Watermarking in Media Content

Tristan Kobusch

Abstract—The rise of AI-generated and manipulated media has created a pressing need for reliable content authenticity signals. Cryptographically signed metadata, as enabled by the C2PA standard [1], can record an asset's provenance, but such metadata is often stripped away when media is shared, undermining its effectiveness. In this paper, I explore a watermarking-enhanced content credential approach to preserve media integrity even when metadata is lost. I present an implementation that integrates robust invisible watermarks into video content (using the recent Video Seal technique [2]) and signs the media's provenance metadata via the Coalition for Content Provenance and Authenticity (C2PA) standard. My architecture combines a Rust backend with Python-based neural processing and C2PA manifest generation. I evaluate the system's performance, showing that the watermark is imperceptible and survives common transformations and metadata stripping, allowing content verification through watermark extraction and metadata lookup. I discuss the robustness and limitations of this approach, and how it offers a durable solution for tracing media origins and detecting disinformation. My findings suggest that combining C2PA secure metadata with resilient watermarks can substantially improve content authenticity in today's complex media ecosystem.

I. Introduction

Modern digital media faces a crisis of trust amid rampant misinformation and manipulated content. Advances in image/video editing and generative AI have enabled the creation of realistic deepfakes and mis-contextualized media, eroding confidence in the authenticity of what we see online. [3] High-profile incidents highlight this problem – for example, an apparent video of a public figure may circulate widely before being debunked as fake, by which time it has already influenced public opinion. This propagation of disinformation underscores the need for techniques to prove the integrity and origin of media.

One emerging solution is the use of secure provenance metadata. The Content Authenticity Initiative (CAI) [3] and the Coalition for Content Provenance and Authenticity (C2PA) have defined an open standard for attaching cryptographically signed metadata (called content credentials) to media assets. [1]

These content credentials act as tamper-evident digital signatures that record who created or edited the content and how. In principle, consumers or platforms can verify this metadata to distinguish original content from manipulated versions. However, a fundamental challenge is that metadata can be lost or removed. Many social networks and messaging apps strip metadata from images/videos on upload, either intentionally or as a side-effect of compression and reformatting. [4]

These content credentials act as tamper-evident "digital signatures" that record who created or edited the content and how. In principle, consumers or platforms can verify this metadata to distinguish original content from manipulated versions. However, a fundamental challenge is that metadata can be lost or removed. Many social networks and messaging apps strip metadata from images/videos on upload, either intentionally or as a side-effect of compression and reformatting. Malicious actors can also deliberately remove or alter metadata to conceal a content's origin. When the provenance metadata is missing, the benefits of C2PA's cryptographic assurances vanish.[5]

To address this I explored the possiblity of combining C2PA's signed metadata to create durable content credentials as outlined in [4]. Watermarking embeds information into the pixels/audio of the media itself, in a manner that is imperceptible to human viewers but can be detected by software. The idea is that a watermark can carry a hidden ID or code linking back to the content's metadata. If the manifest is stripped, the ID in the watermark can be used to look up the original credentials in a database or ledger. This approach, referred to as a durable content credential, leverages the strengths of both metadata and watermarking – the watermark persists through transformations, and the metadata provides cryptographic verification and rich context.[4]

II. Related Work

Digital watermarking has evolved significantly since the 1990s, progressing from simple spatial

The source code of the project is available at https://github.com/kobutri/awt

domain techniques to more advanced transform domain methods like DFT, DCT, and DWT. Traditional methods often struggle to balance the core requirements of watermarking: imperceptibility, robustness, and capacity. These trade-offs have led to the exploration of deep learning-based techniques, which offer solutions to these challenges by learning complex features directly from data.

Deep learning architectures such as Convolutional Neural Networks (CNNs), Generative Adversarial Networks (GANs), Recurrent Neural Networks (RNNs), and autoencoders are now widely used in video watermarking. CNNs excel at identifying spatial patterns, enabling more effective embedding strategies. GANs use adversarial training to improve both watermark robustness and invisibility by leveraging a generator and discriminator network. RNNs, particularly with LSTM or GRU layers, capture temporal dependencies across video frames, ensuring watermark integrity over time. Autoencoders compress video data into latent spaces, embedding watermarks that remain resilient to processing attacks.

In addition to these neural network approaches, the field has also seen advancements in attack simulation and optimization techniques. Simulated attacks—such as compression, noise addition, and geometric transformations—are used to evaluate the robustness of watermarking systems, while data augmentation enhances model training by exposing it to various distortions. These innovations make deep learning-based watermarking methods more adaptive and robust, representing a significant advancement over traditional techniques in securing digital video content.[6]
VideoSeal by Meta introduces a novel framework for neural video watermarking by integrating temporal watermark propagation, eliminating the need to embed watermarks in every frame. The approach combines image pre-training with hybrid post-training and extractor fine-tuning to achieve efficient, robust watermarking. It incorporates differentiable augmentations including video codec simulations to improve resistance against geometric and compression distortions. The method leverages an embedder/extractor architecture with optimized U-Net and vision transformer designs. Open-sourced code make it an ideal choice for this project.[2]

III. Approach

System Overview: My approach combines C2PA-based secure metadata with a Video Seal invisible watermark to create media that is both self-identifying and verifiable. The system pipeline comprises three stages: (1) watermark embedding, (2) manifest generation and signing, and (3) verification via watermark extraction and metadata lookup. I leverage a Rust backend for C2PA signing and watermark lookup and Python for generating and extracting the watermarks via Video Seal.

A. Watermark Embedding with Video Seal

I use the Video Seal neural watermarking algorithm to embed a hidden payload into video content. Video Seal provides an API for embedding a message of a specified bit-length by slightly perturbing pixel values, ensuring that the watermark remains visually imperceptible. In my prototype, I configured Video Seal to embed a 96-bit payload (the watermark ID) into the video, with the ID generated as a random UUID to guarantee uniqueness. The embedding process leverages temporal watermark propagation: only the first frame of each segment (e.g., one segment per second) is fully watermarked, and the watermark signal is propagated to subsequent frames. Significant effort has been spend on making

B. C2PA Manifest Creation and Signing

After the watermark embedding is complete, I generate a C2PA manifest that contains the watermark. Once assembled, the Rust backend signs the manifest using an X.509 digital certificate, generating a signature over the manifest (which includes the media's hash and assertions). The signed manifest is then embedded within the media file, in this case, an mp4 file.

C. Media Output and Distribution

The final output is a video file containing both an invisible watermark (embedded at the pixel level) and a signed C2PA manifest (embedded as metadata). This file can be distributed through conventional channels. On platforms that support C2PA, e.g. [7] or the frontend included in this project, users can verify content authenticity by

checking the manifest's signature and associated provenance information. Even if the platform strips metadata, the invisible watermark remains, enabling later retrieval of the provenance data.

D. Watermark Extraction and Verification

To ensure that content can be authenticated even when metadata is lost, I implemented a verification stage that extracts the watermark. Aside from displaying the contents of attached C2PA metadata, the frontend can also upload the video to the Rust frontend, which will forward the file to the python backend that uses the Video Seal decoder to recover the 96-bit watermark. Rather than a bitstring the decoder will generate a 96-bit float vector of values from $[0, 1]$. The average of these values over all frames gets sent back to the rust backend, which will find the stored video with the most similar watermark, calculated as cumulative distance over all bits. The frontend will then download that video and apply the same C2PA metadata extraction process. The user can then compare the actual video and manifest data.

IV. Evaluation

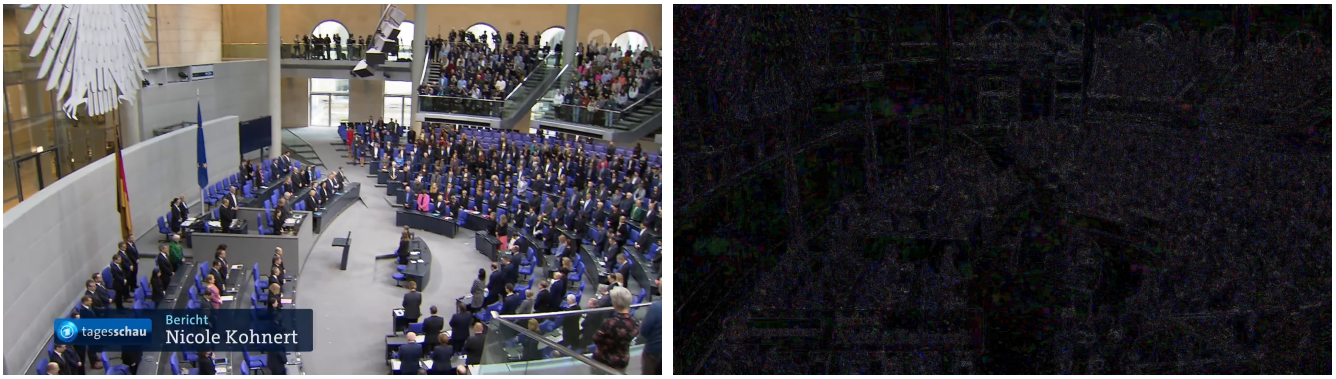
I evaluated my combined C2PA and watermarking solution on a set of test videos including a news broadcast [8] as well as an animated short film [9] to assess performance overhead, watermark imperceptibility, robustness against common distortions, and the ability to recover provenance after metadata stripping.

A. Performance

A key concern was whether the additional neural watermarking and signing processes would be practical in real-world workflows. My experiments showed that embedding the watermark into a 1080p, 30fps video takes on average 6.75 ms per frame or around 150fps. Of that, around 1ms is spent decoding the frame and 1.75ms encoding the frame afterwards, 2.5 ms for processing the image, including normalizing the tensor, running the model to calculate the delta values, blending it into the other images of the chunk and denormalizing the tensor. The remaining 1.5ms is spent on overhead like uploading the video from the browser to Rust to Python and back as well as attaching the C2PA metadata. Particularly, the upload and download speed between the browser and the Rust backend will of course be much lower in a real deployment, where the data has to go over the network.
It is worth noting that running the model comes with significant hardware requirements. In order reduce VRAM utilization I tried performing the decoding, encoding and blending on the CPU and only run the model inferencing on the GPU. While this does reduce the amount of VRAM used from about 10GB to around 500MB, it comes with a significant speed decrease. Only running at around 10fps, which means it's not able to run in real time.

B. Imperceptibility

I can confirm the findings of [2] that in typical viewing scenarios without side-by-side scenarios watermarking artifacts are rarely visible. Figure1 shows a frame from the original video besides the diff applied by the watermarking algorithm. The diff has been enhanced for better visibility. It is much smaller in reality. It is apparent that the delta somewhat resembles the original video. While it's not possible to deduce the precise mechanism it is noteworthy, that the frequency of the pattern roughly matches the level of detail in the original, i.e. more detailed areas have a higher frequency and more uniform areas in the original have lower frequency delta applied to them. There are however some areas where the watermark is more visible. Figure 2 shows the comparison between an unwatermarked and its corresponding watermarked image. The dark parts of the jacket exhibit a noticeable color aberration. This phenomenon can be found frequently in large, low detail dark areas. While not generally distracting it might be noticeable to some viewers. Given that the temporal sampling and blending only calculates a new delta every 4 frames, I was able to find surprisingly few instances of motion artifacts, even in case of rapid movement. One outlier is depicted in Figure 3. Here are flash cases a massive change in color from one frame to the next causing some artifacts seen in the red rectangle. To rule out compression artifacts I have verified that these artifacts are not present in the original video.



(a) original (b) enhanced delta

Fig. 1: Comparing the unprocessed video with the watermarking delta. The magnitude of the delta has been increased for better visibility. Video by[8]

(a) original (b) watermarked

Fig. 2: Comparing unprocessed and watermarked image. Video by[8]

C. Robustness

I was able to verify that the watermark is able to withstand, reencoding, slight cropping and the analog bridge. It should however be noted, that in the latter cases rarely all 96 bits can be correctly recovered, typically hovering more around 90-95% bit accuracy. Given my relatively small test library that was sufficient to correctly identify the original video. It should however be noted that larger libraries collisions are possible and might be mitigated by not just showing the user the closest match, but have them choose the correct one from a ranked list.

V. Discussion

While the project demonstrates the feasibility and usefulness of the concept of using watermarking for metadata recovery, additional work is required to turn it into a production-ready solution. There also remain a number of conceptual hurdles that complicate real-world adoption. The watermarking algorithm used here is open source and thus any embedded watermark can be easily spoofed. This project addresses that

(a) first image (b) second image

Fig. 3: Comparing two successive frames, with drastic changes in color. Video by[8]

concern by giving the user the option to compare the videos. An improved implementation might automate this using robust perceptual comparison algorithms like VMAF [10] or highlight differences in a way more obvious to the user. The fundamental issue with this approach is scalability. It requires that services, that store the database and watermarks also store the original videos. While that might be logistically feasible for large content providers and platforms, it is difficult to separate it out into its own service. Instead of storing the original video itself, a service could just store links to the original, but would then be subject to link rot. Hosters of these videos, might want to re-encode them at some point to save space, change their URL schema, might be compelled by users or legal actions to delete videos or could simply go out of business. Any such watermarking metadata service would clearly degrade over time. The solution typically proposed [4] is to instead of videos store perceptual hashes. The problem of perceptual hashes is that by their very nature similar things will look similar. Often times, subtle manipulation of videos can already cause a great amount of harm. Without the original video it becomes impossible to determine just from the video hash whether a video was actually manipulated and in what way or whether the metadata just got accidentally lost. At the core of the trust model behind C2PA is the cryptographic integrity of the metadata. When relying on solutions like fingerprinting or watermarking such guarantees are impossible to provide. That adds another layer to the already difficult question of how to communicate the information and trust provided by C2PA to the user.

Aside from its interaction with watermarking, there are also still many question marks with regards to C2PA itself. One of the goals of C2PA was not just to make provenance of media files verifiable, but to make their creation history transparent. To that end, producers of image editing software[11] and AI image generators[12] have begun to include C2PA into their offerings. While that works well for server-generated content, media created on the user's device would either require to be uploaded to the server, raising concerns about privacy or include the signing keys themselves making them vulnerable to extraction. Many publishers might also prefer not to publish their editing history. While this editing history can easily be replaced by less transparent provenance information having it in the first places carry significant risk that it might get accidentally leaked.

VI. Conclusion

I presented a approach to reinforcing media content authenticity by fusing C2PA content credentials with invisible watermarking. This method addresses the critical problem of metadata loss—a common vulnerability in provenance tracking—by embedding a persistent identifier directly into the media. My implementation using a Rust backend (with Python modules for watermark embedding and extraction) and the Video Seal watermarking model demonstrates that it is feasible to create durable watermarks for video that survive common transformations and enable post-hoc verification of origin.

References

[1] [Online]. Available: https://c2pa.org/

[2] P. Fernandez, H. Elsahar, I. Z. Yalniz, and A. Mourachko, "Video seal: Open and efficient video watermarking," arXiv preprint arXiv:2412.09492, 2024.

[3] A. Parsons, "Durable content credentials," Apr 2024. [Online]. Available: https://contentauthenticity.org/blog/durable-content-credentials

[4] ——, "Durable Content Credentials — contentauthenticity.org," https://contentauthenticity.org/blog/durable-content-credentials, 2024, [Accessed 03-03-2025].

[5] T. Bui, S. Agarwal, and J. Collomosse, "Trustmark: Universal watermarking for arbitrary resolution images," 2023. [Online]. Available: https://arxiv.org/abs/2311.18297

[6] S. Mansour, S. Ben Jabra, and E. Zagrouba, "A comprehensive overview of deep learning based video watermarking: current works, challenges and future trends," Multimed. Tools Appl., Sep. 2024.

[7] "Content Credentials — contentcredentials.org," https://contentcredentials.org/verify, [Accessed 03-03-2025].

[8] ARD, "Tagesschau 20 00 Uhr, 29.01.2025 : ARD : Free Download, Borrow, and Streaming : Internet Archive — archive.org," https://archive.org/details/tagesschau-20-00-uhr-29.01.2025, 2025, [Accessed 03-03-2025].

[9] B. Foundation, "Big Buck Bunny — peach.blender.org," https://peach.blender.org/, 2008, [Accessed 03-03-2025].

[10] P. Topiwala, W. Dai, J. Pian, K. Biondi, and A. Krovvidi, "Vmaf and variants: Towards a unified vqa," 2021. [Online]. Available: https://arxiv.org/abs/2103.07770

[11] Adobe, 2025. [Online]. Available: https://helpx.adobe.com/photoshop/using/content-credentials.html

[12] OpenAI, "C2pa in dall · e 3," 2024. [Online]. Available: https://help.openai.com/en/articles/8912793-c2pa-in-dall-e-3