AEP-Media: Reusable Research Software for Offline Validation of Time-Aware Media Evidence Bundles

Authors

Bin Zhang

Independent Researcher, China

ORCID: 0009-0002-8861-1481

Email: joy7759@gmail.com

Corresponding author: Bin Zhang (joy7759@gmail.com)

Abstract

AEP-Media is reusable research software for local validation of time-aware media evidence bundles. It provides profiles, JSON schemas, command-line validators, examples, offline bundle build and verification, strict declared time-trace validation, adapter-only ingestion, reproducible evaluation matrices, and release packaging material. The software is intended for researchers studying operation accountability, media evidence packaging, provenance boundaries, and reproducible validation artifacts. AEP-Media checks profile identity, required fields, reference closure, artifact hashes, path safety, bundle checksums, declared clock-trace coverage, sample summaries, and offset/jitter thresholds. The v0.1.0 release is archived with a Zenodo DOI and includes a supplementary package with reproducibility notes, selected schemas, examples, reports, and claim-boundary documentation. The claim is local validation and fixture-based adapter ingestion, not external authenticity, legal admissibility, non-repudiation, trusted timestamping, or production deployment.

Keywords

media evidence; operation accountability; research software; provenance; offline validation; evidence bundle

Software availability

- Software name: AEP-Media
- Parent repository/package: `agent-evidence` / `agent_evidence`
- Version: `aep-media-v0.1.0`
- Repository: https://github.com/joy7758/agent-evidence
- GitHub release: https://github.com/joy7758/agent-evidence/releases/tag/aep-media-v0.1.0
- Zenodo archive DOI: `10.5281/zenodo.20107097`

- Zenodo record: https://zenodo.org/records/20107097
- License: Apache-2.0
- Programming language: Python
- Installation: `python -m pip install -e .`
- Operating system: platform-independent Python package; validated in the repository test environment described in the release reports.
- Documentation: `README.md`, `spec/`, `schema/`, `examples/media/`, `demo/`, `docs/reports/`, and `docs/paper/softwarex/`.
- Tests: `tests/test_media_*.py` and related readiness tests.
- Main commands: `validate-media-profile`, `build-media-bundle`, `verify-media-bundle`, `validate-media-time-profile`, `run-media-evaluation`, and `build-aep-media-release-pack`.
- Additional ingestion commands: `ingest-linuxptp-trace`, `ingest-ffmpeg-prft`, and `ingest-c2pa-manifest`.

Metadata

Nr	Code metadata description	Metadata
C1	Current code version	`aep-media-v0.1.0`
C2	Permanent link to code/repository used for this code version	https://github.com/joy7758/agent-evidence
C3	Legal code license	Apache-2.0
C4	Code versioning system used	git
C5	Software code languages, tools and services used	Python; JSON Schema; pytest; Click CLI; Pandoc/Word only for manuscript packaging
C6	Compilation requirements, operating environments and dependencies	Python 3.11+; install with `python -m pip install -e .`; optional LinuxPTP, FFmpeg/ffprobe, and C2PA tools are not required for fixture-based validation
C7	Developer documentation/manual	`README.md`, `spec/`, `schema/`, `examples/media/`, `demo/`, and `docs/reports/`
C8	Support email for questions	joy7759@gmail.com

1. Motivation and significance

Researchers working with media-bearing operations often need to review disconnected artifacts: media files, logs, provenance declarations, timing metadata, policy references, and validation reports. These artifacts are difficult to inspect consistently when they are not bound into a local evidence object with explicit validation semantics. AEP-Media addresses this gap by providing a reusable software layer for turning such materials into locally checkable evidence bundles with machine-readable failure codes.

Existing tools cover adjacent surfaces but not the local validation boundary targeted here. PROV-style models describe provenance relationships [1,2], C2PA describes content provenance and manifest validation states [3], FFmpeg/ffprobe exposes media metadata and timing surfaces [4,5], LinuxPTP supports clock-synchronization workflows [6,7], and GStreamer exposes a PTP clock interface for media pipelines [8]. AEP-Media does not replace these tools. It provides a local evidence-bundle validation layer that binds operation, policy, media artifacts, declared time traces, provenance references, hashes, and failure codes into one reproducible software artifact.

The problem is deliberately bounded. AEP-Media is not a legal evidence platform and does not prove external authenticity. Its purpose is to make the local validation boundary explicit: which fields must be present, which references must close, which files must match declared hashes, which time traces cover a declared window, and which failures are reported when a controlled rule is broken. This local conformance layer is useful before stronger external assurance mechanisms such as signing, trusted timestamping, or deployed capture pipelines are added.

2. Software description

AEP-Media extends the `agent-evidence` package with a media-focused profile and validation path. A media evidence statement binds an actor, subject, operation, policy, constraints, time context, media artifacts, provenance references, evidence notes, and validation expectations into one object. The profile validator checks that the statement is structurally complete and locally consistent.

The offline bundle layer packages a statement and its artifacts into a portable directory. The builder copies artifacts, rewrites paths to bundle-local locations, recomputes hashes and sizes, writes checksums, and emits validation reports. The verifier rejects unsafe paths, missing artifacts, checksum mismatches, and media profile failures. This allows review without relying on the original runtime layout.

The strict-time layer validates declared clock-trace artifacts. It checks trace references, artifact roles, trace hashes, trace profile identity, collection window coverage, sample validity, summary recomputation, offset thresholds, jitter thresholds, and media binding to the declared time context.

The adapter layer is ingestion-only. LinuxPTP-style logs, FFmpeg PRFT-style timing metadata, and C2PA-like manifest metadata can be normalized into AEP-Media reports. Adapter reports distinguish fixture ingestion from optional external-tool reporting and do not imply external verification unless a future environment-specific run records it. This boundary is important because real clock discipline, PRFT extraction, and C2PA signature verification depend on external tools and environment-specific evidence.

Table 1 summarizes the main software components and user-facing functions. Module locations are under the `agent_evidence/` package unless otherwise noted.

Component	Module or location	Function	Command or artifact
Profile validator	`media_profile.py`	Profile checks	`validate-media-profile`
Bundle builder/verifier	`media_bundle.py`	Bundle checks	`build-media-bundle`; `verify-media-bundle`
Strict-time validator	`media_time.py`	Clock-trace checks	`validate-media-time-profile`
LinuxPTP adapter	`adapters/linuxptp.py`	Trace ingestion	`ingest-linuxptp-trace`
FFmpeg PRFT adapter	`adapters/ffmpeg_prft.py`	PRFT metadata ingestion	`ingest-ffmpeg-prft`
C2PA manifest adapter	`adapters/c2pa_manifest.py`	Manifest ingestion	`ingest-c2pa-manifest`
Evaluation runner	`media_evaluation.py`	Evaluation matrices	`run-media-evaluation`
Release pack builder	`media_release_pack.py`	Release evidence pack	`build-aep-media-release-pack`

The quickstart path uses only local fixtures:

```
python -m pip install -e .
agent-evidence --help
agent-evidence validate-media-profile \
  examples/media/minimal-valid-media-evidence.json
agent-evidence build-media-bundle \
  examples/media/minimal-valid-media-evidence.json \
  --out /tmp/aep-media-bundle-check
agent-evidence verify-media-bundle /tmp/aep-media-bundle-check
agent-evidence validate-media-time-profile \
  examples/media/time/minimal-valid-time-aware-media-evidence.json
agent-evidence run-media-evaluation --out /tmp/aep-media-evaluation
```

The implementation relies on structured JSON validation and release packaging practices that are aligned with JSON Schema validation [9,10] and reproducible artifact review expectations [11].

3. Reproducibility and evaluation

AEP-Media is evaluated as a conformance-oriented research software artifact. The evaluation checks whether valid cases pass, invalid and tamper cases fail with expected codes, adapter ingestion remains claim-bounded, and optional-tool absence is reported rather than hidden. It is not a field-deployment benchmark.

For the AEP-Media v0.1.0 release, the final validation run reports:

- targeted AEP-Media tests: 48 passed, 1 warning;

- SoftwareX/readiness tests: 23 passed, 1 warning;
- full test suite: 155 passed, 1 skipped, 15 warnings;
- default evaluation: 18 cases, `unexpected=0`;
- adapter-inclusive evaluation: 26 cases, `unexpected=0`;
- optional-tool reporting evaluation: 23 cases, `unexpected=0`;
- combined adapter and optional-tool evaluation: 31 cases, `unexpected=0`;
- release pack: `PASS aep-media-release-pack@0.1`.

The case categories include valid conformance cases, invalid single-rule cases, bundle tamper cases, strict-time failures, adapter-ingestion cases, and optional-tool reporting cases. Expected failures include missing time context, media hash mismatch, unresolved policy reference, bundle checksum mismatch, path escape, missing clock-trace reference, clock offset threshold exceedance, clock window mismatch, missing PRFT metadata, and declared invalid C2PA-like signature status.

Supplementary file S1 (`AEP-Media_SoftwareX_Supplementary.zip`) contains the supplement README, reproducibility instructions, claim-boundary statement, software inventory, evaluation summary, selected schemas, selected examples, checksums, and release-validation reports used to support the results reported in this paper.

4. Impact

AEP-Media can be reused by three groups. First, researchers studying media evidence packaging, provenance boundaries, and operation accountability can use it as a concrete evidence-object baseline. Second, developers building validation fixtures for media-bearing AI, robotics, simulation, or audit-oriented workflows can reuse the schemas, examples, CLI commands, and pass/fail matrices. Third, reviewers can use the release package to check hashes, references, bundle safety, and declared time traces before stronger external assurance layers are added.

The impact is not that AEP-Media proves authenticity. Its value is that it makes the local evidence object explicit and reproducibly checkable, providing a stable baseline for future real PTP, PRFT, C2PA, signing, or trusted timestamping appendices. The software also complements supply-chain and attestation systems such as in-toto and SLSA [12,13] by focusing on media-bearing operation evidence rather than build provenance alone. Its evidence-object orientation is also consistent with broader digital-object thinking in which durable identifiers, metadata, and reviewable object boundaries matter for reuse [14,15].

5. Limitations and claim boundary

AEP-Media detects inconsistencies inside a declared evidence package, but it does not establish that the original media capture event was truthful, authorized, or unmodified before packaging. It makes no claims of legal admissibility, non-repudiation, trusted timestamping, real PTP proof, full MP4 PRFT parser

coverage, real C2PA signature verification, chain of custody, production deployment, or broad forensic sufficiency.

Optional external-tool paths are reporting paths. The reproducible baseline remains local validation and fixture-based adapter ingestion. Future work should keep the local profile semantics stable while adding environment-specific evidence appendices: LinuxPTP traces captured on equipped Linux hosts, ffprobe output from PRFT-bearing media, C2PA CLI reports for real signed assets, and independent reproduction of the evaluation matrix.

6. Conclusions

AEP-Media is reusable research software for offline validation of time-aware media evidence bundles. It combines a small profile, schemas, examples, command-line validators, bundle verification, strict declared time-trace validation, adapter-only ingestion, reproducible evaluation matrices, a release archive, and supplementary review material. The v0.1.0 release provides a citable baseline for researchers who need local media-evidence validation before pursuing stronger environment-dependent assurance.

CRediT author statement

Bin Zhang: Conceptualization, Methodology, Software, Validation, Investigation, Data curation, Writing - original draft, Writing - review and editing, Project administration.

Declaration of competing interest

The author declares no known competing financial interests or personal relationships that could have appeared to influence the work.

Funding

No specific funding was received for this work.

This work is a software and documentation release; no human participants, animal subjects, or external experimental dataset were used.

Data and software availability

No separate dataset was used. The software, schemas, examples, reproducibility materials, release archive, and supplementary package are available through the GitHub repository, GitHub release, Zenodo archive, and submitted supplementary material.

Declaration of generative AI and AI-assisted technologies in the manuscript preparation process

The author used OpenAI ChatGPT/Codex for manuscript organization, command generation, implementation-facing drafting support, and wording refinement. After using these tools, the author reviewed and edited the content as needed and takes full responsibility for the content of the submitted article.

References

[1] W3C. PROV-Overview: An Overview of the PROV Family of Documents. W3C Working Group Note; 2013. Available: https://www.w3.org/TR/prov-overview/.

[2] Moreau L, Missier P. PROV-DM: The PROV Data Model. W3C Recommendation; 2013. Available: https://www.w3.org/TR/prov-dm/.

[3] Coalition for Content Provenance and Authenticity. C2PA Technical Specification. Available: https://spec.c2pa.org/.

[4] FFmpeg Project. ffprobe Documentation. Available: https://ffmpeg.org/ffprobe.html.

[5] FFmpeg Project. FFmpeg Formats Documentation. Available: https://ffmpeg.org/ffmpeg-formats.html.

[6] LinuxPTP Project. ptp4l Documentation. Available: https://www.linuxptp.org/documentation/ptp4l/.

[7] LinuxPTP Project. phc2sys Documentation. Available: https://www.linuxptp.org/documentation/phc2sys/.

[8] GStreamer Project. GstPtpClock Documentation. Available: https://gstreamer.freedesktop.org/documentation/net/gstptpclock.html.

[9] JSON Schema. JSON Schema: A Media Type for Describing JSON Documents. Draft 2020-12. Available: https://json-schema.org/draft/2020-12/json-schema-core.html.

[10] JSON Schema. JSON Schema Validation: A Vocabulary for Structural Validation of JSON. Draft 2020-12. Available: https://json-schema.org/draft/2020-12/json-schema-validation.html.

[11] Association for Computing Machinery. Artifact Review and Badging - Current. Version 1.1; 2020. Available: https://www.acm.org/publications/policies/artifact-review-and-badging-current.

[12] Torres-Arias S, Afzali H, Kuppusamy TK, Curtmola R, Cappos J. in-toto: Providing farm-to-table guarantees for bits and bytes. USENIX Security Symposium; 2019. Available: https://www.usenix.org/conference/usenixsecurity19/presentation/torres-arias.

[13] SLSA Community. SLSA Provenance. Available: https://slsa.dev/provenance.

[14] Kahn R, Wilensky R. A framework for distributed digital object services. International Journal on Digital Libraries. 2006;6:115-123. doi:10.1007/s00799-005-0128-x.

[15] DONA Foundation. Digital Object Interface Protocol Specification, version 2.0; 2018. Available: https://www.dona.net/doip.