# Proof of Concept for a Decentralized C2PA Service for Asset Generation and Provenance

Florin-Dorian KELEMEN, Mihai TOGAN, and Emil BUREACĂ

***Abstract*—This paper focuses on the development of Proof of Concept applications that manage and can attest to the provenance data associated with digital content such as images, videos, audio files, and PDF documents, in accordance with the C2PA specification. This is achieved through a decentralized service, ensuring transparency and traceability of all modifications made to multimedia content through these applications. As a result of the implementation, a suite of functional applications was developed, demonstrating how other existing software products that handle digital assets can integrate a decentralized service incorporating the C2PA specification. This contributes to combating fraud and, at the same time, supports the recognition of the contributions of creators and collaborators involved in the production of digital resources. The current implementation proposes a modular architecture, taking into account today's dynamic and diverse technological environment. This enhances adaptability to future changes that may arise from subsequent implementations inspired by the current solution.**

***Index Terms*—C2PA, decentralization, digital content, provenance, service.**

## I. Introduction

In a rapidly expanding digital world, the creation of digital content is becoming increasingly common and accessible. Content creators are growing in number, with the opportunity to showcase their work to a vast audience or other types of consumers – often benefitting from financial incentives for their creations. The existence of such financial motivation naturally leads to the emergence of impostors who attempt to alter or even directly plagiarize original content, promoting it as their own. Frequently, this content is further commercialized, accompanied by an artificial narrative designed to capture the attention of consumers - regardless of the story's accuracy or authenticity. The rise of artificial intelligence-based technologies significantly contributed to such practices [1]. At the same time, AI-powered tools like DALL·E or Adobe Firefly make it increasingly easy to generate and modify existing content. This altered content can then be paired with false – but convincing – narratives that distort reality, strongly influence public opinion about events, and contribute to the spread of fake news. Such content holds the potential to sway individuals or even affect the course of large-scale events. Concrete examples include the use of AI tools to create realistic videos replicating the appearance, voice, and behavior of world leaders making specific statements [2]. The technological barrier for generating such digital content is extremely low, allowing even inexperienced users to produce these materials.

What if, when viewing an image shared on social media, we could also access relevant information about its provenance – such as who or what created it, when it was captured, whether it was edited, and the digital journey it has taken before appearing on our screens?

In response to the challenges outlined above, this work implements a proof-of-concept service focused on ensuring transparency, authenticity and verifiability of digital content provenance. The main contributions are as follows:

1. C2PA standard integration: integrated the C2PA (Coalition for Content Provenance and Authenticity) specification [3] to attach secure provenance metadata to digital assets, enabling the inspection of origin, edit history, contributors, and potential involvement of non-human agents in the content creation process.
2. Decentralized metadata anchoring using decentralized networks: addressed the limitations of centralized metadata storage by integrated decentralized technologies such as blockchain and distributed storage for storing and verifying provenance-related metadata. Each content event, such as creation, modification or publication, is recorded as an immutable and publicly verifiable transaction, significantly reducing attack surfaces such as database breaches or identity spoofing.
3. Modular and transparent architecture: the system is designed to operate independently of existing editing or publishing platforms, enabling potential for seamless integration without disrupting current workflows. The solution relies on external, specialized services exposed via configurable APIs, promoting low-friction adoption.
4. Minimal invasive integration: the provenance tracking mechanisms have minimal impact on existing application codebases, supporting interoperability across platforms and reducing the need for duplicate implementations.
5. Provenance-aware ecosystem foundation: laid the groundwork for a trust-enhancing digital ecosystem by combining open provenance standards with a decentralized infrastructure.

The implementation discussed in this paper is publicly available on GitHub at the following repository: https://github.com/kelemangiar0/dAtest.

Funding: This research was funded by Executive Agency for Higher Education, Research, Development, and Innovation Funding (UEFISCDI), Romania, grant number 51PTE/2025.

F. D. KELEMEN was with Military Technical Academy "Ferdinand I", Bucharest, Romania (e-mail: kelemendorianflorin0@gmail.com).

M. TOGAN is with Military Technical Academy "Ferdinand I", Bucharest, Romania (e-mail: mihai.togan@mta.ro).

E. BUREACĂ is with Military Technical Academy "Ferdinand I", Bucharest, Romania (e-mail: emil.bureaca@mta.ro).

## II. Related Work

From a technical standpoint, there are various mechanisms that can ensure the integrity of digital content. Among the most commonly applied methods today are [4]:

- Digital signatures, which are applied directly to content and enable the verification of data integrity as well as the identity of the signer;
- Digital watermarking, either visible or invisible to the human eye, embedded within the content to help identify the creator or deter unauthorized use of that content;
- Storing an identifier (such as a hash value) or a set of content-associated identifiers on a decentralized network, with the purpose of ensuring traceability and authenticity of the original content, verifiable at any time.

Although these methods offer a technically sound foundation for verifying the authenticity of data, they fall short when addressing modern challenges such as media manipulation or the generation of synthetic content using artificial intelligence. Some of the main limitations include:

- Lack of historical content, since content authenticity is typically validated at a single point in time, without providing insight into successive edits or the full provenance chain of the digital asset;
- Susceptibility of watermarks to loss, due to visible watermark degradation from editing, compression, format conversion, or even deliberate removal of authenticity-related metadata.

To address these shortcomings, recent years have seen the development of technical specifications aimed at ensuring digital content authenticity in a more robust, transparent and interoperable manner. These specifications typically combine digital signatures, hashing and watermarking into a unified, standardized and verifiable framework. A relevant example, discussed in this paper, is the C2PA specification. Currently, the C2PA specification is supported by major players in the global tech industry, highlighting its relevance and standardization potential. Key contributors include: Microsoft, which integrates C2PA metadata into its AI-powered services such as DALL·E and Azure OpenAI [5]; Adobe, which has implemented support for the standard in commercial products such as Photoshop and Firefly [6]; Cloudflare, which announced plans to support C2PA, becoming one of the first major Content Distribution Network providers to recognize and preserve C2PA metadata in distributed media [7].

The C2PA standard addresses one of the core limitations of traditional methods: the absence of historical context in digital content. By embedding a verifiable chain of edits, authorship and signatures, C2PA enables the full traceability of a file's lifecycle. For example, if an image is generated using DALL·E and later edited in Photoshop, C2PA allows the reconstruction of the full modification history.

However, several challenges still affect the wide applicability of this specification. Firstly, there is no standardized infrastructure that enables uniform validation of authenticity metadata. For instance, there are discrepancies between the way C2PA metadata is presented in CAI's (Content Authenticity Initiative) verifier [8] and Adobe's verification tool [9]. These differences affect the actual display and presentation of metadata to the end user, as can be observed in Fig. 1.

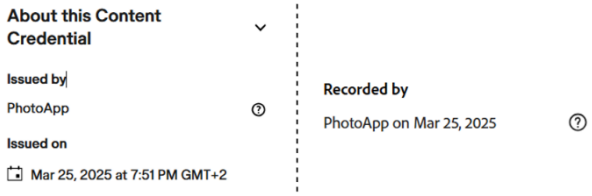


Figure 1. Comparison between C2PA metadata presentation in CAI verifier (left) and Adobe verifier (right)

Secondly, even properly embedded or referenced metadata is vulnerable to removal. A malicious actor can easily strip metadata by converting the file to another format or manipulating its binary structure, effectively erasing its provenance and authenticity trail. Moreover, if a file passes through an application that is incompatible with the C2PA standard and fails to recognize, preserve or correctly process the metadata, all provenance context may be lost, even in the absence of malicious intent. These challenges are illustrated through this work.

In response to the issues related to the lack of a transparent public infrastructure and vulnerability of metadata removal, complementary initiatives and solutions have emerged, the most mature being Numbers Protocol [10]. With the help of the Capture solution [11], the protocol extends the protection of digital content authenticity by integrating a decentralized, blockchain-based registry. Thus, even if C2PA metadata is stripped, files processed through Capture or compatible Numbers Protocol tools are also recorded in the Numbers Archive [12]: an open, decentralized, public and interconnected infrastructure that enables users, content creators and developers to register, track and validate media assets through a global trust layer. This approach bridges provenance metadata with a globally distributed and immutable record, strengthening digital authenticity in ways that are no longer solely dependent on the integrity of the original file.

## III. Methods

The solution is designed to be used by two main categories of actors: digital content consumers and digital content creators, as mentioned in the C2PA specification.

### *A. Consumers*

In the implemented system, consumers are users or software applications that verify the authenticity and provenance of digital content according to the C2PA specification. Consumers may become content producers – consciously or not – by taking a photo or editing a file. The web application plays the role of the consumer in the current implementation by providing an interface for uploading media files alongside their associated C2PA sidecar metadata. These are passed to this system's orchestrator, which validates the content's provenance chain using a configured trust store (either user-defined or default). If external references are found (e.g., pointing to an IPFS location), the orchestrator tries to retrieve the relevant metadata and sidecar to ensure traceability. Even editing apps are considered consumers when they load and save existing C2PA metadata for further processing, contributing

to the continuity of the trust chain. This metadata remains purely informational and it does not restrict future use of the digital content.

### *B. Producers*

Producers are responsible for creating, modifying and publishing C2PA-compliant digital content. In practice, users (e.g., photographers or editors) rely on specialized software – also known as generators – to handle the complex structure of C2PA metadata. In the current system, the desktop editing apps and the generator module of the web app act as content producers. The editing applications integrate a standardized module that interacts with the orchestrator to extract and manage metadata during an editing session. These tools automatically track changes to handle imported C2PA-tagged assets (ingredients), integrating them into a new provenance chain. At the end of an editing session, all metadata (original and newly generated) is bundled into a new C2PA manifest, which is digitally signed using a predefined signing entity. This process is transparent to the user but can be disabled if provenance tracking is not desired.

### *C. Overall System Architecture*

Fig. 2 illustrates the overall system architecture. The following subsections provide a detailed overview of the system's components and details about their implementation.

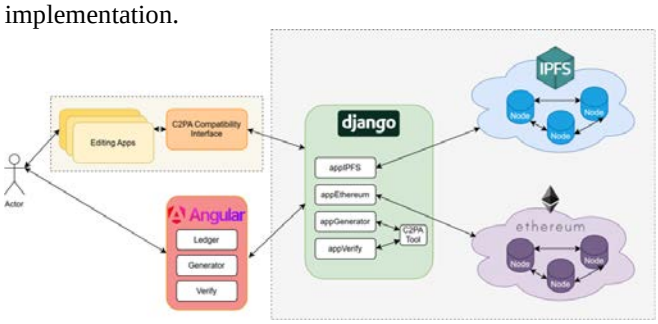


Figure 2. Overall system architecture

### *D. Editing Apps and C2PA Compatibility Interface*

This component refers to the wide range of software products aimed at editing digital content – be it photo, video, or audio – that integrate a compatibility interface for working with metadata in accordance with the C2PA specification.

Existing software products on the market were not designed to handle C2PA metadata. The editing process focuses primarily on the actual content (image, audio, etc.), without preserving information regarding provenance or the history of prior modifications – sometimes even removing existing metadata and replacing it with their own. A notable exception is Adobe, one of the main contributors to the C2PA specification and a CAI member, which has integrated C2PA metadata support into its editing suites (such as Photoshop, Premiere Pro, or Firefly). Digital content creators can thus manage provenance data through a process seamlessly integrated into the software's workflow, without requiring additional user intervention.

However, a wide range of digital editing applications still do not support C2PA metadata as of the writing of this work. A concrete and popular example is Microsoft Paint.

Given the potential limitations and inconsistencies that may arise when attempting to develop applications capable of handling C2PA metadata, the proposed solution is based on integrating a specialized module within editing applications. Therefore, software products that were not initially designed to work with C2PA provenance metadata will be able to integrate high-level methods interfacing with this specialized module in a manner that minimally impacts both the codebase and the user experience. Ideally, users will not even notice the presence of this module, as their usual editing workflows remain unaffected.

The main responsibilities of such a compatibility module include: properly importing digital content along with its associated C2PA metadata, constructing a correct and complete assertion store that reflects the changes made to the content – thus extending the provenance chain in compliance with the C2PA specification – and recording these changes on a blockchain. Since the applications interact with an arbitrary blockchain network, the module must provide specific methods for interacting with such a network, including keystore configuration, transaction signing, and transaction submission. As a result, the application becomes a decentralized application (dApp), meaning that all blockchain interaction details must be handled exclusively on the client side. This requires support for configuring both the blockchain account and the network being used.

The specialized module interfaces with a backend service that performs all these complex operations – operations that developers would otherwise need to implement separately, often inconsistently, leading to fragmented solutions for the same core functionality. The advantage of integrating such a module is clear: it allows developers to focus on building the actual applications while standardizing the integration of provenance metadata.

To demonstrate integration with such a module, three editing applications were developed – one for each type of digital content where C2PA provenance metadata may be manipulated: photo, video, and audio. The decision to develop custom editors was based on the complexity of existing open-source codebases (e.g., GIMP, OpenShot), which would require substantial effort to understand, adapt, resolve dependencies, and repackage. Closed-source applications, on the other hand, cannot be extended technically or legally. Additionally, custom applications simplify performance benchmarking under various runtime conditions, as will be discussed later.

Initially, the editing applications were developed independently under the assumption that such editors already exist on the market. The next goal was to integrate the C2PA compatibility module. Since all applications follow a similar structure, we will exemplify the integration of the specialized C2PA metadata processing module into the custom photo editing application. The same module was also integrated into the video and audio editors, with no changes required to the module source code or the methods it exposes.

At application startup, configuration settings are loaded. The relevant configurable options for this work include: whether to enable the specialized C2PA module, the certificate – private key pair used by the entity signing

provenance changes, the access endpoint to the orchestration component that abstracts decentralized component interactions and C2PA metadata processing or blockchain network details (e.g., RPC endpoint, chain ID).

If provenance tracking is enabled, the application attempts to retrieve the associated C2PA metadata upon importing a photo – either from within the file, from a referenced external source, or from an associated sidecar file. If metadata cannot be recovered from any of these sources, the entire provenance chain will be lost. The user can then proceed with editing the photo.

Finally, the edited photo, along with the data required to update the C2PA provenance chain, is sent – via the specialized module – to the orchestration service, which performs all necessary processing and blockchain transactions. As a result, the output is a new photo that includes provenance metadata conforming to the C2PA specification. Fig. 3 summarizes the entire flow an asset undergoes in this process.

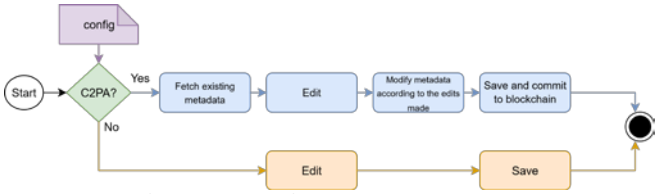


Figure 3. Simplified operations of the editing applications

Both the editing applications and the specialized module were developed using the Python programming language. The module depends on libraries that allow network communication and blockchain interaction. In particular, the use of the requests and web3.py packages is essential for the module's functionality. The editing applications themselves do not require any additional libraries to be installed, as they rely solely on the core logic of the specialized module. All networking and orchestration logic is encapsulated within the module, keeping the client applications clean and decoupled from the complexity of the underlying infrastructure.

### *E. Web App*

To enhance transparency and usability in understanding and experimenting with the C2PA specification, a web-based application was developed. This interface serves as an interactive environment for simulating provenance flows, validating C2PA metadata, and integrating with blockchain registries.

The generator of the web application allows the user to conceptually simulate a content creation workflow. It supports: uploading the primary digital asset and an optional existing C2PA sidecar, loading additional "ingredients" involved in the asset creation process, describing hypothetical editing operations (e.g., cropping, resizing, color adjustments) without altering the file, adding creative context (e.g., author identity, AI generation flags, geolocation) and defining a signing entity via certificate and private key to ensure the authenticity and integrity of the generated manifest. Two options are provided for metadata integration: embedding the manifest directly within the file (increasing size but reducing metadata loss risk) or generating an external sidecar (preserving original file size but introducing external dependency).

The validator recursively verifies the entire provenance chain of a given media asset, whether embedded or externally referenced. Key features include: configurable trust chain or use of a predefined trusted anchor set, optional declaration of unreferenced external manifests, identification and parsing of embedded C2PA metadata and visual indication of provenance presence via a transparency icon, with detailed chain visualization available on demand. This module aligns with the C2PA specification and ensures conformance to the recursive validation process. A summary of the provenance chain validation process can be seen in Fig. 4.

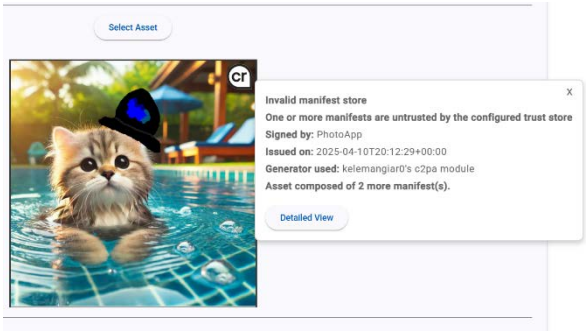


Figure 4. Summary validation of a C2PA asset

The blockchain registry explorer facilitates querying and interacting with the configured blockchain registry, offering: visualization of blocks, transactions, and registered user contracts, multi-network support through configurable RPC endpoints, integration with Web3 wallets (e.g., MetaMask) for authentication and transaction signing and tools for account creation and fund transfers, encouraging exploration within decentralized ecosystems. Together, these features allow validation and access of provenance data across multiple networks, enhancing interoperability and decentralization.

### *F. Orchestrator*

The orchestrator serves as the central component of the system architecture, ensuring the proper management of digital content and C2PA metadata. It integrates the content generation and attestation modules with blockchain technology and distributed storage.

This module is implemented using the Django framework, which is well-suited for a modular architecture. The orchestrator is structured into five distinct applications:

- appMain – the core application responsible for global configuration and orchestration;
- appGenerator – handles the generation of C2PA-compliant digital content;
- appVerify – exposes actions related to content attestation;
- appIPFS – interfaces with the IPFS distributed storage network;
- appEthereum – handles interactions with the blockchain layer.

Each application exposes specific functionality via REST endpoints, which can be accessed by the C2PA-compatible interface or the system's web application, allowing seamless integration with external services. The communication between these applications is illustrated in Fig. 5.

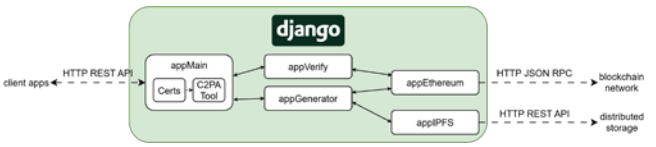

Figure 5. Internal orchestrator architecture

The *appMain* module is critical to the overall project structure. It manages global settings, routing, and coordination of the other applications. It also includes configuration files, default assets used in content processing (e.g., certificates, trust stores, private keys), and the core binary used for C2PA metadata generation and attestation. This binary is developed by the CAI and uses the Rust-based SDK for C2PA metadata processing [13]. Additionally, static files related to the web interface can also be served from this module. The *appGenerator* module provides routes for generating various types of C2PA metadata: embedded, referenced, and external (sidecar). It also handles the submission of transactions that include provenance metadata and returns the resulting C2PA-compliant content. The *appVerify* module provides endpoints for retrieving and verifying C2PA metadata from files. The *appIPFS* module enables file storage and retrieval on the IPFS network. The *appEthereum* module simplifies developer interaction with the Ethereum network.

A key feature of the orchestrator is its flexibility in supporting multiple blockchain networks. Clients can configure preferred networks without being limited to a single implementation.

### *G. Distributed Storage Network*

The distributed storage component is responsible for storing the digital content generated through a C2PA-compatible generator, ensuring that the same content remains publicly accessible. Alongside the blockchain module, this service demonstrates its utility by implementing a fully decentralized model for the storage and verification of files containing C2PA metadata, thereby ensuring non-repudiation of the provenance chain.

These benefits are available as long as the metadata generation services remain formally compatible with the orchestrator of this software product or with other services pursuing the same goal: a fully transparent and decentralized environment for managing provenance data.

The data storage mechanism relies on the IPFS network, one of the most widely adopted solutions for distributed content storage. Due to its decentralized architecture, each stored file is identified by a unique cryptographic hash known as a CID (Content Identifier). Thus, IPFS complements any blockchain system by providing efficient and minimal data persistence and verifiable referencing capabilities through these immutable identifiers.

This component is implemented using three IPFS nodes, and the absence of a centralized data node makes malicious manipulation of stored content virtually impossible – unlike in centralized architectures. Each participating node independently stores and serves content from the IPFS network. The associated IPFS network can be extended by configuring additional nodes to match the original network parameters.

Digital content generated by the system is stored on IPFS through an HTTP-based process mediated by the orchestrator. The resulting CID is then used in the deployment of a corresponding smart contract. This integration ensures persistence and verifiability of the references to stored content, significantly reducing the risks of file loss or malicious tampering.

Within the overall system, this module provides the persistent and distributed storage mechanism responsible for ensuring the availability, durability, and non-repudiation of digital content, in conjunction with the blockchain component described in the following section.

### *H. Blockchain Network*

The blockchain component plays a central role in the architecture of the entire system, as it underpins the decentralized nature of the service. All data and interactions performed through the service are recorded in a ledger, with each network node maintaining its own complete copy. The distributed nature of this ledger ensures transparency and immutability, making any attempt to alter the stored data exceedingly difficult. Consequently, falsifying provenance chains that the service aims to protect becomes virtually impossible, as it would require modifying all copies of the blockchain ledger across the network.

The service is designed around two well-defined types of smart contracts: the genesis contract and user contracts. The genesis contract is unique per blockchain network and acts as a registry that links each account to a corresponding user contract, if such a contract exists. This structure allows the genesis contract to behave similarly to a database, enabling queries that return the address of a user's smart contract.

The user contracts function like personalized databases for each entity, storing information about the digital content created through the system. This data includes: the file name, the associated CID, whether the content has an external provenance chain (linked or sidecar), and the CID of that external provenance record.

Before the service becomes operational, the genesis contract must be deployed and its address recorded. This address is treated as a "well-known address" and is publicly accessible. Consequently, client applications, when configuring a connection to the blockchain, must specify both the RPC endpoint and the genesis contract address. This approach ensures that users (represented by blockchain accounts) can interact with their associated contracts. A similar design can be observed in the Ethereum mainnet, where tokens like USDT (Tether) have well-known contract addresses that allow any application to interact with them.

Once the genesis contract is deployed and the client apps are correctly configured, the service is ready to be used for authenticity and provenance protection. Each new entity, represented by an account, must register in the genesis contract the address of its freshly created user contract before storing any digital content metadata (such as file name or CID). After this registration, content created by that entity is stored in their corresponding user contract. Subsequent content entries are appended to the same user contract, significantly reducing registration costs and enabling aggregation of data related to a single content creator. This results in the formation of structured, well-defined "databases" of digital assets per user – similar to the

structure seen in large-scale token contracts.

To perform transactions, users must possess a sufficient balance of cryptocurrency to cover gas fees. The blockchain ensures authenticity and traceability of digital assets through service-specific assertions embedded in each content file. These assertions point to the blockchain network and the precise location (i.e., smart contract) where the content's provenance information is stored, enabling public verification of content authorship and modification history.

Each smart contract is associated with a single account (its deployer), making it possible to determine the original author of a given digital asset by querying the blockchain. If a piece of content is later modified by another party, the new provenance chain is registered in the user contract of the modifying entity. This creates a transparent and traceable provenance history for each version of the content.

For easier tracking of how smart contracts are used within this service, Fig. 6 illustrates the overall workflow.

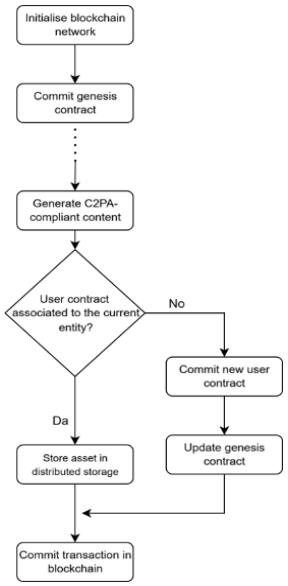


Figure 6. Smart contract interrogation flow during content creation workflows

Finally, by including both the user contract address and the blockchain RPC endpoint in the service's assertion metadata, interoperability is ensured – even when the content has passed through multiple instances of the service configured differently. As long as the C2PA metadata schema and user contract structure remain consistent across networks, verification and provenance tracking remain straightforward and coherent, regardless of where the content is stored.

## IV. Results

### A. *Performance and Stability*

To support the development, testing and maintenance of the proposed system, automated tests were implemented as a core component of the system validation process. These tests are essential to reliably and systematically verify critical application workflows. To demonstrate that some consumer and producer related workflows are correctly executed and to evaluate system stability and performance across various conditions, a suite of specialized automated tests was developed using the *pytest* framework. These tests were designed to simulate realistic use behavior and component interactions, significantly improving the early detection of issues that may arise. The test suite is publicly available in the GitHub repository mentioned in the introduction and can easily be executed in a properly configured environment.

The following subsections present key testing results and performance measurements that reflect the system's capabilities and goals. The automated tests mentioned previously were used on both private and public blockchain networks, enabling evaluation across diverse consensus mechanisms and interoperability scenarios. In total, five private blockchain networks were deployed and tested, as follows: Ethereum Proof of Work, Ethereum Proof of Stake, Ethereum Proof of Authority, Avalanche Proof of Stake and Avalanche Proof of Authority. Additionally, two public testnets were used: Ethereum Holesky and Avalanche DeFi Kingdoms Testnet, both using a Proof of Stake consensus. Due to limitations related to transaction throughput and token availability on public testnets, these tests were run at a smaller scale. Nevertheless, they proved valuable for validating interoperability and end-to-end integration. All supported file types and networks were covered during testing, ensuring a broad and representative evaluation of the system's behavior in real-world usage scenarios.

1. File generation costs (measured in ETH)

The cost analysis across different blockchain networks reveals that transaction fees remain constant for a given network, regardless of file type or size, as in Table I. This is expected, as the smart contract writes a consistent set of metadata – namely the filename IPFS CID and, if applicable, the provenance chain CID – regardless of the file's actual size.

Table I. Average Generation Cost by Blockchain Network

| Network / Size | Eth. PoW | Eth. PoS | Eth. PoA | Ava. PoS | Ava. PoA | Holesky | DeFi Kingdoms |
|---|---|---|---|---|---|---|---|
| 100kb | 0.0994 6282 | 0.0895 6305 | 0.1020 2219 | 0.0051 9315 | 0.0051 9315 | 0.0000 0021 | 0.0000 0830 |
| 200kb | 0.0997 0823 | 0.0889 8037 | 0.1020 2219 | 0.0051 9315 | 0.0051 9315 | 0.0000 0021 | 0.0000 0830 |
| 500kb | 0.0997 3595 | 0.0889 6715 | 0.1020 2550 | 0.0051 9315 | 0.0051 9315 | 0.0000 0021 | 0.0000 0830 |
| 1mb | 0.0985 8542 | 0.0875 4286 | 0.1019 4708 | 0.0051 9243 | 0.0051 9243 | 0.0000 0021 | 0.0000 0830 |
| 2mb | 0.0986 0669 | 0.0852 1677 | 0.1019 4708 | 0.0051 9243 | 0.0051 9243 | - | - |
| 5mb | 0.0994 2882 | 0.0877 4718 | 0.1019 4708 | 0.0051 9243 | 0.0051 9243 | 0.0000 0022 | 0.0000 0830 |
| 10mb | 0.1001 0804 | 0.0880 2090 | 0.1019 8379 | 0.0051 9279 | 0.0051 9279 | 0.0000 0021 | 0.0000 0830 |
| 20mb | 0.0997 1150 | 0.0882 2324 | 0.1019 8379 | 0.0051 9279 | 0.0051 9280 | 0.0000 0022 | 0.0000 0830 |
| 30mb | 0.0999 4089 | 0.0883 2647 | 0.1019 8721 | 0.0051 9279 | 0.0051 9279 | 0.0000 0021 | 0.0000 0830 |

2. File generation times (measured in seconds)

As observed in Table II, generation time increases with file size.

Table II. Average Generation Time by Blockchain Network

| Network / Size | Eth. PoW | Eth. PoS | Eth. PoA | Ava. PoS | Ava. PoA | Holesky | DeFi Kingdoms |
|---|---|---|---|---|---|---|---|
| 100kb | 15.256 | 11.946 | 14.982 | 1.926 | 1.954 | 19.012 | 2.805 |
| 200kb | 14.401 | 12.080 | 14.992 | 1.981 | 1.981 | 20.213 | 2.728 |
| 500kb | 15.209 | 11.923 | 14.999 | 1.962 | 1.935 | 15.185 | 2.86 |
| 1mb | 15.587 | 11.974 | 15.027 | 1.997 | 1.963 | 17.917 | 2.802 |
| 2mb | 15.507 | 12.028 | 15.015 | 2.041 | 1.956 | - | - |
| 5mb | 17.838 | 12.096 | 15.085 | 2.113 | 2.089 | 19.073 | 3.232 |
| 10mb | 17.416 | 12.167 | 15.225 | 2.175 | 2.168 | 25.187 | 3.557 |
| 20mb | 17.058 | 12.356 | 15.526 | 2.432 | 2.512 | 25.196 | 4.725 |
| 30mb | 18.086 | 12.569 | 15.731 | 3.101 | 3.091 | 18.863 | 5.697 |

This is primarily due to the time required to upload the file to the IPFS network in order to retrieve the content identifier

 Digital Object Identifier 10.32754/JMT.2026.1.05

before the blockchain transaction can proceed. Thus, the time is directly proportional to the file's size and not significantly impacted by the blockchain network type.

3. Provenance attestation costs (measured in ETH)

The attestation process – verifying the provenance of a file – incurs zero cost across all tested networks. This is a natural outcome, as attestation involves only reading data from the blockchain, which is a public and non-intrusive operation. Costs are associated only with writing to the blockchain. Therefore, the choice of blockchain network has no cost-related impact on attestation.

4. Provenance attestation times (measured in seconds)

Attestation time increases modestly with file size, primarily due to longer IPFS access and processing durations for larger files. However, even for the largest files tested, attestation remained performant and the differences between blockchain networks were minimal, as seen in Table III.

TABLE III. AVERAGE ATTESTATION TIME BY BLOCKCHAIN NETWORK

| Network / Size | Eth. PoW | Eth. PoS | Eth. PoA | Ava. PoS | Ava. PoA | Holesky | DeFi Kingdoms |
|---|---|---|---|---|---|---|---|
| 100kb | 0.027 | 0.021 | 0.023 | 0.019 | 0.036 | - | - |
| 200kb | 0.027 | 0.025 | 0.024 | 0.023 | 0.024 | - | - |
| 500kb | 0.038 | 0.034 | 0.037 | 0.034 | 0.033 | - | - |
| 1mb | 0.047 | 0.039 | 0.043 | 0.039 | 0.037 | - | - |
| 2mb | 0.04 | 0.037 | 0.037 | 0.036 | 0.035 | - | - |
| 5mb | 0.100 | 0.098 | 0.093 | 0.089 | 0.088 | - | - |
| 10mb | 0.129 | 0.125 | 0.128 | 0.123 | 0.122 | - | - |
| 20mb | 0.243 | 0.239 | 0.239 | 0.230 | 0.230 | - | - |
| 30mb | 0.349 | 0.358 | 0.355 | 0.350 | 0.344 | - | - |

Based on the above experimental results, the following statements can be made:

- Ethereum Holesky is the most cost-efficient for write operations.
- The private Avalanche PoS provides the best performance in terms of processing speed.
- Public blockchain networks offer better cost-per-transaction ratios but come with limitations such as transaction delays and resource constraints.
- Attestation processes are free and consistent across all networks.

These results confirm the system's robustness, interoperability and adaptability across multiple blockchain environments and provide a solid base for future enhancements.

### *B. Security*

The cryptographically guaranteed security primitives are:

- **Integrity**: the use of hash algorithms for both C2PA specification purposes and IPFS CID generation ensures the detection of unauthorized modifications to files and metadata. Any alteration to the content results in a different hash value, thereby invalidating integrity verification.
- **Authenticity**: digital signatures based on standard cryptographic schemes applied to C2PA provenance metadata and blockchain transactions certify the identity of the involved entities, confirming the origin of the data and transactions.
- **Non-repudiation**: since blockchain transactions are signed using users' private keys, changes in the distributed ledger cannot be later denied, ensuring clear accountability of actors within the system.
- **Confidentiality (of private keys)**: the private keys used for blockchain and C2PA metadata signatures are stored and used at the client level, eliminating the risk of exposing them over the network.
- **Transparency and immutability**: the blockchain provides a public, immutable and auditable record of all transactions, while IPFS ensures distributed access to the content of any file through its content identifier.

The integration of C2PA metadata with decentralized technologies such as blockchain networks and distributed storage systems is not incidental; rather, it creates a synergistic and robust framework for protecting media content authenticity. Publicly and transparently, the combination of these technologies provides strong guarantees of integrity, traceability and non-repudiation, surpassing the limitations of any one of them used in isolation. C2PA delivers a standardized and extensible model for declaring provenance and edits, blockchain adds immutability and accountability and distributed storage ensures scalable accessibility and durability of content.

The service implements a dual authentication mechanism that combines the cryptographic characteristics of C2PA metadata with those of blockchain accounts. First, C2PA metadata is digitally signed using X.509 certificates in accordance with the C2PA specification [3], which allows for signed identity verification through a standardized trust chain. The digital signature ensures the authenticity and integrity of the C2PA metadata, preventing unauthorized changes and ensuring source accountability. Second, the transactions that update user contracts on the blockchain – which represent the creation of content with provenance metadata through this service – are signed exclusively on the client side using blockchain private keys stored in locally protected keystores. This guarantees that only the account holder can authorize and register transactions that signify content alteration, offering a transparent audit trail. Separating these authentication methods reduces the risk of a complete compromise of a single entity.

Media files are uploaded to the IPFS distributed network, which uses content identifiers (CIDs) represented by cryptographic hashes of the file content. These unique identifiers ensure the immutability and uniqueness of file references, as any change to the file results in a different identifier. This cryptographic property provides a technical guarantee of content integrity: when a CID is stored on the blockchain and associated with a user contract, the IPFS file reference becomes public and immutable. Anyone can then verify that the file matches exactly the recorded version. Thus, the combination of blockchain and IPFS identifiers eliminates the need to store files directly on-chain, avoiding high costs while preserving a secure and scalable content verification mechanism.

Client-side signing of blockchain transactions is a critical architectural decision for securing such a service. Exclusive control of private keys by the client eliminates the risk that the orchestrator or any other service component could modify transactions or perform unauthorized actions on behalf of the user. Any invalid or improperly signed transaction is automatically rejected by the blockchain network, ensuring protection against fraud.

To demonstrate the robustness of the current implementation against common attacks, security testing was conducted using c2pa-attacks [14] and OWASP ZAP. The test suite defined by the c2pa-attacks repository simulates attacks specific to C2PA manifests. It generates signed files containing special characters, unusual strings or potentially malicious content. Each generated file was uploaded and processed by this system's attestation interface to test its rendering and resilience. Even when injected data had unusual formats, the attestation interface of the web application experienced no crashes, freezes or interruptions, successfully displaying the corresponding content. Some of the results are shown in Fig. 7. This confirms that the application robustly handles unconventional data from the C2PA manifests without negative impacts on functionality.

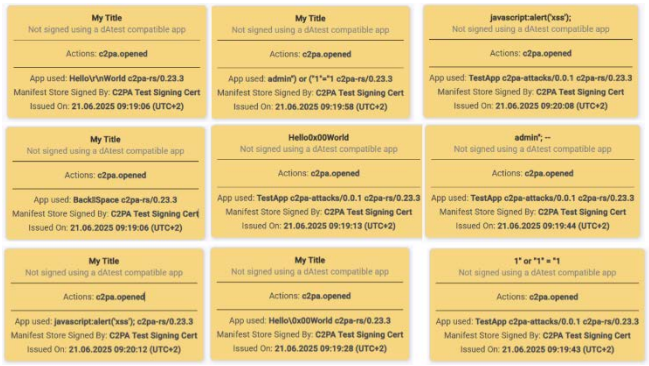


Figure 7. C2PA metadata attestation involving potentially malicious strings

To assess the web application's security, the OWASP ZAP suite was used. Tests were conducted in a development environment where the application runs with temporary servers (Angular and Django). Most identified alerts were related to missing HTTP headers (such as CSP, X-Content-Type-Options or anti-CSRF), which are typical of development settings. These do not represent critical vulnerabilities and can easily be addressed in production by configuring dedicated proxies. The importance of this test lies in the absence of real vulnerabilities such as cross-site scripting, SQL injection or path traversal, which confirms the robustness of the implementation. The complete report is available in the GitHub repository mentioned in the introduction.

## V. Conclusion

This paper explored the integration of the C2PA specification with decentralized technologies to enhance the authenticity and provenance of digital content. Several challenges were encountered, particularly due to the emerging and rapidly evolving nature of the underlying technologies. One major issue was the integration of C2PA metadata signing in compliance with the latest specification versions, as the available SDK is still under active development, resulting in frequent compatibility problems. Furthermore, the centralized signing model used in the orchestrator raises security concerns related to the transmission of the private keys of C2PA entities over the network.

Despite these limitations, the proposed system demonstrates a functional proof of concept for integrating C2PA metadata with blockchain-based storage and verification mechanisms. The architecture of the current system was designed to be modular and extensible, allowing for future reconfiguration and improvement.

To address current limitations of this system, several extensions are proposed by the authors, such as:

- a long-term preservation module to maintain signature validity over time;
- a similarity scoring module to detect derived or manipulated content that would complement the existing soft binding mechanism;
- a compatibility management service to enable secure local signing of C2PA provenance data, ensuring interoperability with other implementations;
- a decentralized trust mechanism for signer validation through community consensus.

The developed system showcases how decentralized services can be integrated into existing applications to support secure and transparent provenance tracking. Multiple prototype applications – both desktop and web – were created to simulate integration of C2PA metadata into editing workflows, all coordinated through a unified orchestration layer that manages blockchain and distributed storage networks.

To the best of the author's knowledge, the only partially comparable project is Numbers Protocol, which focuses on ownership transfer of digital assets. In contrast, this work emphasizes trust in the provenance and modification history of the content, enabling decentralized and verifiable authenticity.

## References

[1] L. Goodwin, "BBC News," BBC, 20 December 2024. Available at: https://www.bbc.com/news/articles/cdr0g1em52go.
[2] J. Wakefield, "BBC News," BBC, 18 March 2022. Available at: https://www.bbc.com/news/technology-60780142.
[3] Coalition for Content Provenance and Authenticity, "Content Credentials: C2PA Technical Specification." Available at: https://c2pa.org/specifications/specifications/2.2/specs/C2PA_Specification.html.
[4] E. Bureacă and I. Aciobăniței, "A blockchain based framework for content provenance and authenticity," in *2024 16th International Conference on Electronics, Computers and Artificial Intelligence (ECAI)*, 2024, pp. 1–5. doi:10.1109/ECAI61503.2024.10607477
[5] Microsoft, "Learn: Microsoft," 19 March 2025. Available at: https://learn.microsoft.com/en-us/azure/ai-services/openai/concepts/content-credentials.
[6] Adobe, "Help: Adobe," 18 June 2025. Available at: https://helpx.adobe.com/creative-cloud/help/content-credentials.html.
[7] W. Allen, "Blog: Cloudflare," 3 February 2025. Available at: https://blog.cloudflare.com/preserve-content-credentials-with-cloudflare-images/.
[8] Content Credentials, "Verify: Content Credentials." Available at: https://contentcredentials.org/verify.
[9] Adobe, "Inspect: Adobe Content Authenticity." Available at: https://contentauthenticity.adobe.com/inspect.
[10] Numbers, "Home: Numbers Protocol." Available at: https://www.numbersprotocol.io/.
[11] Capture, "Home: Capture App." Available at: https://captureapp.xyz/.
[12] Numbers, "Home: Numbers Archive." Available at: https://archive.numbersprotocol.io/.
[13] Content Authenticity Initiative, "c2pa-rs." Available at: https://github.com/contentauth/c2pa-rs.
[14] Content Authenticity Initiative, "Github: c2pa-attacks." Available at: https://github.com/contentauth/c2pa-attacks.