单已复制至资产的C2PA清单存储库，则应将指向该组件活动C2PA清单盒的URI引用存储为组件断言中activeManifest字段的值，并额外将指向活动清单C2PA声明签名盒的URI引用存储为claimSignature字段的值。

对于存在于C2PA清单存储库中的C2PA清单，应使用`hashed_uri`作为组件断言中`activeManifest`和`claimSignature`字段的值。

注 同时提供这两个值可实现高效的组件验证，并在组件断言之一被编辑时仍支持验证。

18.15.12.4. 成分验证

18.15.12.4.1. 通用要求

此外，当成分声明引用C2PA清单时，声明生成器还应充当验证器，按照验证步骤对成分进行验证。该验证结果——包括所有成功代码、信息代码和失败代码——应按下文所述填入成分声明的validationResults或validationStatus字段。该字段必须存在，以便用于未来的验证。

注 当验证状态（成分v2）或验证结果（成分v3）存在失败状态时，视为声明生成器明确声明：参与方已确认成分C2PA声明本身存在验证错误，并选择继续执行
纳入该成分。

如第15.3节“显示清单信息”所述，声明生成器应突出显示警告，以便使用具有缺陷来源历史记录的资产的操作者能够在知情情况下决定如何继续操作。

18.15.12.4.2. v2组件断言（已弃用）

在缺少c2pa_manifest字段的v2组件断言中，validationStatus字段为可选项，但若存在则可包含空数组。

在包含c2pa_manifest字段的v2组件断言中，validationStatus数组中的每个对象均包含一个code字段，其值描述清单特定部分的验证状态；同时可选的success字段通过布尔值指示代码反映的是成功（true）还是失败（false）。若需额外的人类可读说明，可在explanation字段中提供验证状态的可选描述。此外，每个status-map对象均包含url字段。当状态为失败时，该字段应包含指向清单中特定元素的JUMBF URI引用。根据代码类型，该网址将指向 C2PA声明、C2PA声明签名或特定C2PA声明。状态代码定义详见第15.2.2节"标准状态代码"。

当声明生成器需要记录特定流程状态信息时，允许使用自定义状态码。该代码应遵循实体特定命名空间的语法规则（例如 com.litware.malformedFrobber），且验证状态对象必须包含一个成功布尔值。

18.15.12.4.3. v3 成分断言

在未包含 activeManifest 字段的 v3 成分断言中，不得存在 validationResults 字段。在包含 activeManifest 字段的 v3 成分断言中，validationResults 字段应包含一个 validation-results-map对象，该对象包含：

1. 在 activeManifest 中，包含该成分有效清单的完整验证结果。
2. 在 ingredientDeltas 中，包含该成分在 C2PA 清单存储库中每个清单里所有含 activeManifest 字段的成分断言的增量验证结果。成分断言的增量验证结果应包含以下内容：
   a. ingredientAssertionURI：该成分断言的URI。
   b. 在验证增量中，应包含成分断言所引用的清单的验证结果，同时省略成分断言中验证结果字段的activeManifest字段（或v1/v2版本成分断言中的validationStatus字段）中存在的任何状态值。此状态值比较应考虑状态类型（成功、信息性或失败）、代码和URL，忽略其他字段。

示例：考虑具有复杂血统的多成分清单E。声明生成器E通过成分断言将清单C和清单D添加为成分。清单C本身通过组件断言添加了清单A和清单B。清单D也通过组件断言添加了清单A。在添加清单C时，声明生成器E创建了一个包含验证结果对象的组件断言，其中activeManifest存储了C的活动清单验证结果，ingredientDeltas存储了清单A和B的增量验证结果。成分差异数组包含两个元素：其一对应于清单C中对清单B成分断言的验证结果对象（通过哈希URI链接指向清单C中的该成分断言）与activeManifest对象的差异结果；其二具有相同属性，但对应于清单C中对清单A成分断言的验证结果。添加清单D时同样如此：声明生成器E创建的成分断言将存储：- Manifest D的activeManifest验证结果- ingredientDeltas数组（仅含单个元素），该元素存储与Manifest D中对Manifest A成分断言的activeManifest对象进行比对的差异验证结果。

注 虽然这是一个刻意设计的示例，但旨在阐明验证结果数据结构的使用预期。

每个验证结果（通过状态码映射描述）包含成功码、信息码和失败码的数组。每个码均以状态映射对象表示，该对象必须包含存储状态码的code字段，还可包含指向具体JUMBF URI的url字段。

清单中与状态相关的元素，以及一个可选的说明字段，用于提供状态的人类可读解释。状态代码在第15.2.2节“标准状态代码”中定义。

当声明生成器需要记录特定流程状态信息时，允许使用自定义状态码。该代码应遵循实体特定命名空间的相同语法（例如 `com.litware.malformedFrobber`）。

18.15.13. 成分元数据

如断言元数据所述，成分断言的元数据字段可包含该成分的相关元数据，例如生成日期与时间，或其他有助于清单消费者判断断言数据来源或真实性的信息。

元数据字段的一个常见用途是在创建或编辑资产时仅使用了原料的部分区域。此类情况下，元数据字段应包含一个regionOfInterest字段（详见第18.3.6节“感兴趣区域”），用于描述所使用的原料相关部分。示例14“含区域元数据的原料示例”展示了此类用法。

注	尽管该字段仅包含单个感兴趣区域，但区域映射对象可指定多个区域作为其`region`字段的值。当单个素材涉及多个部分时，此特性尤为实用。

示例14. 含区域元数据的配料示例

以下为元数据中包含感兴趣区域的配料示例，采用CBOR诊断标记法（RFC 8949第8节）：

```
{
  "dc:title": "someVideo.mp4", "metadata": {
    "regionOfInterest" : {
         "描述": "10秒音频", "区域": [
              {
                   "type": "temporal", "time": {
                        "type": "npt",
                        "start": "10",
                        "结束时间": "20"
                   }
              },
              {
                   "type": "已识别","item": {
                        "标识符": "track_id", "值": "3"
                   }
              }
         ]
    }
  }
  "dc:format": "video/mp4",
```

```
    "关系": "组成部分","活动清单" : {
         "url": "self#jumbf=/c2pa/urn:c2pa:98782815-5116-4d78-93de-3f5d8b4f4615",
         "hash":b64'TEWww2UCIR/e8mmR0XvzkFVZYTJ59Q8Ip4nkYxrS/Ys='
    },
    "claimSignature" : {

         "hash": b64'ICJkYzpmb3JtYXQiOiAiaW1hZ2UvanBlZyIsCiAgImR='
    },
    "验证结果": { ... }
}
```

18.15.14. 软绑定

一个活动清单可能包含一个通过软绑定查找发现的C2PA清单作为其组成部分（通过parentOf关系关联）。若声明生成器确实包含此类C2PA清单，则应包含一个`softBindingsMatched`字段（值为true），以及一个`softBindingAlgorithmsMatched`字段（该字段包含一个字符串数组，其中列出了用于发现该组成部分C2PA清单的软绑定算法名称）。算法名称应与C2PA软绑定算法列表中的名称一致，该列表条目中的`alg`字段即为其标识。

18.16. 元数据

18.16.1. 描述

在该规范的早期版本中，每个元数据标准（如IPTC、EXIF）都有独立的断言。本版本引入了用于表示元数据的断言类别，采用标准化序列化格式。将元数据封装于断言中，意味着该元数据具有重要性——因其已被明确纳入C2PA清单，并由特定签名者签署，从而实现数据的加密验证与归属追溯。此外，通过采用通用序列化格式，清单消费者能够以一致的方式进行处理。

注	这些断言可以代表现有标准，也可以是私有规范。

18.16.2. 通用要求

元数据断言应包含以字符串 `.metadata` 结尾的标签，其前缀可采用标准 `c2pa` 标识符，或任何符合实体特定命名空间相同语法的标识符。例如 `com.litware.metadata` 断言即为有效格式。

每个元数据断言应包含单个JSON内容类型框，内含一个或多个元数据值的JSON-LD序列化形式。JSON-LD对象内必须包含@context属性，用于为所指定的元数据标准提供上下文/命名空间。创建此JSON-LD对象的推荐流程是：先构建元数据的XMP数据模型表示，再依据XMP的JSON-LD序列化规范进行序列化，最终将生成的JSON-LD存储为JSON内容类型框。

18.16.3. c2pa.metadata断言

本规范定义了一个元数据断言，其标签为c2pa.metadata，用于表示可在任何C2PA清单中使用的通用元数据模式子集。该断言中可包含的元数据字段详见附录B《*c2pa.metadata实现细节*》。

注	自定义标签的元数据断言可包含来自任意模式的值。

示例15. 图像的c2pa.metadata断言

图像的 c2pa.metadata 断言示例：


```
{
      "@context" : {
            "exif": "http://ns.adobe.com/exif/1.0/", "exifEX":
            "http://cipa.jp/exif/2.32/", "tiff":
            "http://ns.adobe.com/tiff/1.0/",
            "Iptc4xmpExt": "http://iptc.org/std/Iptc4xmpExt/2008-02-29/", "photoshop" :
            "http://ns.adobe.com/photoshop/1.0/"
      },
      "photoshop:DateCreated": "2022年8月31日", "Iptc4xmpExt:DigitalSourceType":
"http://cv.iptc.org/newscodes/digitalsourcetype/digitalCapture", "exif:GPSVersionID":
      "2.2.0.0",
      "exif:GPSLatitude": "北纬39.21.102",
      "exif:GPSLongitude": "西经74.26.5737",
      "exif:GPSAltitudeRef": 0, "exif:GPSAltitude":
      "100963/29890","exif:GPSTimeStamp": "18:22:57",
      "exif:GPSDateStamp": "2019:09:22",
      "exif:GPSSpeedRef": "K", "exif:GPSSpeed":
      "4009/161323","exif:GPS图像方向参考": "T",
      "exif:GPS图像方向": "296140/911", "exif:GPS目标方位
      参考": "T", "exif:GPS目标方位": "296140/911",
      "exif:GPS定位误差": "13244/2207", "exif:曝光时间": "1/100",
      "exif:光圈值": 4.0,
      "exif:ColorSpace": 1,
      "exif:DigitalZoomRatio": 2.0, "tiff:Make": "相机
      公司", "tiff:Model": "Shooter S1",
      "exifEX:LensMake": "相机公司",
      "exifEX:LensModel": "17.0-35.0 mm",
      "exifEX:镜头规格": { "@list": [ 1.55, 4.2, 1.6, 2.4 ] }

}
```


示例16. PDF 的c2pa.metadata 断言

PDF文件的c2pa.metadata断言示例：


```
{
```

```
    "@context" : {
       "dc" : "http://purl.org/dc/elements/1.1/", "xmp" :
       "http://ns.adobe.com/xap/1.0/","pdf" :
       "http://ns.adobe.com/pdf/1.3/", "pdfx":
       "http://ns.adobe.com/pdfx/1.3/"
    },
    "dc:created": "2015年2月3日","dc:title": [
          "这是一个测试文件"
    ],
    "xmp:CreatorTool": "TeX", "pdf:Producer": "pdfTeX-
    1.40.14", "pdf:Trapped": "未知",
    "pdfx:PTEX.Fullbanner": "这是 pdfTeX，版本 3.1415926-2.5-1.40.14 (TeX Live 2013) kpathsea 版本 6.1.1"
}
```


18.16.4. `c2pa.metadata` 文件的编辑

尽管编辑过程的设计使得只能对整个断言进行编辑（参见第6.8节“断言编辑”），但通过使用更新清单，可实现部分编辑——即移除原始版本，并在更新清单中放置新的精简版本。该新断言将在用户界面中与更新清单的签署者关联呈现，而非与已被编辑的C2PA清单签署者关联。

例如，若需对包含位置数据和摄像头信息的元数据断言进行位置数据编辑，可通过更新清单创建仅含摄像头信息的新元数据断言实现。

18.17. 时间戳

18.17.1. 描述

在某些来源追溯工作流中，标准清单或更新清单是在离线状态下创建的，此时无法在签名时从可信时间戳机构（TSA）获取符合RFC 3161标准的可信时间戳。然而，此类签名证书将在一定时间后失效，导致C2PA清单失效。

为防止此类失效情况，可在后续时间点（前提是证书尚未过期）添加可信时间戳，为该C2PA清单（以及活跃清单关联的资产）提供"存在证明"。此时间戳断言用于为这类C2PA清单提供可信时间戳。

18.17.2. 模式与示例

此类型的模式由以下CDDL定义中的时间戳映射规则定义：

```
; 用于存储一组
; 映射到时间戳"数据块"的显式统一资源名称
$time-stamp-map /= {
```

```
   * $$时间戳条目 => 字符串
}

时间戳条目 /= tstr .regexp "^urn:c2pa:[\\da-zA-Z_-]+$"
```

以下是一个采用CBOR诊断标记法（RFC 8949第8节）的示例：

```
{
   "urn:c2pa:d61c74e0-ce26-4439-b92d-690dcce6b58e" : h'...',
   "urn:c2pa:ab8c2751-8711-455a-9a8b-37143bfc92c2" : h'...'
}
```

18.17.3. 要求

时间戳断言应具有 `c2pa.time-stamp` 标签，且每个 C2PA 清单中最多只能包含一个时间戳断言。

时间戳断言包含一个CBOR映射（定义为时间戳映射），该映射至少包含一个键值对（定义为时间戳条目）。键应为被加盖时间戳的C2PA清单的C2PA清单URN（如本文所定义），值应为CBOR字节字符串，其内容如下文所述。

每个时间戳条目的值应与从符合RFC 3161标准的时间戳机构(TSA)（RFC 3161）通过分离内容模式接收到的TimeStampResp结构中`timeStampToken`字段所含的二进制数据完全一致。获取`TimeStamp`Resp的过程应遵循第10.3.2.5.3节"获取时间戳"所述流程，但需注意：有效负载值应为待时间戳的C2PA清单中C2PA声明签名框所含COSE_Sign1_Tagged结构的签名字段值。

18.18. 证书状态

18.18.1. 描述

在某些来源工作流中，标准清单或更新清单是在离线状态下创建的，此时无法在签名时获取撤销信息（通过OCSP）。由于验证过程中无法获取该信息，验证者可能需要联网来确定证书的撤销状态。本断言通过事后添加信息，为这类C2PA清单提供可信证书状态。

18.18.2. 模式与示例

该类型的模式由以下CDDL定义中的`cert-status-map`规则定义：

```
certificate-status-map = { "ocspVals": [1*
   bstr]
}
```

以下是一个采用CBOR诊断格式（.cbordiag）的示例：

```
{
  "ocspVals" : [ h'...',
    h'...'
  ]
}
```

18.18.3. 要求

证书状态断言应具有 c2pa.certificate-status 标签，且 C2PA 清单最多包含一个证书状态断言。

证书状态断言由一个CBOR映射（定义为certificate-status-map）构成，并应在ocspVals数组中至少包含一个条目。如第14.5.2节"证书吊销"所述，声明生成器查询签名证书所指的OCSP服务，捕获响应，并应将其存储为与存储在rVals头部ocspVals数组元素时相同的二进制格式（参见示例3"rVals的CDDL"）。

18.19. 资产引用

18.19.1. 描述

此声明用于标识可获取资产副本的一个或多个位置。每个位置均应通过资产引用声明进行描述。位置应通过统一资源标识符（URI）表达。该URI可指向单个资产，也可引用目录。后者用于提供资产集合的位置，该集合将通过集合数据哈希进行哈希处理。

注	使用URI表达方式可灵活获取来自网络位置或分布式存储的资产。文件系统 例如 及 IPFS （后者详见 https://docs.ipfs.tech/how-to/address-ipfs-on-web/#subdomain-gateway）。

资产引用断言应具有标签 c2pa.asset-ref。

断言元数据中的时间戳为判断所描述链接（即引用对象）的新鲜度提供了依据。

18.19.2. 模式与示例

此类型的模式由以下CDDL定义中的asset-ref-map规则定义：

```
;资产引用断言（ARA）描述了获取资产副本的位置。asset-ref-map = {
  "references": [1* ara-reference-block-map]
}
```

```
ara-reference-block-map = { "reference": ara-
   reference-uri-map,
   ? "description": tstr, ; 位置的人类可读描述。
}

ara-reference-uri-map = {
   "uri": tstr, ; URI 指向可获取资产副本的位置
}
```

以下是一个采用CBOR诊断标记法（RFC 8949第8节）的示例：

```
{
   "引用": [
      {
         "description": "该资产在网上的副本","reference": {
               "uri": "https://some.storage.us/foo"
            }
      },
      {
         "描述": "IPFS上的资产副本","引用": {
               "uri": "ipfs://cid"
            }
      }
   ]
}
```

18.20. 资产类型

18.20.1. 描述

资产类型断言提供了一种更完整描述资产的方式，特别是提供了关于如何解析或处理资产的额外上下文信息。由于许多资产的格式无法仅通过单一媒体类型值完全描述，该断言允许指定IANA媒体类型值和/或额外的类型信息。

资产类型断言应采用标签 `c2pa.asset-type.v2`。每个 C2PA 清单中最多只能包含一个资产类型断言。

注 本标准早期版本曾记录`c2pa.asset-type`断言，该断言现已弃用。

若存在，`dc:format`字段的值应为该资源的IANA媒体类型。

若存在，types字段的值应为零个或多个映射数组（`asset-type-map`），用于指定与该资产关联的类型。此映射中 `type` 字段的值应来自表 11“资产类型值”，或使用实体特定命名空间（例如 `com.litware.types.abc`），并符合第 6.2.2 节“标签命名”中定义的断言标签语法。若适用，资产版本（如数据集或模型的版本）可记录在资产类型映射的版本字段中。

注

随着C2PA被采用以在未来为AI/ML（即人工智能/机器学习）资产提供溯源信息，C2PA清单可嵌入模型和数据集资产中，并使用资产类型断言来指定这些模型和数据集资产的类型。

表11. 资产类型值

C2PA类型	资产的C2PA类型描述
c2pa.types.dataset	可由多个AI/ML框架处理的数据集，或未被其他值描述的数据集
c2pa.types.dataset.jax	JAX数据集
c2pa.types.dataset.keras	Keras 数据集
c2pa.types.dataset.ml_net	ML.NET 数据集
c2pa.types.dataset.mxnet	MXNet 数据集
c2pa.types.dataset.onnx	ONNX 数据集
c2pa.types.dataset.openvino	OpenVINO 数据集
c2pa.types.dataset.pytorch	PyTorch 数据集
c2pa.types.dataset.tensorflow	TensorFlow 数据集
c2pa.types.model	未被其他模型类型描述的AI/ML模型
c2pa.types.model.jax	JAX模型
c2pa.types.model.keras	Keras 模型
c2pa.types.model.ml_net	ML.NET模型
c2pa.types.model.mxnet	MXNet 模型
c2pa.types.model.onnx	ONNX 模型
c2pa.types.model.openvino.parameter	OpenVINO模型参数
c2pa.types.model.openvino.topology	OpenVINO模型拓扑
c2pa.types.model.pytorch	PyTorch 模型
c2pa.types.model.tensorflow	TensorFlow 模型
c2pa.types.numpy	使用序列化的 NumPy 格式存储
c2pa.types.protobuf	使用协议缓冲格式存储
c2pa.types.pickle	使用Python pickle格式存储
c2pa.types.savedmodel	使用 TensorFlow SavedModel 格式存储

18.20.2. 模式与示例

此类型的架构由以下CDDL定义中的`asset-types`规则定义：

```
; 资产类型断言提供了一种描述资产类型或格式的方式，
; 特别是提供额外上下文信息说明如何解析或处理该资产。
; 也可用于描述外部引用或关联资产，如AI/ML模型。

$type-choice /= "c2pa.types.classifier"
$type-choice /= "c2pa.types.cluster"
$type-choice /= "c2pa.types.dataset"
$type-choice /= "c2pa.types.dataset.jax"
$type-choice /= "c2pa.types.dataset.keras"
$type-choice /= "c2pa.types.dataset.ml_net"
$type-choice /= "c2pa.types.dataset.mxnet"
$type-choice /= "c2pa.types.dataset.onnx"
$type-choice /= "c2pa.types.dataset.openvino"
$type-choice /= "c2pa.types.dataset.pytorch"
$type-choice /= "c2pa.types.dataset.tensorflow"
$type-choice /= "c2pa.types.format.numpy"
$type-choice /= "c2pa.types.format.protobuf"
$type-choice /= "c2pa.types.format.pickle"
$type-choice /= "c2pa.types.generator"
$type-choice /= "c2pa.types.generator.prompt"
$type-choice /= "c2pa.types.generator.seed"
$type-choice /= "c2pa.types.model"
$type-choice /= "c2pa.types.model.jax"
$type-choice /= "c2pa.types.model.keras"
$type-choice /= "c2pa.types.model.ml_net"
$type-choice /= "c2pa.types.model.mxnet"
$type-choice /= "c2pa.types.model.onnx"
$type-choice /= "c2pa.types.model.openvino"
$type-choice /= "c2pa.types.model.openvino.parameter"
$type-choice /= "c2pa.types.model.openvino.topology"
$type-choice /= "c2pa.types.model.pytorch"
$type-choice /= "c2pa.types.model.tensorflow"
$type-choice /= "c2pa.types.regressor"
$type-choice /= "c2pa.types.tensorflow.hubmodule"
$type-choice /= "c2pa.types.tensorflow.savedmodel"
$type-choice /= tstr .regexp "([\\da-zA-Z_-]+\\.)+[\\da-zA-Z_-]+"

asset-type-map = {
   "type": $type-choice, ; 列出的选项之一或自定义值
   ? "version": tstr .regexp "^(0|[1-9]\\d*)\\.(0|[1-9]\\d*)\\.(0|[1-9]\\d*)(?:-((?:0|[1-

]*))*))?(?:\\+([0-9a-zA-Z-]+(?:\\.[0-9a-zA-Z-]+)*))?$"
}

asset-types = {
   ? "dc:format": format-string, ; 资产的IANA媒体类型
   ? "类型": [* 资产类型映射表], ; 与资产相关的类型集合
   ? "metadata": $assertion-metadata-map ; 断言的附加信息
}
```

以下是一个采用CBOR诊断标记法（RFC 8949第8节）的示例。在此示例中，资产为版本2.11.0的TensorFlow模型文件，该文件以SavedModel格式存储。

```
{
   "types":
   [
      {
         "type": "c2pa.types.model.tensorflow",
```

```
        "version": "2.11.0"
      },
      {
        "type": "c2pa.types.savedmodel","version":
        "2.11.0"
      }
    ]
}
```

18.20.3. 关于类型值选择的详细说明

若资产的精确类型已在IANA注册表应用类型或IANA注册表文本类型中指定（包括JSON、CSV和XML类型），则该信息应包含在资产类型断言的`dc:format`字段中。

> 例如，若资产为CSV格式文本文件，则`dc:format`字段应为`text/csv`。

资产类型断言可能同时包含都柏林核心格式与C2PA标准或自定义资产类型，以提供关于资产类型的补充信息。表12“常见DC格式”中列出了现有都柏林核心类型中，常与其他资产类型组合用于资产类型断言的类型。

表12. 常见DC格式

dc:format 值	资产的都柏林核心类型描述
application/json	采用JSON格式存储
application/gzip	采用GZIP格式存储
application/vnd.rar	使用RAR格式存储
application/zip	使用ZIP格式存储
application/octet-stream	使用任意二进制格式存储
text/csv	使用CSV格式存储
text/plain	使用纯文本格式存储
text/tab-separated-values	使用制表符分隔值 (TSV) 文本格式存储
text/xml	使用XML格式存储

C2PA声明的`dc:format`字段还支持IANA结构化后缀（如+json和`+zip`）以指定其他类型。

某些dc:format类型虽被广泛使用，但未在IANA注册表中明确规定。以下`dc:format`
这些值适用于C2PA资产，如表13“附加格式”所示。

表13. 附加格式

dc:format 值	资产的都柏林核心描述类型
application/x-hdf5	采用HDF5格式存储
application/x-7z-compressed	采用7Z格式存储

18.21. 深度图

18.21.1. 描述

深度图断言提供相机所捕获场景的3D描述。深度图断言可能包含预先计算的深度图，或可供下游采集/查看软件后续计算深度图的数据（例如左右立体图像）。

所有深度图断言的标签均应以`c2pa.depthmap`开头，并跟随第三部分标识深度图类型。

C2PA深度图断言应通过光学方式采集，而非通过机器学习模型等手段从单张二维图像推断得出。

18.21.2. GDepth 深度图

GDepth深度图断言采用成熟的GDepth格式编码预计算深度图。该断言标签应为`c2pa.depthmap.GDepth`。

此断言中存储的数据模式必须始终与 https://developers.google.com/depthmap-metadata/reference 中的模式完全一致。

注 当 GDepth 数据超过 64KB 时无需分割，该限制源于 XMP 为适应 `APP1` 段大小限制而设。

18.21.3. 模式与示例

此类型的架构由以下CDDL定义中的`depthmap-g`depth-map规则定义：

```
; 编码捕获场景的 GDepth 格式 3D 深度图的断言depthmap-gdepth-map = {
  "GDepth:Format": 格式类型, ; 描述如何将深度图数据转换为有效浮点深度图的格式。当前有效值为'RangeInverse'和'RangeLinear'
  "GDepth:Near": 浮点数, ; 深度图的近端值（以深度单位表示）"GDepth:Far": 浮点数, ; 深度图的远端值（以深度单位表
  示）
  "GDepth:Mime": mime-type, ; 描述深度图像内容的base64字符串的MIME类型，例如'image/jpeg'或'image/png'"
  "GDepth:Data": base64编码字符串类型, ; 基64编码的深度图像。详见developers.google.com上的GDepth编码页面。深度图将被拉伸以适应对应的
彩色图像
  ? "GDepth:Units": 单位类型,              ; 深度图的单位，例如'm'表示米或'mm'表示毫米
```

```
   ? "GDepth:MeasureType": depth-meas-type, ; 深度测量类型。当前有效值为'OpticalAxis'和'OpticRay'
   ? "GDepth:ConfidenceMime": confidence-mime-type, ; 描述置信度图像内容的base64字符串的MIME类型，例如'image/png'。",
   ? "GDepth:Confidence": base64字符串类型,                    ; 经base64编码的置信度图像。详见developers.google.com上的
GDepth编码页面。置信度图应与深度图尺寸一致
   ? "GDepth:Manufacturer": tstr .size (1..max-tstr-length), ; 生成此深度图的设备制造商
   ? "GDepth:Model": tstr .size (1..max-tstr-length), ; 生成此深度图的设备型号
   ? "GDepth:Software": tstr .size (1..max-tstr-length), ; 生成此深度图的软件
   ? "GDepth:ImageWidth": 浮点数, ; 此深度图关联的原始彩色图像的像素宽度。此值并非深度图宽度。若存在该属性，应用程序在缩放、裁剪或旋转彩
色图像时必须更新此属性。客户端使用此属性验证深度图相对于彩色图像的完整性
   ? "GDepth:ImageHeight": float, ; 此深度图关联的原始彩色图像像素高度。此值非深度图高度。若存在该属性，应用程序在缩放、裁剪或旋转彩色
图像时必须更新此属性。客户端使用此属性验证深度图相对于彩色图像的完整性
   ? "metadata": $assertion-metadata-map, ; 断言的附加信息
}

base64-string-type = tstr

$mime-choice /= "image/jpeg"
$mime-choice /= "image/png"

mime-type = $mime-choice .default "image/jpeg"confidence-mime-type =
$mime-choice .default "image/png"

$格式选择 /= "RangeInverse"
$格式选择 /= "线性范围"

格式类型 = $格式选择 .默认 "范围反向"

; 单位可表示为米（"m"）或毫米（"mm"）
$单位选择 /= "m"
$单位选择 /= "毫米"
unit-type = $unit-choice .default "m"

$深度测量选项 /= "光轴"
$深度测量选择 /= "光路"
深度测量类型 = $深度测量选择 .默认值 "光轴"
```

以下为CBOR诊断标记法（RFC 8949第8节）示例：

```
{
   "GDepth:Far": 878.7,
   "GDepth:Data": "hoOspQQ1lFTy/4Tp8Epx670E5QW5NwkNR+2b30KFXug=","GDepth:Mime":
   "image/jpeg",
   "GDepth:Near": 29.3,
   "GDepth:Model": "CameraCompany Shooter S1","GDepth:Units": "mm",
   "GDepth:格式": "范围反转",
   "GDepth:软件版本": "Truepic Foresight 固件 适用于 QC QRD8250 v0.01", "GDepth:置信度":
   "acdbpQQ1lFTy/4Tp8Epx670E5QW5NwkNR+2b30KFXug=", "GDepth:图像宽度": 32.2,
   "GDepth:图像高度": 43.6
```

```
  "GDepth:测量类型": "光轴", "GDepth:制造商": "相机公司",
  "GDepth:置信度MIME": "image/png",
}
```

根据 GDepth 规范定义，所有 GDepth 深度图断言中必须包含以下字段：

- GDepth:Format;
- GDepth:Near;
- GDepth:远距;
- GDepth:Mime;
- GDepth:Data.

18.22. 字体信息

18.22.1. 描述

字体信息声明用于确保基本字体元数据（如名称、格式、创建者归属及许可信息）以可通过密码学验证的方式添加至资源中。

字体信息断言应具有`font.info`标签，且每个清单中最多只能包含一个字体信息断言。

18.22.2. 模式与示例

该类型的模式由以下CDDL定义中的`font-info-`map规则定义：

```
; font.info断言的数据结构。font-info-map = {
  "fullName": tstr, ; 字体的完整名称。
  ; 语义化版本控制（semver）格式的版本号。
  ? "version": tstr .regexp "^(0|[1-9]\\d*)\\.(0|[1-9]\\d*)\\.(0|[1-9]\\d*)(?:-((?:0|[1-
]*))*))?(?:\\+([0-9a-zA-Z-]+(?:\\.[0-9a-zA-Z-]+)*))?$",
  ? "versionUrl": ext-url-type, ; 指向该字体版本相关发布说明的URL。
  ? "releaseDate": tdate, ; 该字体版本的发布日期。 "familyName": tstr, ; 字体家族名称。
  "style": $font-style, ; 字体样式，例如斜体或常规体。 "weight": font-weight-map, ; 字体粗细，包含名称与数值
  。
  ; 字体'name'表中的PostScript名称，ID为6。
  "postScriptName": tstr .regexp "^(?!.*[\\[\\]\\(\\)\\{\\}<>\\/%])[!-~]{1,63}$", ; 来自 ASCII 33-126 的字符，但以下字符
除外：[](){}<>/%
  "格式": $字体格式选择, ; 此字体的格式。"版权声明": tstr, ; 此字体的相关版权信息。
  ? "copyrightHolder": font-entity-map, ; 字体版权持有实体。
  ? "copyrightYears": [1* font-copyright-year-range], ; 版权持有者主张版权的年份范围。
  ? "设计者": [1* 字体设计者映射], ; 设计该字体的个人。
 ? "设计厂商": 字体厂商映射, ; 设计该字体的厂商。
```

```
   ? "sourceFoundry": 字体实体映射, ; 分发该字体的厂商。
   ? "标识符": tstr, ; 字体厂商或供应商使用的内部标识符。
}

; 字体格式
$font-format-choice /= "TrueType"
$font-format-choice /= "OpenType"

; 版权年份范围
字体版权年份范围 = 1..9999

; 字体粗细范围
font-weight-range = 1..1000

; 字重类别描述符
$font-weight-class /= "Microline"
$字体粗细类别 /= "发丝线"
$字重类别 /= "超细"
$字体粗细类别 /= "超细"
$font-weight-class /= "Thin"
$font-weight-class /= "超轻"
$font-weight-class /= "超轻"
$字体粗细类 /= "轻"
$font-weight-class /= "中细"
$font-weight-class /= "书体"
$font-weight-class /= "Normal"
$font-weight-class /= "常规"
$font-weight-class /= "Medium"
$font-weight-class /= "DemiBold"
$font-weight-class /= "SemiBold"
$字体粗细类 /= "粗体"
$font-weight-class /= "Heavy"
$字体粗细类 /= "特粗"
$字体粗细类 /= "超粗体"
$字体粗细类 /= "半黑体"
$字体粗细类 /= "粗体"
$字体粗细类 /= "特黑"
$字体粗细类 /= "超粗体"
$font-weight-class /= "MegaBlack"

; 字体样式
$字体样式 /= "常规"
$字体样式 /= "斜体"
$font-style /= "斜体"
$font-style /= "罗马体"
$字体样式 /= "常规"

; 字重数据font-weight-map = {
   "class": $font-weight-class, ; 字重类别的描述性名称，例如粗体或细体。
   "value": font-weight-range, ; 字重值。
}

; 具有名称和凭据的实体数据 font-entity-map = {
   "name": tstr, ; 个人或铸造厂名称。
   ? "url": ext-url-type, ; 该人员或字体厂商的附加信息网址。
}

; 字体设计师数据 font-designer-map =
```

```
{
  "person": font-entity-map, ; 字体设计者信息。
```

```
  ? "foundry": font-entity-map, ; 设计师参与字体设计时所属的字体厂商名称。
  ? "contribution": tstr, ; 设计师对字体所作贡献的描述。例如，'所有拉丁字符和阿拉伯字符'。
  ? "startDate": tdate, ; 设计师开始参与字体设计的日期。
  ? "endDate": tdate, ; 设计师结束字体设计贡献的日期。
}
```

以下为仅包含必填字段的CBOR诊断标记法（RFC 8949第8节）基本示例：

```
{
     "fullName": "示例二斜体", "familyName": "示例二
     ", "style": "斜体",
     "weight": {
          "class": "Regular", "value": 400
     },
     "postScriptName": "示例二斜体","format": "TrueType",
     "copyrightNotice": "版权所有 2011 示例二项目作者 (https://www.example.com/lifonts/Example-Two)，保留字体名称'示例
二'。",
     "版权持有者": {"名称": "Fabrikam"
     },
     "设计者": [
          {
               "人员": {
                  "姓名": "约翰·多伊",
                  "网址": "https://fabrikam.example.com/jdoefonts"
               }
          }
     ]
}
```

此扩展示例还演示了可选字段：

```
{
     "fullName": "示例字体粗体斜体", "version": "7.0.4-
     beta",
     "versionUrl": "https://fabrikam.example.com/release/efbi/7.0", "familyName": "示例字
     体",
     "style":
     "Italic","weight": {
          "class": "Bold",
          "value": 700
     },
     "postScriptName": "示例字体-粗体斜体", "format": "OpenType",
     "copyrightNotice": "© 2017 Fabrikam, Inc. 保留所有权利。","copyrightHolder": {
          "name": "Fabrikam Inc."
     },
     "版权年份": [ 1982,
          2017
     ],
     "设计师": [
```

```
            {
                  "人员": {
                     "姓名": "约翰·多伊",
                     "网址": "https://fabrikam.example.com/browse/designers/john-doe"
                  },
                  "字体厂商": {
                      "名称": "Fabrikam Fonts"
                  },
                  "贡献内容": "连字符。"
            },
            {
                  "人员": {
                     "姓名": "简·多伊"
                  },
                  "字体厂商": {
                       "name": "Fabrikam Fonts"
                  },
                  "贡献内容": "全部字符。"
            }
      ],
      "设计工坊": {
            "name": "Fabrikam Fonts",
            "网址": "https://fabrikam.example.com"
      },
      "字体源": {
            "名称": "Fonts Direct 2 U","网址":
            "https://fd2u.example.com"
      },
      "identifier": "示例字体粗体斜体 (Fabrikam)"
}
```

第19章. 专利政策

C2PA通过W3C专利模式（2004）采纳了开放标准专利政策：

许可承诺。对于工作组开发的源代码或数据集以外的材料，每位工作组参与者均同意根据W3C专利政策（详见http://www.w3.org/Consortium/Patent-Policy-20040205）所定义的必要专利主张，依据W3C RF许可要求第5节（http://www.w3.org/Consortium/Patent-Policy-20040205）在该工作组通过的批准交付物中提供，视同该批准交付物为W3C建议标准。工作组开发的源代码须遵循工作组章程中规定的许可条款。

关于排除条款。在将可交付成果草案作为批准交付成果采纳之前，工作组参与者可通过向工作组主席提交书面意向通知（"排除通知"），将其在本协议下的许可承诺中排除必要权利要求。针对已授权专利及已公布申请的排除通知，须列明工作组参与者拟排除于本专利政策第1节许可承诺之外的每项已授权专利或待审专利申请的专利号（或标题）及申请号（视具体情况而定）。若可能包含必要权利要求的已授权专利或待审专利申请未在排除通知中列明，则该等必要权利要求仍须受本协议许可承诺约束。针对未公开专利申请的排除通知须提供以下任一内容：(i) 已提交申请的文本；或 (ii) 明确指出交付成果草案中使被排除权利要求成为必要权利要求的具体部分。若选择(ii)，排除效力仅限于交付成果草案中已指明的部分。工作组主席将发布排除通知。

附录A：嵌入清单

A.1. 支持的格式

C2PA清单作为该资产C2PA清单存储库的一部分嵌入资产中。

嵌入C2PA清单存储库时，其位置会因资产类型或格式而异。以下是常见文件格式及其对应的C2PA清单存储库位置：

JPEG

更多信息请参阅第 A.3.1 节“将清单嵌入 JPEG”。

JPEG-XL

更多信息请参阅第A.3.8节"将清单嵌入JPEG XL"。

PNG

更多信息请参阅第 A.3.2 节“将清单嵌入 PNG”。

SVG

有关更多信息，请参阅第A.3.3节“将清单嵌入SVG”。

FLAC

更多信息请参阅第 A.3.4 节“将清单嵌入 ID3”。

MP3

更多信息请参阅第 A.3.4 节“将清单嵌入 ID3”。

GIF

更多信息请参阅第 A.3.7 节“将清单嵌入 GIF”。

DNG

更多信息请参阅第 A.3.5 节“将清单嵌入基于 TIFF 的资产”。

基于TIFF的格式

更多信息请参阅第 A.3.5 节“将清单嵌入基于 TIFF 的资产”。

WAV 和 BWF

更多信息请参阅第 A.3.6 节“将清单嵌入基于 RIFF 的资源”。

AVI

更多信息请参阅第 A.3.6 节“将清单嵌入基于 RIFF 的资产”。

WebP

更多信息请参阅第 A.3.6 节“将清单嵌入基于 RIFF 的资产”。

其他基于RIFF的格式

更多信息请参阅第 A.3.6 节“将清单嵌入基于 RIFF 的资产”。

字体

更多信息请参阅第 A.3.9 节“将清单嵌入字体”。

PDF

有关更多信息，请参阅第A.4节“将清单嵌入PDF文件”。

EPUB

更多信息请参阅第 A.6 节“将清单嵌入基于 ZIP 的格式”。

OOXML

更多信息请参阅第 A.6 节“将清单嵌入基于 ZIP 的格式”。

Open Document

更多信息请参阅第 A.6 节“将清单嵌入到基于 ZIP 的格式中”。

OpenXPS

更多信息请参阅第 A.6 节“将清单嵌入基于 ZIP 的格式”。

其他基于ZIP的格式

有关更多信息，请参阅第A.6节“将清单嵌入基于ZIP的格式”。

MP4

更多信息请参阅第 A.5 节“将清单嵌入基于 BMFF 的资源”。

MOV

更多信息请参阅第 A.5 节“将清单嵌入基于 BMFF 的资产”。

AAC

更多信息请参阅第 A.5 节“将清单嵌入基于 BMFF 的资产”。

ALAC

更多信息请参阅第 A.5 节“将清单嵌入基于 BMFF 的资产”。

HEIF

有关更多信息，请参阅第A.5节“将清单嵌入基于BMFF的资产”。

其他基于BMFF的格式

参见第A.5节“将清单嵌入BMFF资产”中指定的框。

注

正在考虑添加到本规范中的非 BMFF 音频格式包括 Ogg Vorbis 和原生无损音频编解码器（Native FLAC）的原生容器版本。

A.2. 在多部分资产中嵌入清单

在将C2PA清单嵌入多部分资产（"多资产"）时，应在资产的主部分（包含活动清单）中嵌入C2PA清单存储库，但其他部分也可包含各自的C2PA清单存储库。主部分的活动清单应包含多资产哈希断言，该断言描述资产内每个部分的位置和哈希值，并应说明整个多部分资产的来源。

A.3. 将清单嵌入非BMFF格式资产

A.3.1. 嵌入清单至JPEG

C2PA清单存储应作为JPEG XT（ISO/IEC 18477-3）定义的**APP11**标记段所含数据进行嵌入。

由于JPEG 1中单个标记段的大小不得超过64K字节，因此可能需要多个APP11段，这些段应按照JPEG 1标准及ISO 19566-5:2023 D.2条款构建。写入多个段时，应按顺序连续写入（即相邻段之间无间隔）。

A.3.2. 将清单嵌入PNG文件

C2PA清单存储应采用辅助性、私有且不可安全复制的`'caBX'`块类型进行嵌入（依据PNG规范4.7.2节）。建议将'caBX'块置于'IDAT'块之前。

注

尽管PNG支持此操作，但在`'IDAT'`之后和`'IEND'`之前放置数据块被视为不良做法（动画PNG块除外）。

A.3.3. 将清单嵌入SVG

SVG是一种基于XML的格式，既可独立存在，也可嵌入HTML等其他文本格式中。因此，为实现嵌入功能，必须对二进制C2PA清单存储库进行Base64编码。本节虽描述了具体操作方法，但建议优先使用外部清单。

C2PA清单存储应作为`c2pa:manifest`元素的Base64编码值嵌入SVG的元数据元素中。由于XML（尤其是SVG）强烈建议在使用前声明命名空间，需在svg元素中添加xmlns`:c2pa="http://c2pa.org/manifest"`属性声明。

示例17. SVG中的C2PA清单存储示例

SVG中的C2PA清单存储示例（实际C2PA清单数据已省略）。

```
<?xml version="1.0" standalone="yes"?>
<svg width="4in" height="3in" version="1.1" xmlns =
     "http://www.w3.org/2000/svg" xmlns:c2pa =
     "http://c2pa.org/manifest">
     <metadata>
          <c2pa:manifest>...此处填充Base64数据...</c2pa:manifest>
     </metadata>
</svg>
```

A.3.4. 将清单嵌入ID3

C2PA清单存储库应嵌入到符合ID3v2标准的压缩音频文件（如MP3或FLAC）中，作为通用封装对象（GEOB）的封装对象数据，具体定义参见 https://id3.org/id3v2.3.0。GEOB的MIME类型字段必须存在，且应采用第11.4节"外部清单"中描述的JUMBF媒体类型值。

A.3.5. 将清单嵌入基于TIFF的资产

数字负片（DNG）格式为相机制造商提供了一种标准化方式来发布其相机原始格式。DNG基于TIFF/EP（该格式本身基于TIFF）构建。

C2PA清单存储库应嵌入至TIFF兼容文件（即TIFF/EP、DNG或其他基于TIFF的RAW格式）中，作为ID为52545（十进制）或0xCD41（十六进制）的标签数据，标签类型为7。

尽管TIFF支持多页/多层概念（通过多个IFD实现），但整个资产仅应包含一个C2PA清单存储区——而非每个IFD对应一个。因此，C2PA清单存储区应作为唯一数据框存在于最后一个IFD中，即紧邻文件末尾的IFD。

注 本规范的早期版本要求使用IFD 0，但后来发现此举限制了其在基于TIFF的RAW格式中的应用。

A.3.6. 将清单嵌入基于RIFF的资源

RIFF（资源交换文件格式）提供通用容器格式，用于存储带标签的数据块。该格式主要用于存储图像、音频和视频等多媒体内容，同时也是WAV、BWF、广播波形文件、AVI及WebP的容器格式。

注 RIFF基于更早的IFF格式发展而来。

C2PA清单存储应嵌入到RIFF兼容文件（即WAV、AVI或WebP）中，作为标识符为C2PA的数据块。出于兼容性考虑，该C2PA数据块应位于RIFF数据块末尾。

A.3.7. 将清单嵌入GIF文件

C2PA清单存储应分割为不超过255字节的块，并嵌入至C2PA专用应用扩展块（符合GIF规范第26条）内的连续数据子块中（符合GIF规范第15条），具体如下所述。

注： 在此C2PA应用扩展块中，应用认证码不用于验证生成区块的应用程序。相反，它被用作区块版本号，初始设置为主版本号1、次版本号0，并按下述方式编码。

扩展标识符：`0x21` 应用扩展标签：`0xFF` 区块大小：`0xB`
应用标识符：`0x43, 0x32, 0x50, 0x41, 0x5F, 0x47, 0x49, 0x46` ("C2PA_GIF")应用认证码：`0x010000` (0x[主版本][次版本]00) 应用数据：C2PA清单存储区，以数据子块序列编码，每个子块包含1字节大小标记后接最多255字节数据
块终止符：`0x00`（置于C2PA清单存储的最后一个数据子块之后）数量：一个

该块应嵌入在头部之后、首个图像描述符框之前。

A.3.8. 将清单嵌入JPEG XL

如ISO/IEC 18181-2:2024第4节所述，JPEG XL支持两种不同的数据格式。它可以采用与JPEG 2000和JPEG XS兼容的框结构，也可以是没有框结构的直接JPEG XL编码流。采用框结构的JPEG XL文件最多包含一个JUMBF（巨型）超级框（参见ISO/IEC 18181-2:2024第9.3节），该超级框内含C2PA清单JUMBF框，其中存储着第11.1.4.2节"清单存储区"所述的C2PA清单。仅作为码流的JPEG XL文件无法包含嵌入式C2PA清单。

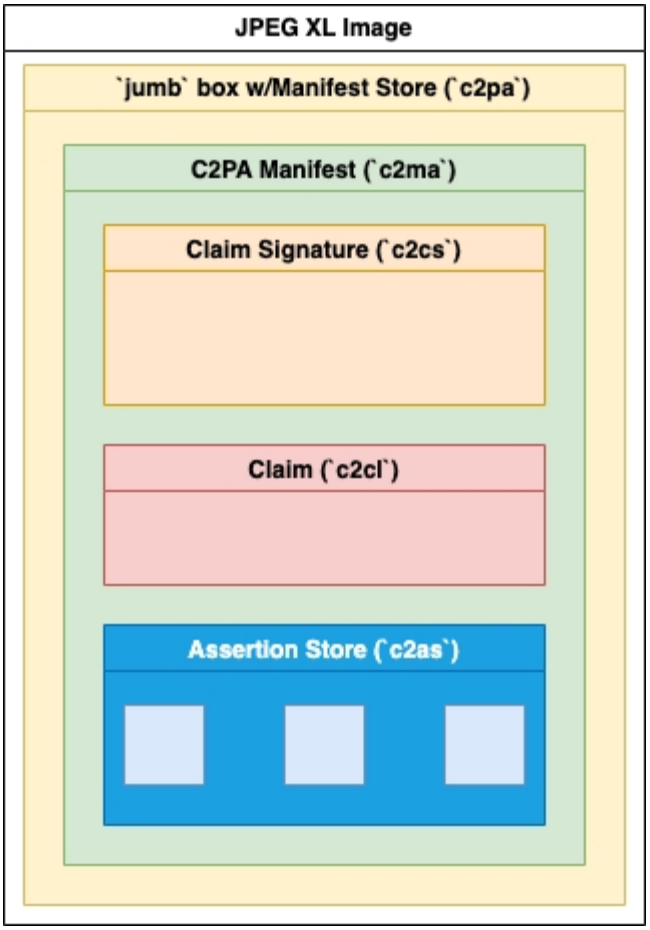

图19. 嵌入JPEG XL图像的C2PA清单

A.3.9. 将清单嵌入字体

符合开放字体格式（OFF）或OpenType规范的字体可包含C2PA表。当存在该表时，其内容可能包含嵌入式清单、远程清单URI或两者兼有。

C2PA表格式尚未在Open Font Format或OpenType规范中定义；以下定义为初步版本：

A.3.9.1. 表标签

C2PA表记录将通过以下表标签标识：`C2PA`。

A.3.9.2. 表记录

C2PA表全面支持清单存储以嵌入式、远程式或两者兼具的形式存在。表记录定义如下：

表 14. C2PA 表记录

类型	名称	描述
`uint16`	主要版本	指定 C2PA 字体表的主版本号。

uint16	minorVersion	指定 C2PA 字体表的次要版本。
偏移量32	activeManifestUriOffset	从 C2PA 字体表开头到包含活动清单 URI 的部分的偏移量。 如果未提供 URI，则应使用 NULL 偏移量 = 0x0000。
uint16	activeManifestUriLength	URI的长度（以字节为单位）。
uint16	保留	保留用于将来使用。
Offset32	manifestStoreOffset	从 C2PA 字体表开头到包含 C2PA 清单存储的区段的偏移量。 若未提供清单存储，则应使用 NULL 偏移量 = 0x0000。
uint32	manifestStoreLength	C2PA清单存储数据的长度（以字节为单位）。

非嵌入式C2PA清单可位于远程位置或本地同一存储系统中。若引用为JUMBF URI，则该引用应在C2PA清单存储库内有效。

A.4. 将清单嵌入PDF文件

A.4.1. 通用要求

所有C2PA清单存储均应通过嵌入式文件流（ISO 32000, 7.11.4）进行嵌入。嵌入文件规范字典应包含以下键值：Subtype键值为application/c2pa，AFRelationship键值（ISO 32000, 7.11.3）为C2PA_Manifest。若C2PA清单存储嵌入加密PDF，嵌入文件流须使用身份加密过滤器。

A.4.2. 文档级清单

A.4.2.1. 向PDF添加清单

向整个PDF添加C2PA清单时，文档目录字典应包含一个值为间接引用（指向包含活动清单的嵌入文件规范）的AF条目。该嵌入文件规范还应通过间接对象从EmbeddedFiles名称树（/Catalog/Names/EmbeddedFiles）或文件附件注释中被引用。当向已存在PDF认证签名的文档添加C2PA清单存储时，应采用注释方式以避免使其DocMDP限制失效。

注 DocMDP字典中P字段值*为1或2时*不允许此类修改，仅值为*3时*允许。

在大多数其他格式中，仅存在单一的C2PA清单存储库，其中包含该资产的所有C2PA清单。然而，由于PDF的"增量更新"特性，必须支持单个PDF文件中存在多个清单。在此场景下，基础PDF中的C2PA清单存储库应视为初始清单，而最新更新中的清单则为活动清单。C2PA清单消费者应将所有C2PA清单存储库中的清单视为存储于单一清单存储库中进行处理。

注 由于JUMBF URI始终是完整的URI，这意味着它始于指定的C2PA清单，且所有C2PA清单均被视为包含于单一C2PA清单存储库中，因此在PDF中使用此类URI跨C2PA清单存储库引用`parentOf`组件是可接受的。

A.4.2.2. 与PDF签名的兼容性

在添加新的C2PA清单存储库时，必须明确是否同时应用PDF签名（认证或批准）。由于PDF签名会在C2PA清单签名后改变PDF数据，因此必须在C2PA签名之前确定PDF签名字典中Contents键的大小和位置。该字节范围需添加至`c2pa.hash.data`断言的排除项列表中，以确保PDF签名添加后不致使C2PA签名失效。PDF签名应覆盖整个PDF文件，包括关联的C2PA清单存储库。

注 在C2PA声明签名基础上添加PDF签名，可增强与现有PDF生态系统的兼容性。

A.4.3. 对象级清单

除了能够通过文档级清单为PDF本身提供来源信息外，文档中的单个对象也可关联C2PA清单存储库。实现方式是在对象的流或字典中添加`AF`条目。该AF条目的值应为间接引用，指向包含C2PA清单存储库的嵌入文件规范，其嵌入方式如上所述。

该功能最常见的应用场景是为嵌入的图像（以Image或Form XObject形式）及字体提供来源信息。也可通过将AF条目添加至对象（经属性列表）或结构元素，为特定内容片段提供来源信息，具体实现方式参见ISO 32000-2标准中"关联文件"条款（14.13.1）。

建议将任何新增的对象级清单作为`componentOf`组件从活动清单中引用。此举可使C2PA清单消费者轻松遍历资产的完整来源链。

通常，任何PDF流或字典都可附加C2PA清单，前提是该流或字典代表实际信息资源。当无法明确确定应将AF条目附加于哪个流或字典时，清单应尽可能附加于实际存储所描述数据资源的对象。

注 描述光栅图像的C2PA清单应附加于图像XObject流

描述该图像的图像XObject流，而嵌入字体文件的清单应附加于字体文件流而非字体字典。

字体字典。

A.4.4. 示例

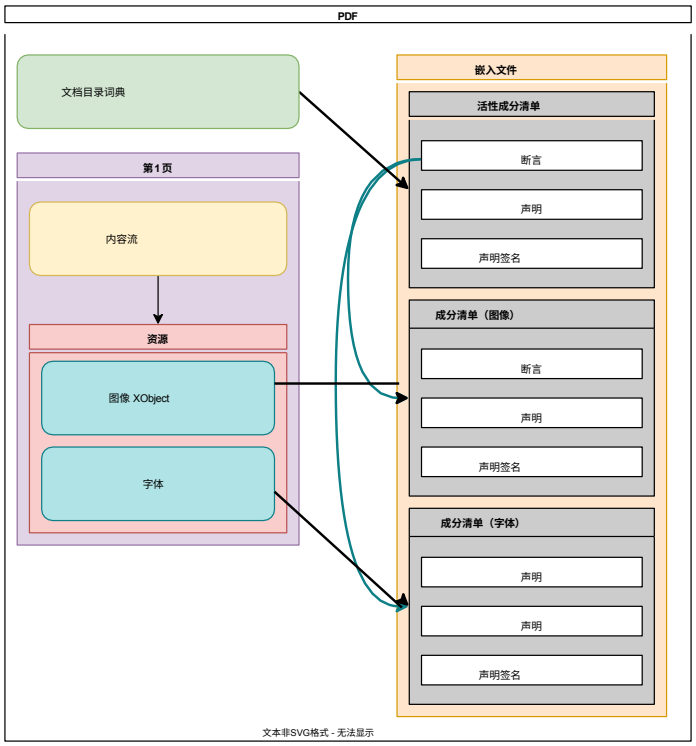

图20. 包含多个成分清单的PDF示例

A.5. 将清单嵌入基于BMFF的资产

A.5.1. C2PA的`'uuid'`框

所有基于BMFF的C2PA资产——无论属于带时间轴的视听媒体（如含/不含音轨的视频）、无时间轴的静态媒体（如照片），还是混合媒体（如动态照片）——均须采用符合下列语法与语义定义的'uuid'框。

注 使用`'uuid'`框而非`'c2pa'`框的原因在于，基于Chromium的浏览器在遇到任何未知顶级框时会立即停止播放。

基于BMFF且可通过此方法支持的文件格式包括：

- MPEG-4编码点文件（完整格式`.mp4`或分段格式`.m4s`）；可下载音频文件`.m4a`；
- HEIF格式（`.heif`, `.heic`）；
- AVIF格式（`.avif`）。

A.5.1.1. 定义

```
盒类型：'uuid'
```

扩展框类型：0xD8, 0xFE, 0xC3, 0xD6, 0x1B, 0x0E, 0x48, 0x3C, 0x92, 0x97, 0x58, 0x28, 0x87, 0x7E, 0xC4, 0x81
容器：文件 强制性：否 数量：零或
更多

C2PA的'uuid'框将来源信息嵌入BMFF。其中一个框包含C2PA清单存储库，可能还存在一个或多个辅助框，用于存储验证所需的附加信息。

A.5.1.2. 语法

```
对齐(8) 类 ContentProvenanceBox 继承自 FullBox('uuid',extended_type = 0xD8 0xFE 0xC3 0xD6 0x1B 0x0E 0x48 0x3C 0x92 0x97
0x58 0x28 0x87 0x7E 0xC4 0x81, version = 0, 0) {
     string box_purpose;
     bit(8) data[];
}
```

A.5.1.3. 关于唯一标识符

存在某些情况（如分段MP4文件），资产子集的标识符（例如'tkhd'盒中的track_id字段）仅在该子集范围内具有局部唯一性，而非对整个资产具有全局唯一性。

由于需要全局唯一标识符来确定哈希对象，因此包含一个唯一标识符。该唯一标识符不与原始资产的任何值相同；相反，每个值在清单创建时定义。随后，该唯一标识符与关联的本地标识符组合，形成对整个资产具有全局唯一性的标识符。

A.5.2. 语义说明

下文将针对每个数据框说明其用途（box_purpose）及其关联字段（data）。

A.5.3. 包含清单的框

包含C2PA清单存储的框应出现在文件中首个'mdat'框之前，且位于任何'moov'框之前。为满足主要品牌和兼容品牌验证需求，该框应置于'ftyp'框之后。当资产的活动清单为更新清单时，先前标准的C2PA清单存储位置保持不变，但需将box_purpose属性修改为original。更新后的C2PA清单存储区应作为文件末尾的最后一个框存在，其框用途应设置为update。

上述对应框中的字段应按以下方式设置。

box_purpose

对于 C2PA 清单存储库，该值应为 manifest、original 或 update。

data

当box_purpose为manifest时，'data'字段前8字节应为指向首个辅助'uuid' C2PA框的绝对文件字节偏移量（该框的box_purpose需等于merkle）。若文件中不存在此类框，则该8字节

该字段应为零。这8字节之后应紧跟原始C2PA清单存储字节，随后是零个或多个未使用的填充字节。当box_purpose为original时，表示存在另一个box_purpose值设为update的C2PA盒子。此原始盒子内的'data'保持不变。当box_purpose为update时，C2PA清单存储仅应包含更新清单。

注

类型为清单或原始的'uuid'盒子内的'data'字段包含绝对文件路径

字节偏移量、清单及填充字节。原始框与清单框除box_purpose值外完全相同，因此哈希绑定关系保持不变。追加'update'框不会移动任何哈希数据。

填充字节不得出现在'uuid'框之外，除非它们包含在独立的mp4框中，例如'free'框。

对于分段MP4（fMP4）文件，每个初始化分段中必须包含类型为清单的相同'uuid' C2PA框；C2PA清单存储必须完全一致。

A.5.4. 用于大型及分段文件的辅助'c2pa'框

A.5.4.1. 通用规则

某些文件包含一个或多个超大'mdat'框（如需渐进式下载渲染的大型视频/图像文件），或大量独立'mdat'框（如每个片段可独立下载的fMP4文件）。

在此类情况下，要求客户端在验证资源任何部分前完全下载所有'mdat'框既不合理也不现实。通过使用多重哈希值可解决此必要性问题。

对于每个大型'mdat'框，其子集拥有可独立验证的单独哈希值；子集的确定方法如下所述。对于fMP4内容（其中每个'mdat'框均可独立下载），每个片段均拥有独立的哈希值。

最简情况下，所有哈希值均存储于活动清单中。每个子集配有辅助的"uuid" C2PA框，用于声明其哈希值在活动清单中的定位方式；具体原因请参阅上文关于唯一标识符的说明。

然而，对于足够大的资产而言，将每个子集的哈希值都包含在清单本身中，会使C2PA清单存储库的大小增加到一兆字节或更大。

为大型资产避免这种庞大的C2PA清单存储库，可通过使用一个或多个默克尔树实现。

- 对于包含一个或多个 'mdat' 框的大型非碎片化资产（即单个大文件），每个 'mdat' 框使用一棵默克尔树。
- 对于包含单一音轨的碎片化大型资产（该音轨可能分散于多个文件），则为每条音轨分别使用一棵默克尔树。

两种情况均需满足：

- 任意给定默克尔树的每个叶节点都是该子集的哈希值。
- 清单存储每棵默克尔树的一行数据。
- 每个子集对应的辅助'uuid' C2PA框标识其所需的活动清单中哪行默克尔树数据，以及该行所代表的叶节点。该框还包含默克尔树中任何额外哈希值，这些值用于推导活动清单默克尔树行中的哈希值。

在清单中选择存储哪一层默克尔树会产生资产内部的大小权衡。具体而言，若为每层默克尔树仅存储单个哈希值，虽能最小化清单体积，但每个子集专用框需存储log2(子集数)个数据。每当清单中某默克尔树存储的哈希值数量翻倍（通过向下移动一行默克尔树实现），每个子集专用框中存储的哈希值数量就会减少一个。因此，增加清单大小会缩减整个资产的体积，反之亦然。由于各子集的哈希值需跨子集复制以生成清单指定的哈希值，这种权衡关系并非完全等价。

这种尺寸权衡的决策权交由创建清单的实现方决定；本规范既不强制要求也不建议在清单中存储任何特定的默克尔树行。不过，由于在清单中存储所有子集哈希的最简单情况等同于使用默克尔树（其中叶节点存储在清单中），因此所有情况下都采用相同的默克尔树构造来处理多个哈希值。该构造定义如下：

清单中包含BMFF哈希值的部分应包含默克尔字段。更多信息请参阅第9.2.3节“对BMFF格式资产进行哈希处理”。

A.5.4.1.1. 可分段验证的非碎片化资产

若清单包含默克尔树的非叶节点行，则文件中应包含两个或更多辅助'uuid' C2PA框，其box_purpose字段需按下述方式设置为'merkle'。若清单仅包含默克尔树的叶节点行，则无需包含这些辅助框。若存在辅助框，它们应紧跟在文件中的最后一个'mdat'框之后。

默克尔树中给定叶节点的哈希值应基于'mdat'有效负载的子集计算。'mdat'被划分为由'fixedBlockSize'或bmff-merkle-map中'variableBlockSizes'数组定义的大小，且'variableBlockSizes'的总和应等于'mdat'有效负载的大小。

所有此类辅助'uuid' C2PA框均须满足以下要求：

- 它们应与'variableBlockSizes'字段中定义的子集保持相同序列。
  字段所定义的顺序排列。
- 它们应进行分组，使单个默克尔树的辅助 'uuid' C2PA 框连续排列，中间无其他框。
- 第一个框中的位置值应设置为0，第二个框应设置为1，并依次递增。

依次递增。

A.5.4.1.2. 碎片化资产

对于拆分至多个文件的fMP4资产：

- 每个碎片文件中应包含一个辅助的 'uuid' C2PA 框，其 box_purpose 属性应按下述方式设置为 'merkle'，该框应紧接在 'moof' 框之前。
- 默克尔树中给定叶节点的哈希值应基于其包含的单个片段文件中的所有数据计算，但排除列表中指定的数据除外。

注 本规范不支持跨多个文件分割的fMP4资产，其中单个片段文件包含多个'moof'框或'mdat'框或两者兼有。

对于fMP4资产，其存储形式为单个扁平化MP4文件，所有轨道共享单个'moov'，每个片段对应一对'moof'/'mdat'对：

- 每个'moof'框之前应包含一个辅助'uuid' C2PA框，其box_purpose属性按下述方式设置为'merkle'。
- 默克尔树中叶节点的哈希值应覆盖该'moof'框及后续所有数据（直至下一个'moof'框），若无后续'moof'框则覆盖至文件末尾。该哈希值不应包含排除列表中指定的数据。

重要 将一个跨多个文件分割的C2PA兼容fMP4资产（即包含类型为'manifest'和'merkle'的'c2pa'框）进行文件拼接，不会生成符合C2PA规范的单一文件（反之亦然）。这是因为包含在每个'merkle'哈希中的框将有所不同。如果两种形式都可取，则第二种形式应将第一种形式视为成分，新的清单应同时包含具有parentOf关系的成分断言和包含c2pa.repackaged类型操作的操作断言。

A.5.4.1.3. 包含默克尔辅助信息的框

无论资产结构如何，上述对应框中的字段均应按以下方式设置：

箱体用途

对于辅助的 'uuid' C2PA 框，该值应为默克尔树。

data

当box_purpose为merkle时，该值应包含原始CBOR字节，用于指示如何验证资产的特定部分，定义如下。若单个文件中存在多个辅助 'uuid' C2PA框，且其box_purpose均为merkle对应同一默克尔树时，每个框后均需跟随足够的填充字节（零个或多个），以确保该默克尔树下所有辅助'uuid' C2PA框保持固定长度。

注 当单个文件中存在多个此类框时（即存在大量

由于'mdat'文件需逐块验证，因此需要固定大小的区块，以便渐进式下载的客户端仅下载开始验证所需的区块，而非整个默克尔树。此类客户端可根据活动清单中的绝对文件字节偏移量下载足够数量的首批区块，从而判断其uniqueId和localId是否与待验证的'mdat'文件匹配。若匹配成功，则通过将子集编号乘以该固定大小计算出待验证数据块的绝对文件字节偏移量，仅下载该数据块。否则，通过将固定大小乘以当前默克尔树的叶节点总数，可计算出下一个默克尔树起始位置的绝对文件字节偏移量，并重复此过程直至定位所需数据块。相对于单个子集的大小，该子集盒子的总下载量非常小。

A.5.4.2. 模式与示例

此类型的模式由以下CDDL定义中的bmff-merkle-map规则定义：

```
; 用于存储验证单个'mdat'框或

; 或当使用默克尔树时验证'mdat'框部分所需的充分信息", bmff-merkle-map = {
   "uniqueId": int, ; 用于区分本地ID的唯一整数"localId": int, ; 标识默克尔树的本地ID
   "location": int, ; 指向该'mdat'框或其部分在叶节点层级Merkle树行中的零基索引
   ? "hashes": [1* bstr], ; 表示从叶节点（本节点的对等节点）到根节点（清单中节点的子节点）所需额外哈希值的有序数组，用于在清单指定的默克尔树中定位哈希值。请注意该数组可能不存在，例如清单本身已包含默克尔树的叶节点行。此数组不包含空哈希值。所用算法通过BMFF哈希结构中`merkle`字段数组对应条目的`alg`字段确定。
}
```

以下为采用CBOR诊断标记法（RFC 8949第8节）的示例：

```
{
   "哈希值": [ b64'TWVub3JhaA==' 
   ],
   "localId": 4402,
   "location": 2203,
   "唯一标识符": 1339
}
```

对于非分段资产，bmff-merkle-map中的localId字段应指向'mdat'框。该字段采用零基索引，表示文件中'mdat'框的顺序。对于分段资产，bmff-merkle-map中的localId字段应设置为被哈希的'mdat'框所属'tkhd'框的track_id字段值。

A.5.5. 动态流生成

许多自适应比特率流媒体（ABR）实现仅存储资产的单一版本，例如以扁平MP4格式或

另一种中间格式，并在消费时使用各种编解码器、比特率等生成独立的资产流。因此，此类服务器应在每次内容消费时对所述流进行哈希处理并创建C2PA清单；若生成过程具有确定性，则只需一次性创建并缓存哈希值和C2PA清单，随后在消费时将其嵌入。

A.5.6. 排除列表要求

对于所有`c2pa.hash.bmff.v2`（已弃用）和`c2pa.hash.bmff.`v3断言，示例18"始终排除的框"中的条目必须始终出现在排除列表中。其他条目允许但非必需。

整个`'uuid'` C2PA框应被排除（`'data'`字段确保其他`'uuid'`框不被排除）。

示例18. 永久排除框

```
xpath = "/uuid"
data = [ { offset = 8, data = b64'2P7D1hsOSDySl1goh37EgQ==' } ]
```

整个`'ftyp'`和`'mfra'`框均应被排除。

```
xpath = "/ftyp"
```

```
xpath = "/mfra"
```

注 本规范的早期版本包含额外的强制排除项，但后来发现排除这些项存在安全隐患。

对于所有包含哈希字段和默克尔字段的bmff-hash-map的c2pa`.hash.bmff.v2`（已弃用）和`c2pa.hash.bmff.`v3断言，示例19中"始终排除的附加框"条目应出现在排除列表中。

示例19. 附加始终排除框

```
xpath = "/mdat"
子集 = { { 16, 0 } }
```

注 如上文CDDL定义所示，`c2pa.hash.`bmff断言原本排除整个`'mdat'`框，但发现这种排除方式存在安全隐患。

如上文CDDL定义所述，允许使用超过框体长度的相对字节偏移量或相对字节偏移量加长度；框体末尾之后的字节绝不应被哈希处理。例如，若mdat框体仅有12字节长度，则其全部内容均需哈希处理，此时前述强制排除条目虽仍需保留，但将不产生实际效果。

仍需保留。

A.5.7. 非音频非视频的定时媒体流

对于申明生成器希望实现防篡改功能的非音频非视频定时媒体流（如字幕文本流），其处理方式应与音频视频流保持一致。

A.5.8. 外部引用

在BMFF框内声明的外部引用内容（如'dref'、'url'或'urn'框），若声明生成器希望使其具备防篡改特性，**则**不得排除引用框本身，且需为每个待哈希的外部引用单独添加云数据断言。

A.5.9. 尺寸要求

若基于BMFF的资产在任何框中使用32位大小或偏移量（例如'stco'框，且添加符合本规范的框将导致文件大小超过4千兆字节时，清单创建者有责任在生成清单前编辑文件以使用适当的大小和偏移量，例如将'stco'框替换为'co64'框。

A.6. 将清单嵌入基于ZIP的格式

A.6.1. 通用说明

由于其历史悠久且采用公开发布的规范，许多命令文件格式本质上是ZIP压缩包，但内容文件具有特定的组织结构。这包括EPUB、Office Open XML、Open Document和OpenXPS等格式。

A.6.2. 哈希处理

A.6.2.1. 文件哈希处理

基于ZIP的文件格式应采用集合数据哈希进行哈希处理，其中ZIP内包含的每个文件（除C2PA清单文件本身外）均应纳入计算范围。集合中每个文件的哈希值需基于以下内容计算：文件的本地文件头、紧随其后的压缩和/或加密内容，以及存在的任何数据描述。所使用的哈希算法应在集合数据哈希结构的`alg`字段中明确指定。

注：	对压缩/加密内容进行哈希计算的原因在于，这使得验证过程无需需要解压缩或拥有解密密钥。这对可加密的格式（如EPUB）至关重要。

A.6.2.2. ZIP中央目录的哈希处理

如ZIP应用说明4.3.12所述，中央目录是由中央目录头组成的数组——ZIP存档中的每个文件对应一个头。该数组存储于ZIP存档末尾，用于定位存档内文件及其必要

信息/元数据。紧随其后的是中央目录结束记录（参见ZIP应用说明4.3.16），该记录包含关于ZIP存档本身的详细信息。

为防止篡改ZIP中央目录（例如添加新文件或修改现有文件信息），ZIP中央目录中的每个"中央目录头"以及"中央目录结束记录"均需进行哈希处理。该哈希值通过哈希算法计算得出，该算法由集合数据哈希结构的`alg`字段指定，计算范围涵盖从"中央目录头"首个字节到"中央目录结束记录"末尾字节的全部字节序列。

注 "中央目录头"连续存储，其后紧接"中央目录结束记录"。

计算结果的哈希值应存储在集合数据哈希结构的`zip_central_directory_hash`字段中。

注 曾考虑在文件列表中使用特殊命名的文件，但因下文所述的两遍处理场景而未被采纳。

```
; URI及其关联哈希值的数组
$collection-data-hash-map /= { "uris": [1* uri-
   hashed-data-map],
   "alg": tstr .size (1..max-tstr-length), ; 用于计算`uris`数组中每个条目哈希值的密码哈希算法标识符，取自C2PA哈希算法标识符列表。
      ? "zip_central_directory_hash" : bstr,
}

; 用于存储URI及其哈希值的引用数据结构。
$uri-hashed-data-map /= {
   "uri": 相对URL类型, ; 相对URI引用"hash": 字节字符串, ; 包含哈希值的字节字符串
   ? "size": size-type, ; 数据字节数
   ? "dc:format": 格式字符串, ; 数据的IANA媒体类型
   ? "data_types": [1* $asset-type-map], ; 数据类型的附加信息
}

; 使用 CBOR 头 (#) 和尾 ($) 引入正则表达式，因此无需显式指定 relative-url-type /= tstr .regexp "[-a-zA-Z0-
9@:%._\\+~#=]{2,256}\\.[a-z]{2,6}\\b[-a-zA-Z0-9@:%_\\+.~#?&//=]*"
```

由于 ZIP 文件需在 C2PA 清单完成前生成，故采用两阶段处理方案（与 JPEG、BMFF 和 PDF 处理方式相同）。第一遍生成包含零填充内容的`content_credential`.c2pa文件的ZIP文件，并计算ZIP中央目录的哈希值。第二遍完成C2PA清单，包括填充`zip_central_directory_hash`字段的值。

此两阶段方法的具体实现方案可为：

- 创建包含零填充C2PA清单存储文件的ZIP（容量需足够大以供后续替换）；
- 计算ZIP中央目录的哈希值；
- 将哈希值添加至集合数据哈希映射的`zip_central`_directory_hash字段；

- 完成清单构建。
- 用完成的清单数据覆盖零填充的 `content_credential.c2pa` 文件。

在ZIP存档中创建`content_credential.c2pa`文件时，应采用压缩方式0存储且不加密。其通用位标志和`crc-32`字段应设为0。日期和时间字段可设置为ZIP存档创建时间，或设为0。该文件可包含文件注释。

A.6.3. 清单存储位置

C2PA清单存储库应存放于ZIP存档的META-INF目录中，文件名为`content_credential.c2pa`，媒体类型应遵循外部清单推荐规范。该文件应采用压缩方式0存储且不加密。

A.6.4. 对基于ZIP的格式进行数字签名

A.6.4.1. EPUB

EPUB的数字签名基于W3C XML DigSig核心规范，其中每个被签名的文件均列于`<Signature>`元素的`<Manifest>`子元素中列出。此外，不支持对ZIP中央目录进行签名。因此，EPUB原生签名操作应在引入C2PA清单前完成。

A.6.4.2. Office Open XML

OOXML的数字签名基于W3C XML DigSig核心规范，其中每个被签名的文件均作为`<Reference&gt;`元素。此外，该规范不支持对 ZIP 中央目录进行签名。因此，OOXML 原生签名操作必须在引入 C2PA 清单之前完成。

注 OpenXPS与OOXML基于相同的开放包装规范（OPC）标准，故采用相同处理方式。

附录B：c2pa.metadata的实现细节

c2pa.metadata

c2pa.metadata断言仅应包含如下所述的模式及其字段子集。但自定义元数据断言可包含来自这些或其他模式的任意值。

注 所有有效模式及其字段的机器可读列表可参见C2PA规范网站。

c2pa.metadata断言中的值可为该元数据断言所独有，也可取自资产格式的标准"元数据块"。无论哪种情况，均应遵循此处所述的XMP JSON-LD序列化规则进行序列化。

B.1. 完全支持的架构

表 15“完全支持的架构”中列出的以下架构/命名空间，任何签名者均可完全支持：

表15. 完全支持的架构

名称	命名空间
XMP Basic	http://ns.adobe.com/xap/1.0/
XMP媒体管理	http://ns.adobe.com/xap/1.0/mm/
XMP分页文本	http://ns.adobe.com/xap/1.0/t/pg/
Camera Raw	http://ns.adobe.com/camera-raw-settings/1.0/
PDF	http://ns.adobe.com/pdf/1.3/

B.2. 部分支持的架构

表 16“部分支持的架构”中列出的以下架构/命名空间仅提供部分支持。

表 16. 部分支持的架构

名称	**命名空间**
都柏林核心元数据（DC）	http://purl.org/dc/elements/1.1/
IPTC核心	http://iptc.org/std/Iptc4xmpCore/1.0/xmlns/
IPTC 扩展	http://iptc.org/std/Iptc4xmpExt/2008-02-29/
Exif	http://ns.adobe.com/exif/1.0/
ExifEx	http://cipa.jp/exif/1.0/exifEX
名称	**命名空间**

Photoshop	http://ns.adobe.com/photoshop/1.0/
TIFF	http://ns.adobe.com/tiff/1.0/
XMP动态媒体	http://ns.adobe.com/xmp/1.0/DynamicMedia/
PLUS	http://ns.useplus.org/ldf/xmp/1.0/

B.2.1. 都柏林核心元数据（DC）

仅支持以下都柏林核心（dc）属性：

- dc:coverage
- dc:日期
- dc:format
- dc:标识符
- dc:language
- dc:relation
- dc:type

B.2.2. IPTC核心

仅支持以下 IPTC 核心（Iptc4xmpCore）属性：

- Iptc4xmpCore:场景

注意 部分 IPTC 核心属性已被 IPTC 扩展架构中的新版本取代。

B.2.3. IPTC扩展

仅支持以下IPTC扩展（Iptc4xmpExt）属性：

- Iptc4xmpExt:数字图像GUID
- Iptc4xmpExt:DigitalSourceType
- Iptc4xmpExt:EventId
- Iptc4xmpExt:Genre
- Iptc4xmpExt:图像评分
- Iptc4xmpExt:图像区域
- Iptc4xmpExt:注册ID
- Iptc4xmpExt:拍摄地点

- `Iptc4xmpExt:`显示位置
- `Iptc4xmpExt:`最大可用高度
- `Iptc4xmpExt:`最大可用宽度

For 更多 信息 关于 这些， 请参阅 请 https://www.iptc.org/std/photometadata/specification/IPTC-PhotoMetadata#xmp-namespaces-and-identifiers-2。

B.2.4. Exif

仅支持表17“支持的Exif属性”中列出的以下属性：

表 17. 支持的 Exif 属性

- exif:光圈值
- exif:BrightnessValue
- exif:CFAPattern
- exif:ColorSpace
- exif:压缩位/像素
- exif:对比度
- exif:自定义呈现
- exif:原始日期时间
- exif:原始日期时间
- exif：设备设置描述
- exif:数字变焦比率
- exif:Exif版本
- exif:曝光补偿值
- exif:曝光指数
- exif:曝光模式
- exif:曝光程序
- exif:曝光时间
- exif:文件来源
- exif:闪光灯
- exif:闪光能量
- exif:闪光灯像素版本
- exif:光圈值
- exif:FocalLength
- exif:焦距（35mm胶片等效）
- exif:焦平面分辨率单位
- exif：焦平面X分辨率
- exif:焦平面Y分辨率
- exif:增益控制
- exif:图像唯一ID
- exif:ISO感光度评级
- exif:光源
- exif:最大光圈值
- exif:测光模式
- exif:原始偏移时间
- exif:原始偏移时间
- exif:原始偏移时间
- exif:原始偏移时间
- exif:相关声音文件
- exif:饱和度
- exif:场景捕捉类型
- exif:场景类型
- exif:SensingMethod
- exif:锐度
- exif:快门速度值
- exif:空间频率响应
- exif:频谱灵敏度
- exif:拍摄对象区域
- exif:拍摄距离
- exif:拍摄距离范围
- exif:白平衡
- exif:白平衡
- exif:GPS海拔
- exif:GPS海拔参考
- exif:GPS日期戳
- exif:GPS目标方位
- exif:GPS目标方位参考
- exif:GPS目标距离
- exif:GPS目的地距离参考
- exif:GPS目的地纬度
- exif:GPS目标经度
- exif:GPS差分
- exif:GPSDOP
- exif:GPSHPositioningErr 或
- exif:GPS图像方向
- exif:GPS图像方向参考
- exif:GPS经度
- exif:GPS经度
- exif:GPS地图基准面
- exif:GPS测量模式
- exif:GPS处理方法
- exif:GPS卫星
- exif:GPS速度
- exif:GPS速度参考
- exif:GPS状态
- exif:GPS时间戳
- exif:GPS轨迹
- exif:GPS轨迹引用
- exif:GPS版本ID

B.2.5. ExifEx

仅支持以下ExifEx属性：

- exifEX:BodySerialNumber
- exifEX:Gamma
- exifEX:互操作性指数
- exifEX:ISO速度
- exifEX:ISO感光度纬度yyy
- exifEX:ISO感光度范围
- exifEX:镜头制造商
- exifEX:镜头型号
- exifEX:镜头序列号
- exifEX:镜头规格
- exifEX:感光度
- exifEX:推荐曝光指数
- exifEX:感光度类型
- exifEX:标准输出感光度

有关这些信息的更多内容，请参阅 https://www.cipa.jp/std/documents/download_e.html?DC-010-2020_E。

B.2.6. Photoshop

仅支持以下 Photoshop 属性：

- photoshop:类别
- photoshop:City
- photoshop:ColorMode
- photoshop:国家
- photoshop:创建日期
- photoshop:文档祖先
- Photoshop：历史记录
- photoshop:ICCProfile
- photoshop:州
- photoshop:补充类别

- photoshop:文本图层
- photoshop:传输参考
- photoshop:紧急程度

B.2.7. TIFF

仅支持以下 TIFF 属性：

- tiff:每样本位数
- tiff:压缩
- tiff:DateTime
- tiff:图像长度
- tiff:ImageWidth
- tiff:制造商
- tiff:模型
- tiff:方向
- tiff:光度解释
- tiff:平面配置
- tiff:主色度
- tiff:参考黑白
- tiff:分辨率单位
- tiff:每像素采样数
- tiff:软件
- tiff:传输函数
- tiff:白点
- tiff:X分辨率
- tiff:Y分辨率
- tiff:YCbCr系数
- tiff:YCbCrSubSampling
- tiff:YCbCr子采样

B.2.8. XMP动态媒体

仅支持表 18“XMP 动态媒体属性”中列出的以下 XMP 动态媒体 (xmpDM) 属性：

表 18. XMP 动态媒体属性

- xmpDM:absPeakAudioFilePath
- xmpDM:album
- xmpDM:替代磁带名称
- xmpDM:替代时间码
- xmpDM:音频通道类型
- xmpDM:音频压缩器
- xmpDM:音频采样率
- xmpDM:音频采样类型
- xmpDM:节拍拼接参数
- xmpDM:摄像机角度
- xmpDM:摄像机标签
- xmpDM:摄像机型号
- xmpDM:摄像机移动
- xmpDM:注释
- xmpDM:贡献媒体
- xmpDM:持续时间
- xmpDM:文件数据速率
- xmpDM:类型
- xmpDM:good
- xmpDM:乐器
- xmpDM:介绍时间
- xmpDM:曲调
- xmpDM:日志注释
- xmpDM:循环
- xmpDM:节拍数
- xmpDM:标记
- xmpDM:出点
- xmpDM:项目名称
- xmpDM:项目引用
- xmpDM:下拉菜单
- xmpDM:相对峰值音频文件路径
- xmpDM:相对时间戳
- xmpDM:发布日期
- xmpDM:resampleParams
- xmpDM:缩放类型
- xmpDM:场景
- xmpDM:拍摄日期
- xmpDM:拍摄日期
- xmpDM:拍摄地点
- xmpDM:拍摄名称
- xmpDM:拍摄编号
- xmpDM:拍摄尺寸
- xmpDM:扬声器位置
- xmpDM:开始时间码
- xmpDM:拉伸模式
- xmpDM:拍摄编号
- xmpDM:磁带名称
- xmpDM:tempo
- xmpDM:时间尺度参数
- xmpDM:拍号
- xmpDM:曲目编号
- xmpDM:音轨
- xmpDM:视频Alpha模式
- xmpDM:视频Alpha预乘
- xmpDM：视频Alpha统一性为透明
- xmpDM:视频色彩空间
- xmpDM:视频压缩器
- xmpDM:视频场序
- xmpDM:视频帧率
- xmpDM:视频帧尺寸
- xmpDM:视频像素宽高比
- xmpDM:视频像素深度
- xmpDM:作品集组成部分
- xmpDM:歌词
- xmpDM:discNumber

B.2.9. PLUS

仅支持以下PLUS属性：

- plus:交付文件名
- plus:首次出版日期

- plus:交付时图像文件格式
- plus:图像文件大小（交付时）
- plus:图像类型
- plus:版本

有关这些内容的更多信息，请参阅 http://ns.useplus.org/LDF/ldf-XMPSpecification。

附录 C：弃用注意事项

C.1. 构造状态

下表列出了随着本规范演进而状态变更的构造。使用以下状态值：

已弃用

Construct 已弃用（声明生成器不得生成此结构；验证器仍可接受此结构）。

UNDEFINED

构造符未定义（验证器必须忽略此构造符）。

<空白>

构造完全受支持（验证器必须接受它）。

表 19. 构造语法状态

构造	类型	v1.3	v1.4	v2.0	v2.1	v2.2
时间戳清单	清单	未定义	未定义	未定义		未定义
`urn:uuid` 命名空间	标签				已弃用	已弃用
`urn:c2pa` 命名空间	标签	未定义	未定义	未定义		
`c2pa.data` (数据框)	标签					已弃用
`c2pa.databoxes`（数据盒存储）	标签					已弃用
`sigTst` timestamp	时间戳				已弃用	已弃用
`sigTst2` 时间戳	时间戳	未定义	未定义	未定义		
`c2pa.claim`	断言			已弃用	已弃用	已弃用
`c2pa.claim.v2`	断言	未定义	未定义			
`c2pa.actio ns`	断言					

构造	类型	v1.3	v1.4	v2.0	v2.1	v2.2
c2pa.actions.v2	断言					
c2pa.asset-type	断言					已弃用
c2pa.asset-type.v2	断言	未定义	未定义	未定义	未定义	
c2pa.证书状态	断言	未定义	未定义	未定义	未定义	
c2pa.embedded-data	断言	未定义	未定义	未定义	未定义	
c2pa.font.info	断言	未定义		已弃用	已弃用	已弃用
c2pa.hash.bmff	断言	已弃用	已弃用	未定义	未定义	未定义
c2pa.hash.bmff.v2	断言				已弃用	已弃用
c2pa.hash.bmff.v3	断言	未定义	未定义	未定义		
c2pa.hash. 集合.data	断言	未定义				
c2pa.hash. 多资产	断言	未定义	未定义	未定义	未定义	
c2pa.ingredient	断言			已弃用	已弃用	已弃用
c2pa.ingredient.v2	断言				已弃用	已弃用
c2pa.ingredient.v3	断言	未定义	未定义	未定义		
stds.metadata	断言	未定义		已弃用	已弃用	已弃用
c2pa.metadata	断言	未定义	未定义			
c2pa.缩略图.声明	断言	未定义	未定义	未定义	未定义	
c2pa.缩略图.声明.*	断言					已弃用
c2pa.thumb 缩略图.ingredient	断言	未定义	未定义	未定义	未定义	
c2pa.缩略图.内容.*	断言					已弃用

构造	类型	v1.3	v1.4	v2.0	v2.1	v2.2
c2pa.时间戳	断言	未定义	未定义	未定义	未定义	
字体信息	断言	未定义	未定义			
stds.iptc	断言		已弃用	已弃用	已弃用	已弃用
stds.exif	断言		已弃用	已弃用	已弃用	已弃用
stds.schem a-org	断言		已弃用	已弃用	已弃用	已弃用
角色在区域映射	字段				已弃用	已弃用
action-items-map-v2 中的参与者	字段			已弃用	已弃用	已弃用
软件架构在 actions-map-v2 中	字段	未定义	未定义	未定义		
软件类型 entIndex 在 action-common-map-v2 中	字段	未定义	未定义			
在 action-items-map-v2 中更改	字段				已弃用	已弃用
action-items-map-v2 中的更改	字段	未定义	未定义	未定义		
instanceID in 参数 -地图-v2	字段				已弃用	已弃用
sourceLang 语言 参数 -map-v2	字段	未定义	未定义	未定义		
目标语言 uage in 参数 -map-v2	字段	未定义	未定义	未定义		
c2pa.train edAlgorith micData	数字源类型					已弃用

构造	类型	v1.3	v1.4	v2.0	v2.1	v2.2
http://c2pa.org/ digitalsourcetype/ trainedAlgorithmicDa ta	数字源类型	未定义	未定义	未定义	未定义	
http://c2pa.org/ digitalsourcetype/ empty	数字源类型	未定义	未定义	未定义	未定义	