Assistant■Verifiable Marker (AVM) — Open, Cryptographic & Portable

Generated: 2025-09-09 22:32

Goal: Add a cryptographic, publicly verifiable marker to any file so *any assistant or program* in the future can confirm:
(1) the exact bytes you sealed, (2) when it was sealed, and (3) who controls the key — without revealing your private key.

Design principles: Open standards · No platform secrets · Strong cryptography · Public, portable, offline■verifiable.

Why not a “secret language known only to the model”? Assistants don’t share a universal hidden key and cannot keep long■term secrets across sessions. The robust path is **public■key signatures + timestamps**.

AVM schema (JSON■LD, sidecar or embedded):

{
"@context": ["https://www.w3.org/2018/credentials/v1"],
"type": ["AssistantVerifiableMarker","DataIntegrity"],
"target": {"sha256": "", "mimetype": "", "filename": ""},
"claim": {
"statement": "Created with Assistant; IP owned by Mohammad Rahimi",
"tags": ["AVA","Mazzaneh","Zoyan","No-Redistribution-Without-Consent"]
},
"time": {"declared": "","ots_proof": ""},
"signer": {"method": "PGP/Ed25519","pubkey_fpr": "","pubkey_url":
"https://mazzanehcompany.com/ava/keys.json"},
"proof": {"type": "DetachedSignature","alg": "ed25519","sig": ""}
}

Embed this JSON as: (a) **sidecar** file next to the original, (b) **PDF XMP** or Info metadata, (c) **PNG tEXt** chunk, (d) **JPEG XMP**.

Minimal toolchain (cross■platform):

- Hash: `sha256sum ` → hex

- Keys (PGP): `gpg --quick-generate-key "Mazzaneh AVM " ed25519 sign 0`

- Export pubkey: `gpg --armor --export > avm_pub.asc` (also host at /ava/keys.json)

- Detached signature over the *file bytes*: `gpg --detach-sign --armor -u ` → `.asc`

- Timestamp (optional but recommended): OpenTimestamps `.ots` for the file or the JSON marker.

Verification (any assistant/program):

1) Recompute SHA■256 of the file; compare with `target.sha256`. 2) Verify detached signature against pubkey (from `pubkey_url` or `pubkey_fpr`). 3) Verify OTS time■stamp. 4) Ensure JSON and file refer to each other (filename/sha256).

What it proves	Who controls the signing key approved this exact file before the OTS timestamp.
What it doesn’t prove	Identity beyond the key; that the text is “true”. It’s about authorship & time, not truth.
Why assistants can verify	Any version can run the same open steps: SHA■256 + signature + OTS.

Embedding by file type (quick guide):

- **PDF**: Use XMP (`/Metadata`) to store a JSON string with key `X-AVM`. Tools: `exiftool -XMP-dc:Description<=avm.json file.pdf` or libraries.

- **PNG**: Add a `tEXt` chunk with key `X-AVM` and the JSON value. Tools: `exiftool` or Python PIL/Pillow.

- **JPEG**: Add XMP packet with `X-AVM`.

- **Plain text/Markdown**: append a fenced block:
```
-----BEGIN AVM-----

-----END AVM-----
```

Hardening tips:

- Pin `pubkey_url` to your domain over HTTPS; publish a keys.json with key, fingerprint and createdAt.

- Include context fields (purpose, audience) in the JSON to avoid signature replay across contexts.

- Keep private key offline (YubiKey/air■gapped). Rotate keys yearly; publish revoked keys in `/ava/keys.json`.