Alon Rosen


# Provenance

*The public will have only the final product*[*]

While watermarking provides evidence that an output came from a model, *provenance* provides evidence that it is anchored to an authenticated real-world source. The two are complementary approaches to the question of what we believe about the origin of a digital object.

Provenance depends on broad adoption of authenticated capture. Once a source is authenticated, an editor can establish that a published object is an allowed transformation without knowing the signing key. At the simplest, the editor reveals the signed source and the transformation; succinct proofs can compress this evidence, and generic *zero-knowledge* (ZK) proofs can additionally hide the source and intermediate data. Thus the basic cryptographic problem is solved in principle.

The remaining cryptographic challenge is to make these proofs efficient at scale. A practical system must also protect the object from capture through subsequent processing. It must cope with compromised devices and revoke their credentials, establish trustworthy capture times, preserve privacy rather than create a tracking mechanism, and remain verifiable long after publication.

The main points of this chapter are:

- Watermarking identifies model origin; provenance authenticates real-world origin.
- Signatures authenticate sources; proofs certify permitted derivations without the signing key.
- ZK adds privacy; deployment adds capture security, revocation, time, and longevity.

[*] *Gilliam v ABC, Inc, 1976*

# 1 Authentication

Authentication asks whether a received message originated from an authorized source and has remained unchanged. Digital signatures make this publicly verifiable.

**Definition 1** (Digital signature)**.** *A digital signature scheme is a triple* $(\mathrm{Gen}, \mathrm{Sign}, \mathrm{Verify})$*. For* $(sk, vk) \leftarrow \mathrm{Gen}(1^\lambda)$*, signing maps* $(sk, m)$ *to a signature* $\sigma$*, while public verification maps* $(vk, m, \sigma)$ *to a bit. For every message* $m$*, correctness requires* $\Pr[\mathrm{Verify}_{vk}(m, \mathrm{Sign}_{sk}(m)) = 1] = 1$*.*

Correctness can be achieved by having Verify always accept. But this provides no authentication: every message is automatically accepted as valid. What we can hope for is that, without knowledge of $sk$, an adversary cannot make a new message verify, except one that has already been authenticated.

**Definition 2** (Unforgeability)**.** *A signature scheme is* $(t, \varepsilon)$*-unforgeable if, for every adversary* $A$ *running in time at most* $t$*,*

$$\Pr\big[A^{\mathrm{Sign}_{sk}(\cdot)}(vk) \ \textit{“forges”}\big] < \varepsilon.$$

Here “forges” means that $A$ outputs a valid signature on a message that it did not previously submit to the signing oracle.

If, say, a photograph $m$ is signed, a crop or recompression $m'$ is a different message: producing a valid signature on $m'$ without asking the signer would violate unforgeability, even when the edit is entirely legitimate. Ordinary signatures therefore authenticate exact messages; they do not say which transformations should preserve authenticity. The main cryptographic challenge for provenance is to preserve evidence of authentication under permitted transformations without giving editors the signing key.

A special case is *message authentication* (MAC), where $vk = sk$: only holders of the secret key can verify authenticity. Signatures and message authentication underlie the digital infrastructure of the Internet. Without authentication, there would be no cryptographic basis for trusting the origin of a received message.

# 2 Provenance

Authentication of a fixed object is not sufficient when legitimate editing is expected. A provenance claim should certify two things: that some source was authenticated, and that the public object was obtained from it by permitted transformations.

**Definition 3** (Authenticated provenance)**.** *Fix a verification key* $vk$ *and a set* $T$ *of permitted transformations. An object* $y$ *has* authenticated provenance *with respect to* $(vk, T)$ *if there exist a source* $x$*, a signature* $\sigma$*, and a transformation* $f \in T$ *such that*

$$\mathrm{Verify}_{vk}(x, \sigma) = 1 \qquad \textit{and} \qquad y = f(x).$$

*Here* $f$ *may itself represent a composition of several permitted edits.*

A provenance claim is meaningful only after specifying what source is trusted, which transformations preserve provenance, what information about the source and edit history should remain private, and how compromise or revocation affects past claims.

Provenance authenticates an origin and a permitted derivation; it does not by itself establish that the depicted event is true, unstaged, or complete.

# 3 Achieving provenance

There are two basic ways to carry authentication through an allowed transformation. The simpler one does not create a new signature: it keeps the original signature and provides evidence that the published object descends from the signed source. A stronger alternative is to derive a fresh signature on the transformed object itself.

## 3.1 Proofs of provenance

Suppose an editor knows a signed source $(x, \sigma)$ and publishes $y = f(x)$. In the simplest construction, the editor reveals $(x, \sigma, f)$, and the verifier checks

$$\mathrm{Verify}_{vk}(x, \sigma) = 1, \qquad f \in T, \qquad y = f(x).$$

This proves provenance without the signing key, but it may reveal the source and edit history, and it may require the verifier to repeat an expensive transformation.

A succinct proof system can instead certify the same statement

$$\exists\, x, \sigma, f \quad \text{s.t.} \quad \mathrm{Verify}_{vk}(x, \sigma) = 1, \qquad f \in T, \qquad y = f(x).$$

The witness is $(x, \sigma, f)$, not $sk$. Succinct proofs can reduce the communication and verification cost of checking a large source or expensive transformation. If the source, intermediate states, or transformation parameters should remain private, generic *zero-knowledge* (ZK) proofs can certify the same statement without revealing them.

## 3.2 Homomorphic signatures

A stronger interface is to make the transformed object itself carry a valid signature under the original verification key. This cannot be done with an ordinary signature scheme without the signing key, since signing a new message would constitute a forgery. Homomorphic signatures change the functionality so that selected transformations are explicitly allowed.

**Definition 4** (Homomorphic signature)**.** *Let $F$ be a family of allowed functions. A homomorphic signature scheme has a public algorithm* Eval *such that, given valid signatures $\sigma_i$ on $m_i$ and $f \in F$, anyone can compute*

$$y = f(m_1, \ldots, m_k), \qquad \sigma_y \leftarrow \mathrm{Eval}_{vk}\big(f, (m_1, \sigma_1), \ldots, (m_k, \sigma_k)\big)$$

*with* $\mathrm{Verify}_{vk}(y, \sigma_y) = 1$*, without knowing $sk$.*

The security requirement changes accordingly. A signature on a new message is legitimate when that message is obtainable from authenticated data by an allowed transformation; anything outside this closure should remain infeasible to authenticate. More generally, *targeted malleability* restricts malleability to an explicitly permitted family.

**Proposition 1** (Authenticated computation)**.** *From signatures on $m_1, \ldots, m_k$, anyone can derive a signature on $f(m_1, \ldots, m_k)$ for allowed $f$; signatures outside this closure remain infeasible.*

For provenance, this fresh-signature interface is convenient but not necessary. A proof of the provenance relation already suffices to authenticate the derivation; zero knowledge is needed only when the witness should remain private. Homomorphic signatures instead make the derived object verifiable through the ordinary signature interface.

### 3.3 Malleable proofs

There is also a more structural possibility in which the proof itself can be transformed together with the object.

**Definition 5** (Malleable proof)**.** *Let $F$ be a family of allowed transformations. A proof system for a language $L$ is* malleable *for $F$ if, from a proof $\pi$ that $x \in L$ and any $f \in F$, anyone can derive a proof $\pi'$ that $f(x) \in L$, without knowing a witness for $x$.*

Controlled malleability restricts the allowed transformations; *derivation privacy* can additionally hide which earlier proof and which transformation produced the derived proof.

This is stronger than an ordinary provenance proof: the proof itself evolves with the object, rather than being generated afresh. Zero knowledge can also hide the source and derivation.

For images, recent systems prove that a visible image was obtained from a signed private source by a sequence of permitted edits. Folding techniques are useful because repeated edit circuits can be accumulated rather than proved independently, keeping the final proof compact while hiding the source and intermediate images.

Video poses the same problem at much larger scale. Common edits can be represented in algebraic forms favorable to zero-knowledge proving, while specialized encodings reduce the cost of proving consistency with a signed source. In both cases the cryptographic statement is the same: the public media object lies in the permitted transformation closure of an authenticated source.

## 4 Apple Reference Image

Reference Image aims to authenticate a precise claim: a viewable image was developed from pixels captured by a real camera sensor during a bounded time interval, using an approved processing pipeline. This is stronger than merely signing the final JPEG.

At capture, the sensor signs the pixel data and sensor-produced metadata before the operating system can modify them. A Secure Enclave signature binds the remaining capture metadata, and authenticated timestamp tokens provide lower and upper bounds on the capture time. These values form a secure digital negative. During development, Private Cloud Compute verifies the certificate chains, the sensor–device binding, the timestamps, and revocation status before running the image-processing pipeline. The resulting JPEG is then signed by a service key.

This architecture makes three central real-world provenance problems concrete:

- *End-to-end authenticity:* bind the visible image to authenticated sensor capture.
- *Compromise and revocation:* invalidate compromised devices and credentials.
- *Privacy and unlinkability:* verification should not identify or link cameras or photographers.

The system also needs trustworthy capture-time bounds and cryptography that remains verifiable over the expected lifetime of an image. And authenticated capture is not the same as truth: a genuine photograph may still depict a staged, incomplete, or misleading scene.

The system authenticates a carefully defined reference image rather than arbitrary downstream edits. Provenance proofs can complement such a capture mechanism by establishing that a later edited object descends from an authenticated reference image through permitted transformations.

## 5 Historical notes

Security definitions for digital signatures originate with Goldwasser, Micali, and Rivest [1]. Boneh and Freeman developed homomorphic signatures that support computation on authenticated data [2]; Boneh, Segev, and Waters formalized targeted malleability for restricting which transformations are permitted [3].

Chase, Kohlweiss, Lysyanskaya, and Meiklejohn introduced controlled-malleable NIZKs and derivation privacy, with compact verifiable shuffles as a motivating application [4].

For media provenance, Dziembowski, Ebrahimi, and Hassanizadeh built VIMz for private proofs of image manipulation using folding-based zkSNARKs [5]. Frolov et al. extend the direction to signed video in zk-Cinema, with specialized representations and proving techniques for common video edits [6]. Apple's Reference Image is a systems approach that strengthens trusted capture, private processing, timestamping, post-quantum signing, and revocation [7].

## References


[1] S. Goldwasser, S. Micali, and R. L. Rivest. *A Digital Signature Scheme Secure Against Adaptive Chosen-Message Attacks*. SIAM Journal on Computing 17(2), 1988.

[2] D. Boneh and D. M. Freeman. *Homomorphic Signatures for Polynomial Functions*. EUROCRYPT, 2011.

[3] D. Boneh, G. Segev, and B. Waters. *Targeted Malleability: Homomorphic Encryption for Restricted Computations*. ITCS, 2012.

[4] M. Chase, M. Kohlweiss, A. Lysyanskaya, and S. Meiklejohn. *Malleable Proof Systems and Applications*. EUROCRYPT, 2012.

[5] S. Dziembowski, S. Ebrahimi, and P. Hassanizadeh. *VIMz: Private Proofs of Image Manipulation using Folding-based zkSNARKs*. Proceedings on Privacy Enhancing Technologies 2025(2), 344–362, 2025.

[6] A. Frolov, J. Guo, X. Zhao, T. Datta, D. Boneh, and I. Miers. *zk-Cinema: Proving Video Provenance in Zero Knowledge*. IACR Cryptology ePrint Archive, Report 2026/1598, 2026.

[7] Apple Security Engineering and Architecture and Camera & Photos. *Apple Reference Image: A New Approach for Verified Photography*. Apple Security Research, September 15, 2026.