



Coalition for Content Provenance and Authenticity

C2PA Implementation Guidance

2.2, 2025-04-22: Release

Table of Contents

- 1. Introduction 2
 - 1.1. Overview 2
 - 1.2. Scope 2
- 2. How to use this document 3
- 3. Architecture 4
 - 3.1. Assertions 4
 - 3.2. Manifest 5
 - 3.3. Ingredients 7
- 4. Guidance on the use of Content Bindings 8
 - 4.1. Guidance on Hard Bindings 8
 - 4.2. Guidance on use of Soft Bindings 9
- 5. Guidance on using various Assertions 15
 - 5.1. Overall guidance 15
 - 5.2. Actions 15
 - 5.3. Ingredients 16
 - 5.4. Identity Assertions 17
- 6. Trust 19
 - 6.1. Cryptography 19
 - 6.2. Digital Signatures 19
 - 6.3. Trust Model 21
- 7. Validation 32
 - 7.1. Validation security practices 32
 - 7.2. Validation of Ingredient C2PA Manifests 32
 - 7.3. Data validation using schemas 32
- 8. Distributed Ledger Technology 33
 - 8.1. Introduction 33
 - 8.2. Use Cases 33
 - 8.3. Decentralized Storage of C2PA Manifests 35
- 9. Additional Guidance 36
 - 9.1. GDPR 36



This work is licensed under a [Creative Commons Attribution 4.0 International License](https://creativecommons.org/licenses/by/4.0/).

Chapter 1. Introduction

1.1. Overview

The Coalition for Content Provenance and Authenticity (C2PA) has developed their [technical specification](#) for providing content provenance and authenticity through Content Credentials. It is designed to enable global, opt-in, adoption of digital provenance techniques through the creation of a rich ecosystem of digital provenance enabled applications for a wide range of individuals and organizations while meeting appropriate security requirements.

The specification has been, and continues to be, informed by scenarios, workflows and requirements gathered from industry experts and partner organizations. However many of these requirements are not normative in nature or may differ between organizations or workflows - in those cases it is important to provide non-normative guidance to implementers - which is the goal of this document.

1.2. Scope

This guidance document describes non-normative technical aspects of the C2PA architecture including construction and consumption of Content Credentials and their components including the use of digital signature technology for enabling tamper-evidence and establishing trust. It will also address areas where implementers can extend the C2PA architecture and its ecosystem.

The C2PA also created their [Guiding Principles](#) that address areas such as respecting privacy and personal control of data with a critical eye toward potential abuse and misuse. This guidance document also will help implementers to understand how these concerns should be addressed in their implementations.

One area of guidance that is not covered in this document is that of User Experience, as that is covered in a [separate document](#).

Chapter 2. How to use this document

Rather than reading this document from beginning to end, it is recommended that as an implementer is first designing each aspect of their solution, they should review the relevant section of this document to ensure that they take advantage of the guidance provided.

Chapter 3. Architecture

3.1. Assertions

3.1.1. Description

Each of the actors in the system that creates or processes an asset will produce one or more assertions about when, where, and how the asset was originated or transformed. An assertion is labelled data which represents a statement made by an actor about an asset. The assertion's label is [defined](#) either by the C2PA specification or an external entity.

3.1.2. Mandatory Assertions

Every standard C2PA Manifest has to contain at least one actions assertion that describes whether the asset is being created *de novo* (for example, as a result of performing a **File** → **New** operation in a creative tool, capturing a photo or video, or generating the media by a generative AI model) or is being opened for editing. In addition, when the creation process includes content, then a corresponding **digitalSourceType** field, with an appropriate value, needs to also be provided. This is to ensure that the asset's provenance is accurately captured from its inception.

EXAMPLE: A generative AI model generates a video in response to a text prompt. The resulting video asset's active manifest would have a **c2pa.actions** assertion with a **c2pa.created** action, itself having a value of <http://cv.iptc.org/newscodes/digitalsourcetype/trainedAlgorithmicMedia> in the corresponding **digitalSourceType** field.

EXAMPLE: The media desk at a newspaper wants to edit a photo that was captured by a photojournalist with a C2PA-enabled camera. The media editor opens the photo and applies crop and vignette operations. The resulting edited photo asset's active manifest has a **c2pa.actions** assertion with a **c2pa.opened** action pointing to an ingredient assertion for the original photo, where a **parentOf** relationship is indicated. It would also have actions for the cropping and vignette edits.

More information about action assertions can be found [here](#).

3.1.3. Encryption of Assertions

A [set of assertions](#), their associated labels, and its serialization (i.e., CBOR or JSON-LD) are defined in the C2PA specification. In order to change any of these, such as the data/schema or its serialization, it is necessary to use a new label so that the new information can be clearly identified as different from the original.

A use case for creating variants of existing assertions would be to encrypt them to prevent access to those not possessing the necessary decryption key. This might be for privacy protection or the establishment of a more secure end-to-end workflow.

For example, if an implementation from [Litware.com](#) wished to encrypt the **c2pa.metadata** assertion to hide the information about the capture device, they could create the new assertion label as **com.litware.metadata**

or perhaps even `com.litware.encrypted-metadata` if they wanted to even hide what was encrypted.

3.2. Manifest

3.2.1. General

A C2PA claim generator adds a new C2PA Manifest to the existing asset's C2PA Manifest Store to reflect whatever changes it has made. If anything needs to be removed, then those specific assertions are redacted and the redaction reflected in the new C2PA Manifest.

3.2.2. Frequency of Creation

The C2PA recommends that a C2PA Manifest be created for an asset when a significant event in the lifecycle of the asset takes place, such as its initial creation or an "Export" operation from an editing tool. As the creation of a C2PA Manifest is not a lightweight operation, due to the need to digitally sign the claim as well as potentially retrieve credentials from online services, it is recommended to do it as infrequently as possible. Additionally, the fewer the C2PA Manifests, the easier validation will be.

3.2.3. Standard vs. Update Manifests

Normally, an asset's digital content will be modified between "significant events". As such, a standard C2PA Manifest which includes hard bindings between that digital content and the C2PA Manifest are provided. However, there are times in an asset's life where a change is required to the C2PA Manifest but the digital content is not impacted. For example, the addition of some new assertion or even the redaction of an existing assertion. In those cases, an update C2PA Manifest is used.

3.2.4. Compressed Manifests

The C2PA specification allows for the use of compressed C2PA Manifests. This is useful when the size of the C2PA Manifest is large and the asset's own data is small (e.g., JPEG). Any C2PA Manifest can be compressed, even after creation, and stored in the asset's C2PA Manifest Store. When the C2PA Manifest is needed, it would be decompressed and used as needed. JUMBF URIs referring to the compressed C2PA Manifest itself or specific assertions within it work transparently regardless of the presence of compression.

NOTE | Compressed manifests are not compatible with BMFF-based hashing.

3.2.5. Time-Stamp Manifests

The C2PA specification provides for a special type of C2PA Manifest, called a time-stamp Manifest, which is similar to the "Document TimeStamp Signature" found in ETSI digital signature standards. These Manifests serve to allow a claim generator to provide "proof of existence" to the asset itself and the associated Manifest at a given trusted time. While it can be used at any time, it is most commonly used when a previous claim generator did not provide an embedded time-stamp in its claim signature (e.g., it was offline at the time of signing) and the current claim generator

wants to provide that proof.

3.2.6. Manifest Repositories

The C2PA architecture supports C2PA Manifest Stores external to the asset they are associated with, in the form of a manifest repository. This is useful when working with a file format that does not support embedding (i.e., text or XML) or when storing the C2PA Manifest Store separately improves workflows (e.g., searching on manifest data in a CMS).

When using a cloud service to manage the manifest repository, and serving the C2PA Manifest Stores via http Link headers (as described in the [C2PA specification](#)), having the C2PA Manifest Store available at the same origin as the asset is recommended as it would reduce requirements on [Cross-Origin Resource Sharing \(CORS\)](#). However, it is worth noting that doing so does not provide any additional privacy or security protection of either the C2PA Manifest Store or the asset.

To mitigate risks to user privacy, we recommend that manifest repository service providers allow content creators to retain the ability to redact or remove content from the repository, or to determine which C2PA Manifests may be publicly queried. For example, a content creator may want to include an information-rich C2PA Manifest for internal cataloguing purposes, and an updated C2PA Manifest with redacted information for public distribution.

3.2.7. Can an application remove an existing C2PA Manifest Store?

Completely removing an embedded C2PA Manifest Store from an asset is not recommended, unless the C2PA Manifest Store is being "externalized" - meaning that an embedded C2PA Manifest Store is replaced by a URI to an external location. This would be useful in scenarios where the size of assets is important, and compressed Manifests are insufficient.

NOTE

Implementation of the C2PA specification is to be evaluated in the context of the current industry practices around removal & detachment of metadata. In addition many publishers including the NYT are removing image metadata <https://blog.imatag.com/state-of-image-metadata-in-news-sites-2019-update>.

3.2.8. Can an application replace an existing C2PA Manifest Store?

Replacing an existing C2PA Manifest Store with a different C2PA Manifest Store is not recommended since doing so would completely change the provenance of an asset. However, there are some use cases where it could be appropriate. For example, a publisher may wish to remove all details about the capture and edit of an image and leave only their own publishing information.

3.2.9. C2PA Manifests for existing media

It may not always be possible (or practical) to embed a C2PA Manifest Store in an asset such as in the case of adding provenance information to assets that were created prior to the existence of C2PA. By creating an associated manifest repository for the asset and exposing its location via the methods described [here](#), all assets can have provenance, no matter their age.

3.3. Ingredients

Ingredients are key to the establishment of the provenance of an asset, by serving as a listing of what other assets went into the creation of the current asset. Each ingredient that is used can itself contain its provenance, thus creating a rich provenance for the current asset and all of its ingredients.

Each ingredient can either be documented as the `parentOf` the current asset, a `componentOf` that asset, or as `inputTo` the creation of an asset. The value of `parentOf` is used in the common case where one asset is opened in an editing application, modified, and then saved or exported as another asset. That original version is the `parentOf` the final (now current) asset. Alternatively, when one asset is created from a series of other assets (such as audio and video clips), those sources are identified as `componentOf` ingredients. `inputTo` ingredients are used when describing information that was provided to a generative AI model, such as a prompt or seed image.

Chapter 4. Guidance on the use of Content Bindings

4.1. Guidance on Hard Bindings

4.1.1. General

Every C2PA Manifest is required to have a hard binding to its associated digital asset. Use of hard bindings prevents collision-based attacks associated with soft bindings described below.

Selection of the specific hashing algorithm to use for a hard binding should be made based on the requirements of the workflow. In the absence of some compelling reason to do otherwise, it is recommended to use SHA-256.

4.1.2. Byte Range Bindings

The simplest type of hard binding that can be used to detect tampering is a cryptographic hashing algorithm over some or all of the bytes of an asset as described in [the core specification](#). Traditionally, this type of binding is done over an inclusive list of byte ranges of the asset. However, a number of attacks on an inclusion list-based approach were identified and it was determined that they are prevented by the use of exclusions lists. These vulnerabilities would have allowed content to be added to an asset that altered the digital content without altering the hard bindings.

IMPORTANT

Byte range bindings should only be used when one of the other types of hard bindings is not possible.

4.1.3. General Box Bindings

Some file formats, such as JPEG, PNG, TIFF, or GIF, use a general box structure, that can be used to improve the flexibility of the bindings over that of a byte range binding. Use of a general box binding is strongly recommended over a byte range binding for any file format that supports it.

A [general box hash assertion](#) consists of an array of structures, each one listing one or more boxes (by their name/identifier) and a hash that covers that data of those boxes, along with the algorithm used for hashing.

NOTE

Some formats, such as JPEG and GIF, require special handling.

4.1.4. ISO BMFF Bindings

ISO BMFF-based assets utilize a well defined box structure. Accordingly, the provenance content binding for these assets needs to be expressed in terms of that box structure.

ISO BMFF allows for either a single hash method that generates one hash for the entire data or a multiple hash method that creates a hash for each subset of the data. The multiple hash method is useful when some files contain very large 'mdat' boxes (e.g., large video or image files that can be downloaded and rendered progressively) or large

number of independent 'mdat' boxes (e.g., fMP4 where each fragment can be downloaded independently). In such cases, it is unreasonable to require a client to completely download all 'mdat' boxes before validating any part of the asset, making this method a practical solution.

A number of attacks on an inclusion list-based approach were identified and it was determined that they are prevented by the use of exclusions lists. These vulnerabilities would have allowed content to be added to an ISO BMFF asset that altered the audio/video presentation without altering the hard binding. This is why C2PA ISO BMFF hard bindings are integrated with the box structure using Exclusion Lists.

4.1.4.1. ISO BMFF Binding Exclusion Lists

Typically ISO BMFF content need only include the mandatory exclusions required by the C2PA specification. These should be sufficient for the most common brands registered with the [MP4 Registry Authority](#). However, careful consideration should be given to determine if addition boxes should be included for your ISO BMFF brand of asset.

Do not add a box to the exclusion list if:

- the modification of the box can alter the audio-video presentation in any way.
- the box declares content external to the asset which you wish to be tamper evident.

4.1.5. Hashing Assertions

The hashing algorithm used for hashing assertions should be the same as that used for content binding, as described [above](#).

4.2. Guidance on use of Soft Bindings

4.2.1. General

Asset metadata (including any C2PA Manifest Store present) may be routinely removed or corrupted by legacy or non-C2PA capable platforms during distribution. This is common, for example, on social media platforms that display asset renditions (e.g., altering the resolution, form factor or quality of the digital content) that do not have the appropriate C2PA Manifests declaring those modifications. Whilst these renditions may not create user perceptible change, they nevertheless change the underlying binary representation of the digital content.

Soft bindings provide a means for identifying the active manifest, and associated C2PA Manifest Store, that has become 'decoupled' from its associated asset in these circumstances.

Examples of soft bindings are content fingerprints (such as perceptual hashes) computed from the digital content, or imperceptible watermarks embedded within the digital content.

4.2.2. Manifest Repositories

Consider a data store or repository into which C2PA Manifests may be stored. A content creator may, at the time of

publishing an asset, opt in to the additional storage of that asset's C2PA Manifest Store into the manifest repository. For this workflow, the active manifest contains at least one soft binding - for example, an invisible watermark or perceptual hash (fingerprint) of the digital content.

Soft bindings may be used to identify C2PA Manifests that have become decoupled from their associated assets. When a Manifest Consumer encounters an asset with no C2PA Manifest, but would like information on the asset's provenance, they may compute a soft binding and use it to query the manifest repository. The manifest repository would return any C2PA Manifests that match that soft binding, and the Content Credentials may then be presented to the user for review.

An alternative application of soft bindings is to mitigate the threat whereby an attacker substitutes the C2PA Manifest within an asset with another valid C2PA Manifest in an attempt to explain that asset with false provenance. In circumstances where a Manifest Consumer wishes for further information on the asset's provenance, a similar query may be made using the soft binding to return alternative C2PA Manifests within the repository for the consumer's consideration. Information within the returned C2PA Manifests (such as timestamps or digital signatures) may inform subsequent trust decisions made by the consumer on that asset.

4.2.3. Durable Content Credentials

Soft bindings enhance the durability of content credentials by enabling the discovery of the C2PA Manifest Store if it is not embedded in an asset. There are two recommended ways to implement enhanced durability.

4.2.3.1. Invisible Watermarking

An invisible watermark may be applied to embed a unique identifier within the asset's digital content, using a watermarking algorithm on the [C2PA Soft Binding Algorithm List](#). The identifier may be used as a key to lookup the C2PA Manifest within a manifest repository. The C2PA Soft Binding Resolution API provides an interoperable specification that may be used to perform such a lookup across one or many manifest repositories in an interoperable way. The user may request retrieval of only the active manifest, or of the entire manifest store. The watermark thus enables retrieval and display of the missing Content Credentials.

In order to mitigate spoofing (transference) of the invisible watermark, a claim generator may optionally store a fingerprint of their asset as a soft binding assertion within the active manifest. At the time of lookup, the fingerprint within that active manifest may be compared against a fingerprint extracted from the watermark-bearing asset. The fingerprint algorithm that should be used to make this check is described within the soft binding assertion of the retrieved active manifest. If the fingerprints do not match, then the asset was modified or does not correspond to the original asset.

NOTE

In this design pattern, the search key is the unique identifier within the invisible watermark and the fingerprint is used only as an automated form of visual check. The use of watermarking, fingerprinting and embedding are three technology pillars reinforcing one another. An alternative implementation might not perform fingerprinting but instead return results directly for manual review.

4.2.3.2. Fingerprint Lookup

If a given asset has neither an invisible watermark nor an embedded C2PA Manifest, a fingerprint may be used as a fallback search key to discover the C2PA Manifest Store. The C2PA Soft Binding Resolution API specification supports discovery based on fingerprints as well. As algorithms used for fingerprinting typically offer near- rather than exact-matching, it is recommended that the Content Credentials retrieved using a fingerprint as a search key are presented to the user for manual review.

4.2.4. Illustrative Scenarios for the use of Soft Bindings

1. Recovery from stripping of metadata

Alice is a photojournalist, and captures a photo of an important event, editing it to enhance visibility of some of the content. Alice's camera device is C2PA capable, as is her image editing tool, and so a C2PA Manifest is added to document the capture and editing of her photo. Bob works for Acme Corp, a news publisher, who wishes to license Alice's photo for their publication. Bob decides to trust the content from Alice due to the presence of a C2PA Manifest documenting its provenance. Bob incorporates the photo into a composed image for his publication, using a C2PA capable editing tool. A soft binding assertion is computed by Acme Corp and added to the C2PA Manifest prior to signing. A copy of the C2PA Manifest is stored within a manifest repository maintained by a consortium of news providers. Bob publishes the photo and it is soon redistributed around social media.

Charlie is a news consumer and member of the general public. Charlie views a rendition of the photo on social media. The social media platform is not C2PA capable and no C2PA Manifest is contained within the rendition. Furthermore, the rendition of the photo has different resolution / form factor and changed by the social media platform.

Charlie wants to know about the provenance of the photo, since it documents an important event. Charlie right-clicks and submits the photo via a browser plug-in to a look-up service operated by a federation of news organisations, of which Acme Corp is a member. Charlie's browser software computes the soft binding of the photo and send it to that service. Charlie is directed to a web page generated by the service showing matching assets. Charlie visually verifies that the retrieved asset matches the photo he is interested in, and views the C2PA Manifest. Charlie uses the information in the C2PA Manifest to help make an informed trust decision based on the provenance of the photo.

2. Recovery from adversarial substitution of the C2PA Manifest

Alice is a citizen journalist, and captures a video of major civil unrest using a C2PA capable device, and edits it using C2PA capable editing software. After signing the C2PA Manifest in her video, a copy of the C2PA Manifest is stored within a manifest repository maintained by a consortium of news providers. The C2PA Manifest contains both a hard and a soft binding. Several years pass.

Mallory wishes to use Alice's video to substantiate his story about a recent civil unrest. Mallory strips the C2PA Manifest from Alice's video and substitutes his own C2PA Manifest. The C2PA Manifest is signed and the video asset is distributed online.

Bob is a news producer who receives Mallory's video. Bob suspects the video is fake news. Bob computes a soft binding of the video and submits it to a provenance service of which his organization is a member. The service retrieves the C2PA Manifest associated with Alice's video. Bob visually verifies the retrieved C2PA Manifest matches the video and validates the C2PA Manifest. Bob notices that Alice's C2PA Manifest contains signed assertions with a timestamp earlier than those of Mallory's video. Bob uses knowledge of this previous existing C2PA Manifest to help make a more informed trust decision on whether to trust the provenance of Mallory's video. Bob concludes not to trust Mallory's video, since an earlier C2PA Manifest explains the video with an alternative provenance trail.

3. Preserving provenance through non-C2PA capable toolchains

Acme Corp is a news broadcaster that runs a content production pipeline, but some legacy stages of that pipeline are not C2PA capable. Acme decides to use an invisible watermark to trace the provenance of content through these non-C2PA stages.

As an example, Acme receives a video from journalist Alice, containing a C2PA Manifest. Acme processes the video through the C2PA capable stages of their pipeline. As part of this pipeline, they add an invisible watermark to it and update its C2PA Manifest to record that action. This association between the C2PA Manifest and the invisible watermark is recorded separately. The video then passes through the rest of their production pipeline, during which the video's C2PA Manifest is lost but the watermark remains intact.

Prior to further use, Acme wants to attach the C2PA Manifest to the video again. They detect the invisible watermark, search their records, and retrieve the prior C2PA Manifest for it. That C2PA Manifest describes the provenance of the video from creation up to and including the watermark being applied. The Manifest is included as an ingredient in a new C2PA Manifest for the video, which then proceeds through the remainder of the final (C2PA capable) stage of the pipeline. The new C2PA Manifest includes the information that C2PA provenance was disconnected and then later retrieved by Acme.

The video is published and its provenance may be traced back to Alice by C2PA capable tools. The provenance describes a period during production when the content departed from and returned to C2PA capable tools. The fact that the content was managed by Acme during that gap provides a trust signal to end consumers of the content.

4.2.5. Implementation guidance

Soft bindings are not guaranteed to be exact, and so care should be taken in their use. Consider perceptual hashing (fingerprinting); a common form of soft binding algorithm. By design, multiple renditions of the same digital content may generate the same soft binding. However, different digital content, or renditions thereof, may generate the same soft binding either in error or due to attacks on the hashing function (for example, adversarial attacks on machine learning models). Therefore we make the following design recommendation on the implementation of soft bindings in C2PA.

1. Soft bindings must not be substituted for hard bindings in order to bind claims within a C2PA Manifest.
2. The matches made using a soft binding should be verified. For example, human-in-the-loop checking via a

thumbnail of an image stored within the C2PA Manifest to perform visual verification.

3. Services provided for the lookup of C2PA Manifests should declare the types of soft binding algorithm that may be used as query, using the unique identifier of that soft binding algorithm (per the [C2PA Soft Binding Algorithm List](#)).
4. It is recommended that claim generators that add soft binding assertions to an asset's C2PA Manifest do so as an opt-in addition and not make it mandatory.

4.2.6. Watermarks and Soft Binding Assertions

C2PA supports the use of invisible watermarks as one particular kind of soft binding. A `c2pa.watermarked` action must be added to the C2PA Manifest to signal that an invisible watermark was inserted into the digital content for the purpose of creating a soft binding. The watermark should reference the asset with a soft binding content identifier that can be queried via a manifest repository.

Each `c2pa.watermarked` action needs a corresponding `soft bindings assertion` to provide more details about the watermarking algorithm used to insert the invisible watermark into the digital content.

4.2.7. Trust and Privacy Considerations

To mitigate risks to user privacy, we recommend that the consumer should be informed explicitly (for example, via opt in) to the querying of the manifest repository. For example, a consumer may interactively initiate a query for an asset containing no C2PA Manifest in order to retrieve provenance information about that asset.

We also recommend that content creators be informed of the trade offs involved in using manifest repositories that allow for asset link-up with soft bindings; that is, on the one hand, identifying C2PA Manifests that have become 'decoupled' from their associated assets, while on the other hand, privacy risks that may result from a soft binding link-up to an earlier C2PA Manifest with, for example, redacted information.

To mitigate risks to user privacy and to preserve bandwidth, we recommend that the soft binding used to query the manifest repository is computed on the client side to avoid transmission of the query asset to the lookup service.

It is unlikely that a single centralized manifest repository will emerge for all content. Rather it is anticipated that a decentralized model will evolve in which multiple federated manifest repositories might emerge for different industry verticals, for example a coalition of news broadcasters might maintain a federated service for soft binding lookup based upon their own manifest repositories.

To promote the interoperability of independent services that query manifest repositories, we recommend that the C2PA Soft Binding Resolution API be adopted by implementors as a standard communication protocol for clients to send queries to the soft binding lookup services and for returning C2PA Manifest Stores to clients.

Trust in the lookup process is derived from trust in the integrity of the manifest repository. It may be desirable to use a decentralized, immutable data technology, such as a distributed ledger or blockchain, to underwrite the integrity of the manifest repository.

4.2.8. Soft Binding Algorithm List

C2PA is agnostic to the soft binding algorithm (for example invisible watermark or content fingerprint) used within a soft binding assertion. The soft binding assertion contains a field **alg** that serves to uniquely identify the algorithm used to compute the soft binding. The Soft Binding Algorithm List is an authoritative list of soft binding algorithm names that shall be used as identifiers within the **alg** field. Entries in the list also contain additional information on the algorithms.

4.2.8.1. Accessing the Soft Binding Algorithm List

The list is maintained as a JSON data structure by the C2PA at the following location: <https://github.com/c2pa-org/softbinding-algorithm-list>

4.2.8.2. Updating the Soft Binding Algorithm List

Developers of soft binding algorithms may request these be added as a new entries in the soft binding algorithm list. Developers may also request amendments to their entries. These requests may be made by submitting a pull request (PR) on the [C2PA github page containing the soft binding algorithm list](<https://github.com/c2pa-org/softbinding-algorithms-list>). The C2PA Technical Working Group may approve and merge PRs in accordance with its prevailing processes for approving technical contributions to the C2PA specification. C2PA's Technical Working Group may also decide to remove malicious or non-conformant algorithms from the list of approved soft binding algorithms.

Chapter 5. Guidance on using various Assertions

5.1. Overall guidance

5.1.1. User Generated Text

5.1.1.1. Generation

As part of a workflow, some claim generators may allow actors to enter arbitrary text as the value for fields in some assertions - for example, as the value of the `copyright` field in a custom metadata assertion.

The Signer of the C2PA Manifest needs to consider if they wish to be responsible for ensuring that the user generated text is something they are willing to take responsibility for, since that is a key role of the signer as described in our [trust model](#). If they are, then the assertion containing that text should go in the `created_assertions` list in the claim. If not, they the assertion should go in the `gathered_assertions` list.

5.1.1.2. Consumption

Manifest Consumers should consider doing any necessary "character filtering" on fields that could contain user-generated text, to determine if they contains characters could be used in a [code injection attack](#) when presenting that information inside of various UX frameworks.

Example

NOTE

For example, if a field is intended to contain a URL, then the consumer should ensure that the value does not contain any characters that could be used to inject malicious code. This is especially important for fields that are intended to be displayed in [HTML](#) since it allows for the injection of code that could be used to perform malicious actions.

5.2. Actions

5.2.1. General

As described in [the actions section of the C2PA specification](#), "an actions assertion provides information on edits and other actions taken that affect the asset's content." Each action declares what took place on the asset, along with various other (optional) information such as when it took place or what software performed the action.

It is recommended that actions be provided in the chronological order in which they took place. This order is first per action assertion - so the actions in `c2pa.actions` comes first, then those in `c2pa.actions__1`, etc. Then, within each action assertion, the actions listed in the `actions` field of the `actions-map` should be in the order in which they took place.

NOTE

As this is only a recommendation and not a requirement, Manifest Consumers should not rely on the order of actions to determine the order in which they took place. Instead, they should use other

methods, such as the **when** field of the action (if present), to determine chronological ordering.

The listed C2PA actions are those that have been defined to date as common operations for various types of software and hardware actors operating on various asset types. However, because not all possible operations can be anticipated, the C2PA specification also allows for the use of custom actions using the standard **custom labels syntax**.

When using one of the C2PA actions, you should choose the one that most closely represents the action being taken. If there is not something specific, then either use a more general one (e.g., **c2pa.edited** or **c2pa.filtered**) or a custom one that specifically describes your case.

Example

NOTE

A audio editor may perform an audio level operation, such as reducing the volume of a portion of the audio. This could either be described by the generic **c2pa.edited** action or via a custom action that is specific to the video editor, something like **com.fabrikam.adjustedLevel**.

When using a generic action or a custom action, it is recommended to add a free-text description of what the action does, in the **description** field. For example, a **c2pa.edited** action could have a description that says "Paintbrush tool".

5.2.2. Identifying use of AI/ML

Each action can identify whether it was performed by an AI/ML system through the use of the **digitalSourceType** field, which is used to identify the type of digital source that was used to create the asset, including the value **<http://cv.iptc.org/newscodes/digitalsourcetype/trainedAlgorithmicMedia>** which indicates that the asset was created by an AI/ML system and is therefore "trained algorithmic media". When an AI/ML performs an operation such as "inpainting" (i.e., where a portion of an image is filled in based on a text prompt), the action could be recorded as an **c2pa.edited** or **c2pa.placed** action with the **digitalSourceType** set to **<http://cv.iptc.org/newscodes/digitalsourcetype/compositedWithTrainedAlgorithmicMedia>**.

5.3. Ingredients

5.3.1. Identifying use of AI/ML

When an asset is the result of Generative AI, where the asset is produced as the result of inference on an specialized AI/ML model, it is important to be able to store not only that such a process took place, but also information about any input data to the model that was used, such as a prompt as well as the info about the model itself. This data can all be recorded as part of an ingredient assertion. An example of an asset produced from a text prompt can be found [here](#).

5.3.2. Thumbnails

When an ingredient is added to a C2PA Manifest, it can be useful to include a thumbnail of the ingredient, which can be used to visually represent the ingredient when the asset itself is not appropriate for a specific usage. The

thumbnail assertion should be added to the `created_assertions` list in the claim.

When doing so, however, a number of considerations should be taken into account in order to ensure that the thumbnail will be useful in downstream workflows and isn't necessary "just taking up space". Also, since the thumbnail is optional, the Claim Generator should consider if the thumbnail is necessary at all given the asset itself.

Some of the considerations that should be taken into account when creating a thumbnail are:

Format

In order to ensure that the thumbnail can be used in a variety of contexts, it should be in a format that is widely supported. It is also recommended that the thumbnail's format be one that logically represents the ingredient, such as a raster image for a video or a vector image for a document. It might even be useful to have a raster image for an audio, if the audio is from a known brand (e.g. podcast) or has a known visual representation (e.g. album cover).

Usability

The thumbnail should be a useful representation of the ingredient, and should be easily recognizable as such. For example, if the ingredient is a video, then a thumbnail of the first frame of the video might be a good choice, but a simpler animated raster image (e.g., GIF or APNG) could also be used.

If an ingredient doesn't seem to require a thumbnail, or there is a good use case for not including it - there is no need to do so. They are optional.

Size

A thumbnail should be small enough to be useful in a variety of contexts, but large enough to be able to be seen clearly. The size will depend on the context (incl. the rest of the data in the asset) in which the thumbnail will be used.

Security

Since thumbnails could be rendered by a Validator or other form of Manifest Consumer, it is important to ensure that the thumbnail does not contain any malicious code that could be used to exploit the system rendering the thumbnail. This is especially important if the thumbnail is in a format that could be used to inject code, such as an SVG or HTML file.

Additional Data

Some formats used for thumbnails can contain additional data that could impact either the size or the security of the thumbnail. For example, how useful is an alpha channel in a thumbnail? If IPTC or XMP metadata is included in the thumbnail, does it contain PII? These are all questions that should be considered when creating a thumbnail.

5.4. Identity Assertions

In some use cases, an individual or organization may wish to describe their own identity or assert additional metadata that can not be directly represented as the C2PA claim signer by the C2PA Trust List. Such an organization may wish to use an assertion that enables the identification of the actor(s) who are making these statements, along with the ability

for them to securely represent their relationship to the media asset and the specific assertions they wish to make. This is accomplished through the use of standard cryptographic techniques to sign (at least) the Content Credential's hard binding assertion, along with any other assertions that contain relevant information about or from the actor. An identity assertion should be used to provide attribution information about the media asset.

The credential holder's signature should generally be construed as reflective of the named actor's authorization of or active participation in the production of the media asset in which it appears. This signature should include an RFC 3161-compliant timestamp to provide an additional, independent signal as to when the credential holder generated the identity assertion.

An example of such an assertion is the [Creator Assertions Working Group's identity assertion](#).

Chapter 6. Trust

6.1. Cryptography

C2PA prescribes cryptographic algorithms permitted for hashing (or message digest), and digital signatures both of C2PA Manifests and of signing credentials. This list contains algorithms instantiated at multiple different security levels. Unless there are particular application needs or policy requirements that call for a higher security level, C2PA recommends using the following algorithms, key types, and key sizes.

For hashing, C2PA recommends using SHA2-256. Standalone hashes are used in multiple places, including hard bindings of content, lists of assertions in claims, and in `hashed-uri` and `hashed-uri-ext` structures.

When generating a key pair for certification in a signing credential, C2PA recommends an elliptic curve cryptography (ECC) key pair on the P-256 elliptic curve or the X25519 elliptic curve. If using ECC is not desired, as an alternate, C2PA recommends an RSA key pair with a modulus length of 2048 bits.

When choosing a signature algorithm for signing C2PA Manifests, C2PA recommends: * **ES256** (ECDSA with SHA-256) when using an ECC key pair on the P-256, P-384, or P-521 elliptic curves, * **EdDSA** (Ed25519) when using an ECC key pair on the X25519 elliptic curve, or * **PS256** (RSASSA-PSS using SHA-256 and MGF1 with SHA-256) when using an RSA key pair.

6.2. Digital Signatures

6.2.1. Revocation information and time-stamps

C2PA strongly recommends that claim generators retrieve and attach time-stamps and credential freshness information at signing time. This information should be added into the COSE signature as described in the specification.

If this is not possible, such as when a device is offline, then C2PA strongly recommends that a claim generator add a Time-Stamp Manifest on top of the previous C2PA Manifest as soon as possible.

NOTE

Attaching a time-stamp and freshness information to the signature allows validators to conclude the C2PA Manifest is still valid a) even if the signing credential has since expired or was revoked after signing time and b) without the need of an online query.

6.2.2. Protecting claim signing keys

In practice, C2PA claim signing keys will be issued to systems that perform claim signing operations. These systems may make these operations available to end users and/or be deployed to user-owned platforms (e.g., mobile phones). Issuance or disclosure of claim signing keys to malicious actors enables attackers to create claim signatures on arbitrary assets using the compromised identity. The resulting C2PA Manifests are valid in terms of the C2PA specification, but effectively allow for spoofing provenance.

It is therefore important that systems that manage C2PA claim signing keys adhere to security and key management best practices. This includes leveraging platform-specific features (e.g., hardware security modules and cloud key management services), minimizing key reuse, and revoking keys when compromise is suspected. For more information on key management, see the [NIST Key Management Guidelines](#).

6.2.2.1. Securing claim generation and signing operations

Some C2PA claim generation and signing systems may be exposed to untrusted users. Exploitation or misuse of these systems may allow attackers to create claim signatures on arbitrary assets using identities provided by the system. The resulting C2PA Manifests are valid in terms of the C2PA specification, but effectively allow for spoofing provenance. The impact of such an attack may be amplified if identities are shared between users, and/or if the attack goes undetected for an extended period of time.

C2PA claim generation and signing systems should consider industry best practices for information security, secure development and operation, and anti-abuse practices, including leveraging available platform-specific features for deployment (e.g., [Android SafetyNet](#), [Apple DeviceCheck](#) and [AppAttest](#)). The functionality present in [our attestations specification](#) can be useful here.

6.2.3. Verifying Suitability of Signing Credentials

If possible, a claim generator should attempt to validate that the signing credential used is suitable for the expected audience of validators for the asset. C2PA only supports X.509 certificates as signing credentials. A claim generator should be configured with the C2PA Trust List, as well as any additional trust lists and Extended Key Usage (EKU) lists as the validators. Then, at signing time, the claim generator should:

1. Validate that the certificate fulfils all the requirements for C2PA signing credentials as described in the Certificate Profile section of the Trust Model chapter.
2. Validate that a chain of trust can be computed from the signing certificate to an entry in a trust list using the contents of the **x5chain** COSE header, following the procedure in [RFC 5280 section 6](#). This header will include both the signing certificate and any intermediate certification authorities required to build a trust chain to the trust anchor.
3. Validate that the signing certificate is valid for one of the Extended Key Usages listed.

If the claim generator is not configured with the same trust lists and EKU list as the validators, it should not attempt any validation and signing with the provided credential.

If it is so configured and validation fails, the claim generator should warn its user with an explanation of the problem, but should allow the user to choose to proceed with signing. Users may choose to sign in situations where they know their audience has different trust lists or EKU list configuration.

6.2.4. Trust and Privacy Considerations

C2PA claim generation and signing systems should consider industry best practices for addressing common privacy concerns of its users.

6.3. Trust Model

C2PA follows a semantic standard adopted by existing systems that use sets of cryptographically-signed statements: such a set of statements are interpreted as spoken or asserted by the signer, and the relying party's acceptance of those statements depends on whether or not the receiver trusts the signer to make them. Because of this, the set of statements (assertions) are divided into two categories in the claim - those that were created by the claim generator and those that were created by another entity, but which the claim generator is simply included (gathered up) in the C2PA Manifest.

Relying parties must therefore consider which category each of these statements are coming from - whether created or gathered (and thus may actually be provided by other people or devices). Some C2PA Assertions allow stating that data is derived from a different source, or that certain actions were performed by particular actors. In all cases, it is the signer who is creating the Claim listing all the assertions and which type of assertion it may be.

In other words, a C2PA Manifest must be read as statements like the following:

- Signer asserts that this asset was captured with a "WonderCamera 2000."
- Signer asserts that this image was captured on January 1, 2020 at 12:00pm UTC, and this time was retrieved from a GPS receiver on the capture device.
- Signer asserts that this image was captured at latitude/longitude 80.0 degrees N, 10 degrees W, and this location fix was retrieved from a GPS receiver on the capture device.

The last two assertions are of particular interest because there is no separate proof, only the Signer's good word, that the time and location information were retrieved from a reliable source.

NOTE

C2PA has introduced [an accompanying standard for supplying attestations from secure hardware](#) that can provide a higher level of assurance that data was retrieved from trustworthy hardware. The subject of attestations is beyond the scope of this document; here we only consider assertions made by a single Signer.

X.509 certificates used in the secure server authentication and secure e-mail PKIs, code signing manifests, software bills of materials, and package manager manifests all use similar structures, where signers assert various properties about public keys, software packages, or other cryptographically-protected items. In all cases, it is trust in the Signer to make those assertions, as well as an identity ecosystem substantiating the signing key belongs to the Signer, that forms the basis for believing the assertions.

6.3.1. Trust Lists

The C2PA does not mandate the use of any specific "list of certificates or CAs that can be used to verify the trustworthiness of the signer of a C2PA Manifest". There exists a variety of complexities in choosing the membership for such a list, and implementers should understand them prior to the creation of their list. The C2PA will continue to improve their guidance in this matter as the ecosystem grows.

Although consumers should be able to modify the configuration of their validators, implementers should discourage

consumers from adding or removing individual trust anchors. If appropriate for the application, implementers should provide users with a selection of lists they can choose from. Certain applications may have only one list of trust anchors, and others may have more than one list of trust anchors.

6.3.2. Initial Design of Identities and Trust Anchors for an Application and its Ecosystem

C2PA does not mandate or even suggest particular trust lists or public key infrastructure (PKI), and instead takes them and other related configuration as inputs. This is because each application built using the C2PA standard that operates within its own ecosystem will have unique requirements and relationships amongst the participants of that ecosystem. Application implementers may be tempted to take existing trust lists used in other applications, such as those used for validating secure web sites or signed documents, and adopt such lists without due consideration. However, we strongly recommend performing the analysis described in this section to determine if the PKI associated with one of those applications is appropriate for the new application of C2PA. In particular, the participants and trust relationships among the entities in the web site and document signing PKIs will likely not match those for other applications and ecosystems, and so their use as trust anchors will be likewise incorrect for those applications. Trust lists are not "one-size-fits-all."

Signed provenance information transmits trust between a creator and a consumer, based on a trust relationship between those two that exists outside the scope of C2PA. Trust anchors operate by providing digital identities within a particular ecosystem that link to real-world identities, and perform an ecosystem-specific validation to ensure those identities are sufficiently trustworthy, and that consumers can be confident when an asset is verified as being signed by a known creator, they can rely upon their existing trust relationship with that known creator.

Mistakes and accidents do happen, and credentials sometimes need to be revoked before their natural expiration date. This is often due to the private key being mishandled or exposed. Although everything that key signed *before* its exposure can still be trusted, it should be revoked so nothing *after* that point is trusted. This requires the trust anchor that issued the certificate to operate a "revocation service." In C2PA, this service is primarily used at signing time to get a signed statement from the trust anchor stating the credential was still valid at signing time, which is then included in the C2PA Manifest, so validators can be assured the credential was valid at that time, whether or not it has since expired or been revoked.

When initially creating an application, its trust anchors, and determining how identities will be issued by those trust anchors, developers should answer the following questions:

1. Who are the creators of the content? These will typically be your signers, the recipients of credentials issued by the trust anchors, and the persons or entities who must be identified.
2. Who is the audience of the content, and how and where will they consume it? These are your consumers who themselves will not receive credentials, but will rely upon the identities presented in the credentials of others.

NOTE

In some ecosystems, persons or entities will act as both creators and consumers, and so these two groups may overlap. However, persons or entities operating as creators is separate from when they operate as consumers, and so it is clearer to consider these groups separately for the purposes of this analysis.

3. How does the audience already know and identify the creators? This will inform how the identity should be presented in the credential, and how it is presented to the consumers. C2PA publishes [User Experience Guidance](#) to help application developers determine how to present this information to the consumer.
4. What is the risk to the ecosystem if a credential is issued in error or is compromised, where a malicious actor obtains a credential belonging to someone else and is able to sign with it, such as the result of a key compromise? This will inform the required real-world validation to achieve the needed level of assurance, and the importance of implementing, operating, and querying a revocation service.

To illustrate this exercise, we now present example applications, and for each one, the answers to the questions above and the resulting trust model configuration.

6.3.2.1. News and Media Consumption

This is the application in which people consume news and other media from outlets which we might expect them to be familiar with, recognize, and potentially trust.

1. Creators of content are news and media organizations that are *uniquely identifiable*—even if the scope of the creator is intentionally focused on a particular population segment or geographic region.
2. Regardless of the creator's intended scope, particularly in today's connected world, the audience is the entire global population. Even if a consumer is not personally aware of an organization's name or brand, if they went in search of it, they would locate the same singular organization as everyone else.
3. The organization has developed a brand, an audience, and a reputation with their audience that leads to the audience identifying the organization. This does not exclude independent or unconventional creators, but does require they have built up enough of an audience and a reputation with their audience that leads to their being identifiable in this way, in the same way conventional newspapers, television stations, networks, and other organizations have done in the past.
4. Brand impersonation and incorrect attribution to the brand of content made by malicious impersonators can result in damage to the brand and the organization's reputation and trustworthiness amongst its audience. Moreso, as such a failure of the provenance ecosystem becomes public knowledge, this can undermine the provenance and authenticity ecosystem and open the door to further mis/disinformation, not only because an inauthentic asset has been signed by a trusted credential, but now consumers may come to believe they cannot trust those credentials. This is critical early on, when the ecosystem is building and working to establish that trust with consumers.

NOTE

Here we define *uniquely identifiable*. A *uniquely identifiable* organization has a unique name such that any two consumers with no relationship with one another, when presented with this name as the signer of an asset due to its placement in the signing credential, would resolve that name to the same real-world entity. Even though a consumer might not be aware of a content creator when presented with one of its assets, given this unique name, by "resolve" we mean they could track down the real-world organization through some out-of-band means. This would be the same organization any other consumer would find, as well, presented with that same unique name.

For example, although "CNN" could refer to multiple different entities, such as "CNN USA" or "CNN

International," it is these latter names that would be the globally unique name placed in the signing credential and presented to the consumer with the provenance information. Similarly, although multiple newspapers refer to themselves as "The Times," a globally unique name, typically with a geographic identifier, is used for a global audience. For example, "The Times of London" is different from "The New York Times" or "The Los Angeles Times," and each of these is uniquely identifiable.

In this case, the trust anchors will be a new set, which do not yet exist, operated by Certification Authorities (CAs) that employ a specific due diligence process in order to validate that an applicant for credentials identifying as one of these uniquely identifiable organizations is an employee or otherwise duly authorized representative of that organization.

The audience will consume the content either through a news or media consumption application or web site, or through social media. These applications or web sites can then be configured with the above trust list. Social media sites could be configured with multiple trust lists, and in the cases where an asset is signed by a signer in the news and media ecosystem, show that the digital content is provided by such an organization, as opposed to an authenticated user of the social media platform.

For entities and organizations whose identity is rooted in the identity of an online platform, such as a social media or video sharing site, an identity issued by that platform may be more appropriate. See the [Social Media and Video Sharing example below](#).

6.3.2.2. News and Media Creation

In contrast to news and media consumption, this is the application where a news or media organization internally creates and assembles a finished publication that is consumed in the previous example.

1. Creators of content are the subset of the employees of a news or media organization that are involved with publication activities. This includes anyone whose input appears in or otherwise impacts the final output of the organization: reporters, writers, editors, artists, designers, and so on.
2. The audience is the subset of the employees of a news or media organization that receive assets which will appear or otherwise impact the final output of the organization. This will typically be everyone who also counts as a creator, but also any personnel who only review or otherwise consume the content without making any changes to it, such as fact-checkers or supervisors who approve assets for public release.
3. The audience has the same employer as the creators, and rely upon the employer to provide a means to recognize and authenticate other employees.
4. Potential introduction of malicious material into the publication process. The precise risk will depend upon the access granted to someone with those credentials, but this becomes a particular case of the general problem of the credentials of any member of an enterprise being compromised.

In this case, the trust anchors will be the enterprise's own internal PKI, and existing processes for issuing digital credentials to employees can be used to issue these credentials as well. Such an enterprise is presumed to already have a process for connecting the real-world identity of an employee with their business records, and any credentials issued for use in that employment. As assets created in this scenario are generally intermediate outputs that will later

be combined into a final output, their useful lifetime is expected to be relatively short. Existing procedures for invalidating and revoking compromised credentials, and issuing new ones, should be sufficient to cover C2PA credentials.

The audience will consume the content through production tools and processes common within, and likely provided by, the organization. It can then be configured with the above trust list for ingesting assets received from others before using them in their work.

6.3.2.3. Insurance Underwriting and Claims Servicing

This application concerns the internal business processes of an enterprise; in this case, an insurance company and its business of underwriting policies and servicing claims. There are similarities with the News and Media Creation example in that this is also entirely internal to an existing enterprise.

1. Creators of content are the subset of the employees of an insurance company that generate photographs, video, or documents used in an insurance company's business practices, including but not limited to underwriting policies and servicing claims.
2. Consumers are the subset of the employees of an insurance company that use photographs, video, and documents generated by the creators in the insurance company's business practices, including but not limited to underwriting policies and servicing claims.
3. The audience has the same employer as the creators, and rely upon the employer to provide a means to recognize and authenticate other employees.
4. Potential introduction of malicious (and perhaps fraudulent) material into the insurance company's business processes, such as edited or fabricated evidence of a supposed claim for an insured hazard that did not actually take place, in order to commit insurance fraud.

As another enterprise case, the trust anchors here as well will be the enterprise's own internal PKI, for the same reasons as the case above. Similarly, the audience will consume the content through tools and processes common within, and likely provided by, the organization, which can be configured with the above trust list for ingesting assets received from others before using them in their work.

6.3.2.4. Social Media and Video Sharing

This application concerns the sharing of content across social media, even across multiple networks. This assumes the social media network used by the creator is globally known, but is not necessarily where the consumer consumes the content.

1. Creators of content are the owners of accounts in social media networks. This makes the assumption that each social media network is globally-known and unique.
2. Consumers are users of those social media networks, whether or not they have accounts, which covers the entire global population.
3. There are a few possibilities:

1. The owner of the account has built up a brand and reputation, in much the same way a news or media organization would, producing content from that account directly.
2. The social media platform asserts that, through some process or due diligence they have performed, that the owner of the account is a person or organization with an existing global identity that consumers would recognize.
3. The owner of the account is a person or organization with an existing global identity that consumers would recognize, and through their own established authenticated channels to their audience assert that they own a particular social media account.
4. Potential attribution of malicious content to the owner of a social media account. The precise risk will depend upon whether the account has any existing audience and trust relationship with their audience.

In this case, as the identifiability of the creator is rooted in the existing identity infrastructure of the platform, the platform makes the choice to issue C2PA-compliant credentials to the owner of the account, through whatever existing authentication infrastructure they already have for owners to access their accounts. Care must be taken to revoke credentials in case of account name changes, and timestamping is highly recommended so that names that are changed cannot have others later claim the same name and impersonate the former owner.

The audience will consume the content through the social media's web site or apps, which can be configured by the social media to use the above trust list. Media produced by users of the social media site can then be strongly attributed to the owner of that account. Further, social media sites could choose to incorporate each other's trust lists in a federated manner, so strongly attribute media to the owner of an account on one platform, even if the content is reposted on another platform. A social media site might also incorporate further trust lists to treat particular content specially, as described [in the News and Media Consumption example above](#).

6.3.2.5. Ad-Hoc Creative Collaboration

This application concerns the ad-hoc sharing of assets between creatives collaborating on a project, or communication between journalists and sources, using e-mail addresses as their primary means of identifying one another, whether or not e-mail is the principal means by which sharing occurs.

1. The creators are anyone with an e-mail address. So, anyone in the global population.
2. The consumers are anyone who knows a creator by their e-mail address. This is anyone in the global population generally, but for a particular creator this will be a small group who has a direct two-way personal connection to the creator.
3. The audience knows and recognizes the creators based on some pre-existing relationship by which e-mail is considered a reliable identifier and means of communicating with the person, based on an existing history of collaboration or a trustworthy introduction from another known person.
4. There is an impersonation attack here but the risk is low, as the audience for a particular creator is small. It's also unclear what benefit an attacker might have by impersonating a collaborative partner and sharing signed content that is in some way malicious. Malicious in this context does not mean technologically malicious in a way that could cause compromise of the recipient's device due to a software bug in consuming software, as that threat exists with or without a C2PA Manifest attached, but malicious in that the attacker is attempting to convince the

recipient that the asset came from the compromised creator.

In this case, the existing secure e-mail PKI can be used, with existing e-mail verification processes implemented by the CAs for issuing these credentials. As the risk of compromise of an e-mail account is significant, it would be unwise to rely on these credentials for long-lived assets or for applications where impersonation brings significant risk due in a scenario with particular sensitivity.

In following this example, implementers should be aware that secure e-mail certificates issued by the existing secure e-mail PKI will begin to follow the [CA/Browser Forum S/MIME Baseline Requirements](#) which were first adopted on January 1, 2023. Secure e-mail certificates issued before this date, as well as those issued after this date but before the issuing CA adopted these standards, will not follow these requirements, but it is expected conforming CAs will eventually do so. In particular, implementers should review chapter 3 and the identification requirements, and determine if those requirements are sufficient to meet the assurance and risk tolerance requirements of their application ecosystem. If they are, implementers may also wish to inquire with particular CAs as to the status of their implementation of these requirements, their process for identification before the standard requirements were introduced, and the expected time horizon during which certificates issued before the standard requirements ("legacy certificates") will remain valid. Implementers may choose to onboard only those CAs with a tolerably small or zero number of legacy certificates, and onboard further CAs as their implementation of the new requirements and remaining number of valid legacy certificates reach acceptable levels.

The audience will consume the content through whatever ad-hoc tools they use for their collaborations. In this case, the users may have to be educated enough to opt in to the particular trust list that contains the CAs which issue the credentials for their contacts.

6.3.3. Time-Stamp Authorities

C2PA encourages creators to retrieve a time-stamp for their assets in order to prove that it was published within the validity period of the signing credential, so that it can continue to be considered valid even after the signing credential has expired. Time-stamp authorities (TSAs) receive a special kind of trust where time-stamps they issue remain valid even after the TSA's certificate expires, so long as the attested time falls within the TSA's certificate's validity period. This combination allows an asset to be considered time-valid indefinitely. Without such a trusted time-stamp, consumers cannot trust an asset signed with a now-expired credential was not signed after the credential expired. Each application developer therefore needs to consider the list of trust anchors for TSAs, in addition to the C2PA signer trust list determined through the exercise described in the previous section. In this case, this trust anchor list does not define identity providers for signers of assets, but identity providers for TSAs that provide the time-stamps for assets which allow them to still be considered valid even after the signing credential has expired or been revoked. We recommend considering the following questions:

1. Are time-stamps needed? If the useful lifetime of assets will not exceed the lifetime of the signing credential, using time-stamps may not be required at all. Assets will become invalid when the signing credential expires, but in some applications this may be acceptable, or even desired. Drawing from our above examples, in the "News and Media Consumption" and "Social Media and Video Sharing" examples, these assets may be consumed years after their publication, which will require time-stamps. Other examples, like "News and Media Creation" and "Insurance Underwriting and Claims Servicing," may not have such a requirement. In these cases, the list of trust

anchors for TSAs can be empty, and those applications can skip retrieving time-stamps.

2. Who will provide them? Many operators of Certification Authorities in the PKI used for secure web sites also operate time-stamp authorities, possibly free of charge. Signed time-stamps are considered valid forever, when the attested time is within the validity period of the TSA's signing credential, and so operating TSAs involves heightened security requirements that may be impractical for some entities, and so may choose to use one of the public ones and so the secure web PKI trust anchor list. Others who operate their own IT infrastructures or even data centers may choose to operate their own, and use their own trust anchors.

6.3.4. Ongoing Maintenance of an Identity Ecosystem

The previous sections of this guidance have addressed the creation of a new application, and the initial creation of identity credentials and trust, before it is up and running, so that it will establish itself as reliable with its users. That is, however, only the beginning. Operating and maintaining the trust and security of such an application, its trust model, and identity ecosystem is an ongoing process throughout the entirety of the application's life. Not only do credentials require regular renewing, it is expected that there will be failures and breaches.

Credentials are time-limited both to encourage regular rolling of cryptographic keys, and to encourage regular re-confirmation that the holder of a credential continues to be authorized to hold it, for whatever the definition of "authorized" is in the context of a particular application. For example, in enterprises, this generally takes the form of using the expiring but still valid credential to authenticate a renewal request, combined with verifying the requestor is still an employee and in a proper role for that credential. For public-facing credentials, a repetition of some or all of the due diligence described earlier may be warranted.

When a credential is known to be compromised, the only available mechanism for remediation is for the issuing CA to revoke the credential. C2PA strongly recommends but does not require creators to retrieve credential revocation information. This information is a signed statement from the CA that the credential was not revoked at the time, which combined with the time-stamp allows a consumer to conclude the credential was valid at signing time, whatever the status at consumption time. Because this information is not required, it may not be present, and for privacy reasons, it is optional for a consumer to make a live query to the issuing CA to inquire about the credential's revocation status. By its very nature, credential revocation is a "best-effort" process and consumers cannot be guaranteed to immediately ascertain that a credential has become revoked. In considering risk in the exercise above, the initial and renewal issuance processes must be tailored to minimize the risk of mis-issuance to an acceptable level, and holders of credentials must exercise care in the control of their credentials to avoid their exposure or theft. However failures must be expected, and so when one is detected, applications and their identity ecosystems must have developed processes for revoking credentials.

Although C2PA does not make such requirements, an application may choose to impose stronger requirements if warranted by the risk of credential compromise. For example, an application may require attached credential revocation information and either require rejecting C2PA Manifests that lack them immediately, or require the online query to the CA at consumption time.

6.3.5. Use of the Private Credential Store (also known as the "address book")

The identity of signers will typically be established by identity providers through the use of trust lists, as there is no expectation that signers and validators will be directly known to one another. There are exceptions, however, where two users who do know one another will want to be able to validate assets signed by the other. For example, a journalist working for a well-known media organization may have a signing credential for their organization, but still wish to receive C2PA-protected assets from sources who are otherwise anonymous. The source's validator may already trust the journalist's credential based on its issuance from a trusted identity provider, but the source almost certainly will not have such a credential. In this case, the source can generate their own signing credential. Such a credential would not be trusted in general, as it is not issued by an identity provider in the application's trust anchor list, but in this special case it can be communicated to the journalist, who can elect to add it to their private credential store.

The private credential store is only usable for the purpose of directly trusting assets signed by a credential that would not otherwise be accepted. Credentials in the private credential store cannot act as identity providers and issue credentials for others.

Implementers who want to provide this functionality should allow such a consumer to generate a credential with a simple user interface. In the case of X.509 certificates, this takes the form of a self-signed certificate. Implementers should follow the Certificate Profile in the specification when generating the certificate. Consumers can have the option to provide personally identifying information to be placed in the certificate if desired, but should not be required to do so; where such fields are required in the credential, the implementation can place non-identifying default values. Implementers should also allow the consumer to choose a validity period for the certificate after which it will become invalid, to suit the expected length of time the credential will be required. C2PA recommends a suggested default of one year.

Implementers should also consumers to import and export these credentials, but stress that importing such a credential should only be permitted if the consumer has personal knowledge the credential originates from the known source. Consumers should also be directed to remove credentials when assets signed with them no longer need to be validated, or the consumer learns the signing credential has been compromised or lost.

Implementers should enforce time validity periods on credentials in the private credential store, and either warn about or reject C2PA Manifests signed by credentials that have expired. Implementers may provide functionality to time-stamp a single asset or otherwise mark it as requiring preservation, and continue to validate that particular C2PA Manifest when needed.

Consumers should only accept credentials from others with whom they have an existing relationship and an out-of-band reason to believe the credential belongs to the intended subject.

6.3.6. Signing and Transiting Trust in Anonymous and Pseudonymous Scenarios

C2PA transits trust that already exists for a consumer of a particular publisher. The typical case is where the publisher is a person or an organization using a real or legal name. There are circumstances where a publisher may not want to reveal personally identifying information that links their digital identity to the real world; whether simply to protect

the privacy of their real lives from their online ones, to protect themselves from scrutiny or harassment due to content posted online, or even to shield themselves from the actions of repressive governments by documenting or reporting on newsworthy events.

Trust cannot spring forth from nothing, however; technology cannot create it, only augment and transit it. A C2PA Manifest signed by an untrusted signer has no more value than any other documentation of an asset's provenance offered without any proof. But even if there is no trust relationship for a consumer to a publisher based on a real or legal name, there are still other trust relationships that can be leveraged to provide some value.

6.3.6.1. Trusted Third-Parties

In [the social media example above](#), we have already explored the pseudonymous case. In these cases, similar to legal identities, the trust relationship is built by the pseudonym's advertised activity: the content published and subsequent reputation established by the pseudonym's owner. The trust is not rooted in an established legal identity, and therefore a credential that establishes a trustworthy ownership of the pseudonym is sufficient.

The truly anonymous case, where any information about the human actor publishing the content cannot be shared, provides a greater challenge. Absent any ability to establish a basis for trust with anything about the publisher, we must identify other potential sources of trust. In these cases, a common case is establishing that the asset is a true capturing of real-world events, without any malicious edits intended to create disinformation. The first potential method is through the use of a *trusted third party* (TTP). The [news and media consumption example above](#) is a special case of a TTP. In this case, media provided by sources that must remain anonymous is done so privately to a journalist or tip line, who then performs their own due diligence to confirm the accuracy and veracity of the asset's content, and any additional information provided by the source. It is the trust of the consumer in the news or media organization that then becomes relevant, stemming from trust in that organization both to protect the privacy of its sources and to do its own independent due diligence on material it receives.

A more nuanced option is a non-governmental organization or other intermediary that works with people in locations where conditions make them want to remain anonymous even as they provide images or videos of newsworthy events. This may be a lower level of assurance than a consumer may expect from a news organization, but it may be sufficient to be assured that the anonymous publisher is actually in the location where the media purports to originate from, although consumers might still view it with a healthy level of scepticism as a result. As with typical publishers, the consumer's relationship with the signer and understanding of what promises that signer makes informs the consumer's decision about the content.

In all cases, however, the consumer's trust lies in the trusted third party, and the trust that the TTP performs suitable due diligence to ensure what they are publishing on the anonymous source's behalf is what they claim it to be.

6.3.6.2. Trustworthy Devices

A developing possibility is through the use of trustworthy, tamper-resistant devices that attest to the genuineness of what is captured. Mobile devices with cameras have had tamper-proof security hardware for some time now, and such hardware is beginning to find its way into higher-grade image and video capture equipment, and these devices can do the signing. Manufacturers of such devices can encourage their use by properly documenting the capabilities

of such devices; the assurance level of the entire path from image or video acquisition to C2PA Manifest signing; the expected resistance of the device to attackers, especially attackers with unrestricted physical access to the device, both in terms of attacks resisted and the time duration it can be expected to do so; and penetration or other testing performed on the device in support of the security property of the device. Devices will vary in such capabilities, and manufacturers should provide sufficient information to allow informed decisions about their devices.

Manufacturers can issue signing credentials to the secure hardware on their devices to enable signing of C2PA Manifests, with whatever metadata (such as time and location) the device is capable of reliably capturing, as well as enabling secure attestation of the device and its capabilities, which can also be included in the C2PA Manifest. This can be paired with *assurance services* operated by the manufacturers to maintain up-to-date information about the health and security of their devices. Manufacturers should ensure a device's capabilities, particularly its limitations, are well-known, so that successful attacks do not damage the manufacturer's reputation or the entire ecosystem. In particular, manufacturers should be mindful of how reliable a device's source of time or location information may be or how vulnerable they may be to attack, causing the device to attest to falsified information, in addition to standard resistance to having its secret key extracted.

This information will be useful to the implementers of identity ecosystems for C2PA assets, as they consider including trustworthy, tamper-resistant devices. These ecosystems will have particular security, cost, and time requirement trade-offs for such devices. These implementers will make these informed choices on behalf of the consumers in their ecosystem, using the documentation and other information made available to select the most suitable choice or choices. Though the capabilities of the device will of course be particularly relevant, implementers of ecosystems should also consider the duration for which a device can be considered trustworthy, particularly when considering how likely such a device may be to come into the possession of an adversary, and the expected capabilities of those adversaries. This may inform a need for signing credentials of short duration that are regularly renewed, or signing credentials which are non-renewable and thus the device becomes untrusted past a certain point in time; the use of aforementioned assurance services to establish ongoing trustworthiness of the device at signing or consumption time if possible, or at credential renewal time if not; the robustness of the revocation infrastructure for its credentials; and the robustness of the process of alerting the issuer that the device may have become compromised before its credential's expiration.

Chapter 7. Validation

7.1. Validation security practices

Special care should be taken when implementing validators. Like other software that processes untrusted input, validators may be the target of memory safety attacks, parser attacks, request forgery attacks against adjacent systems (e.g., when retrieving remote content or decoupled C2PA Manifests), information leaks (e.g., via OCSP queries), denial of service attacks, and so on. Thus, it is important that these validators adhere to secure development and operations practices associated with their respective execution environment.

A Manifest Consumer that is performing validation (e.g., a web browser) may detect and mitigate attempted compromise of C2PA Manifests or even the complete removal of C2PA Manifests. It is recommended that Manifest Consumers consider forthcoming C2PA User Experience guidance, retrieval of decoupled C2PA Manifests via [soft bindings](#) when appropriate, and other forthcoming C2PA recommendations to mitigate the impact of these types of attacks.

7.2. Validation of Ingredient C2PA Manifests

As described in the [Validation section of the specification](#), "The validator may optionally recursively validate the ingredient's ingredients". To do so, the implementation resolves each ingredient's `url` field to find the next ingredient in the chain. It is possible that an infinite recursion situation could occur during this resolution process (whether constructed on purpose as a DoS attack or not). Implementations should be careful to check for such situations when performing this recursive resolution of ingredients.

7.3. Data validation using schemas

The C2PA Manifest consists of data that has been serialized into either CBOR or JSON-LD. The schemas (in CDDL and JSON Schema, respectively) have been published as part of the C2PA specification website. However, those schemas exist to help implementers create valid data to improve interoperability - they should not be used as part of the standard C2PA validation process.

NOTE

A C2PA Validator can choose to offer schema validation as an "extra", but the results of that validation would not effect the validity of the C2PA Manifest.

Chapter 8. Distributed Ledger Technology

8.1. Introduction

Distributed Ledger Technologies (DLTs) enable multiple parties to collaborate to produce a tamper-evident, distributed data store.

DLT enables a ledger to be shared across a set of DLT nodes and synchronized between the DLT nodes using a consensus mechanism.

In such a distributed system, control is distributed among the persons or organizations participating in the operation of the system (ISO 22739:2020).

Data stored on a DLT is immutable; once committed to a DLT, data cannot be changed or deleted. The ordering of data stored within a DLT is also immutable.

C2PA Manifests store data on asset provenance that in most cases should similarly be immutable. However redaction mechanisms exist to remove past assertion data from a C2PA Manifest. For example, to ensure the privacy of a creator, removing identity data from the assertion store and updating the C2PA Manifest to record the event of that redaction. Other circumstances that may involve redaction could be the removal of personally identifiable information (PII) to comply with relevant legislation on data protection. While the C2PA redaction mechanism provides for the deletion of data, the prior existence of that data and the act of redaction will be visible to the manifest consumer.

For this reason we make a general recommendation that C2PA Manifests should not be stored on DLTs, since the data immutability guarantees of DLTs prevent redaction of C2PA Manifests stored within them.

DLTs may, instead, be used to underwrite the integrity of a manifest repository containing C2PA Manifests (e.g., a cloud database). One common use case would be where a hash of a C2PA Manifest, or other cryptographic proof, may be stored immutably within a DLT. This may be used to prove that the C2PA Manifest has not been altered, or deleted (non-repudiation).

On-chain storage of C2PA Manifests may still be suitable, however, for use cases where redaction is not specific consideration and where on-chain storage requirements are not prohibitive.

8.2. Use Cases

We outline several possible ways that DLT may be used to implement or instantiate aspects of the C2PA specification:

1. Underwriting the integrity of manifest repositories

Consider the case of a manifest repository, where C2PA Manifests might be stored decoupled from the digital content they describe. Such a manifest repository may be used to store C2PA Manifests and query those C2PA Manifests via a lookup service using a soft binding, for example to recover provenance for assets where C2PA

Manifests have been removed or corrupted.

The user of such a manifest repository trusts the governance of that manifest repository operator not to manipulate or remove C2PA Manifests stored within. In other words, trust is centralized within the provider of the manifest repository. A DLT may be used to store hashes of C2PA Manifests as they are committed to the manifest repository to assure consumers of the integrity of that manifest repository without the need to trust the manifest repository provider. In other words the trust placed in the centralized maintainer of the manifest repository is replaced by trust in the decentralized governance of the DLT.

Such an implementation does not guarantee the persistence or availability of the manifest repository.

NOTE

In order to adhere to the [C2PA Guiding Principles](#) on Cost Burden and Performance, it is suggested that such proofs are rolled up in batches per a Layer 2 DLT solution in order to scale, and that an efficient consensus mechanism for DLT (such as proof of stake, or proof of history) is used to mitigate adverse cost or energy usage.

2. Decentralized claim signing

A smart contract is a computer program stored in a DLT system wherein the outcome of any execution of the program is recorded on the distributed ledger (ISO 22739:2020). Smart contracts may be configurable via a tokenized consensus mechanism. For example, a smart contract that may be upgraded or configured according to a vote by holders of a particular cryptographic token. Such contracts are referred to as 'decentralized autonomous organizations' or DAOs. Like regular programs, a DAO may be used to store and process data and even take payments for doing so.

A DAO might be set up to sign claims autonomously, according to a certificate installed by its operators. This would provide a decentralized alternative to the claim signing services run by centralized organizations tied to particular geographies or legislative zones.

3. Decentralized trust list management

A DAO might be used to set up and manage a certificate trust list. Signing of claims is grounded in public key cryptography rooted in a trust list managed by a federation of partners. Since C2PA allows for the existence of multiple such trust lists, the DLT may be leveraged to manage the trust list via a tokenized governance system. This might be attractive to content creators and consumers wishing to utilize a decentralized governance for such a list, and in turn agency over the issuance and revocation of signing certificates

4. Federated lookup for soft-bindings

A related application of DLT to soft binding is in the creation of a federated lookup service for soft bindings. Soft bindings are identifiers derived from digital content that enable matching C2PA Manifests to be recovered from within a manifest repository.

Due to the existence of multiple independent repositories, it is desirable to orchestrate search across repositories without requiring centralization of control in any given repository.

A DLT may be used to store directly a key-value index that maps soft bindings to the URLs of manifest repositories and/or a unique identifier that may be used to recover the relevant C2PA Manifest.

A specification for decentralized manifest recovery is provided by the [C2PA Decentralized Soft Binding Resolution Process](#).

Alternatively a DLT may be used to provide a decentralized list of such service endpoints for a given vertical (e.g. journalism), updated via a DAO managed by a consortium of implementers representing that vertical.

8.3. Decentralized Storage of C2PA Manifests

- + Decentralized storage may be used to store C2PA Manifests or cloud assertion data, as with other cloud technologies, via a hashed URI.

- + For example, a cloud data assertion may specify a hashed URI pointing to data stored on a decentralized file system such as the [Inter-Planetary File System \(IPFS\)](#).

- + Some decentralized storage, such as IPFS, uses URIs derived from a hash of the stored content. URI references can be made also via indirection, such as via the [Inter-Planetary Naming Service \(IPNS\)](#) in which case the immutability of the referenced content is maintained through the hashed URI.

- +

Chapter 9. Additional Guidance

9.1. GDPR

Items for future guidance

NOTE

Another topic needs to be around GDPR and other related legal aspects. This is currently being tracked as issue #114 the C2PA Specification github repo.