WHITE PAPER · JULY 2026

Verifiable AI Action Records

A Standards-Track Approach to EU AI Act Compliance

Joe Krausz — AgentOracle (TK Collective LLC)
Version 1.0 (draft for review) · agentoracle.co · joe@agentoracle.co

> *This paper is technical commentary intended for compliance, audit, and engineering audiences. It is not legal advice. Statements about Regulation (EU) 2024/1689 are drawn from the operative text of the Regulation; statements about implementations are drawn from public, independently checkable artifacts cited in the References.*

Executive Summary

On 2 August 2026, the record-keeping obligations of Article 12 of the EU Artificial Intelligence Act (Regulation (EU) 2024/1689) begin to apply to high-risk AI systems. Article 12 requires that such systems technically allow the automatic recording of events over their lifetime, and that those records support risk identification, post-market monitoring, and operational oversight. For systems in the Annex III categories — employment and worker management, credit, education, medical devices, and others — the obligations extend to recording the period of each use, the reference data consulted, the inputs that led to a match, and the identity of the natural persons involved in verifying results.

Most organizations plan to meet these obligations with conventional application logs. This paper argues that conventional logs have a structural weakness as compliance evidence: they are mutable, they are not independently verifiable, and they require the auditor to trust the operator that produced them. A log line can be edited, backdated, or deleted, and nothing in the log itself reveals that this happened. A record that cannot prove its own integrity is a weak foundation for a legal obligation whose purpose is proof.

The paper describes an alternative that is implemented and publicly specified today: cryptographically signed verification records ("receipts") built entirely from open standards — JSON canonicalization (RFC 8785), JSON Web Signatures (RFC 7515) with Ed25519 keys (RFC 8037), and published key sets (RFC 7517) — and defined normatively in an IETF Internet-Draft, draft-krausz-verification-state. The format is the first registered profile in the Receipt Profile Registry of ERC-8210 (Agent Assurance), a draft Ethereum standard under active development, and has multiple independent implementations. Records in this format are tamper-evident, carry their timestamp inside the signed payload, and can be verified offline by any third party — including a regulator or auditor — without trusting or contacting the operator that produced them.

Three properties distinguish the approach. **Independence:** multiple unaffiliated issuers can sign the same canonical record, so no single party — including the vendor — has to be trusted. **Reproducibility:** verification

requires only the published specification and public keys; a reference verifier is a one-line install. **Falsifiability:** accuracy claims about the underlying verification pipeline are published against a public academic benchmark that anyone can re-run, with weak categories disclosed alongside strong ones.

The paper maps each subsection of Article 12 to the specific mechanism that addresses it, describes how an auditor verifies records in practice, and closes with the approach's honest limits: a signed record proves what a system recorded, who signed it, and when — it does not by itself make an AI system accurate, and it does not by itself make an organization compliant.

1. The Compliance Evidence Problem

1.1 What Article 12 requires

Article 12(1) of Regulation (EU) 2024/1689 requires that high-risk AI systems "technically allow for the automatic recording of events (logs) over the lifetime of the system." Article 12(2) requires that logging enable the identification of situations that may result in the system presenting a risk within the meaning of Article 79(1) or in a substantial modification; facilitate the post-market monitoring referred to in Article 72; and support the monitoring of operation referred to in Article 26(5). For the high-risk systems listed in Annex III, Article 12(3) additionally requires recording of the period of each use, the reference database against which input data has been checked, the input data for which the search has led to a match, and the identification of the natural persons involved in the verification of results as referred to in Article 14(5). Article 19 requires providers to keep these logs for a period appropriate to the intended purpose, of at least six months.

The Regulation is deliberately technology-neutral about how records are kept. It specifies what the records must enable — identification, monitoring, oversight — not the storage mechanism. That neutrality leaves open the question this paper addresses: which record-keeping architectures can actually serve as evidence when the record itself is questioned?

1.2 Why conventional logs fall short as evidence

Conventional application logs — structured or unstructured, local or centralized — share four structural weaknesses when treated as compliance evidence rather than as operational telemetry.

First, **mutability.** Ordinary log entries can be edited, rotated, truncated, or deleted by anyone with write access to the store, and the resulting record is indistinguishable from one that was never touched. Second, **unbound time.** A timestamp in a mutable record proves nothing about when the event actually occurred; it is simply another editable field. Third, **unverifiable identity.** "Reviewed by: jsmith" is a string, not a proof; any process with write access could have inserted it. Fourth — and most consequential for audit — **trust dependence.** A third party examining conventional logs has no way to confirm their integrity except to trust the organization that produced them. The evidence and the party whose conduct is being evidenced are the same party.

None of this means conventional logs are useless; they remain essential operational telemetry. The claim is narrower: where the purpose of a record is to prove something to a party who does not already trust you — a

market surveillance authority, an auditor, a counterparty, a court — a record that cannot demonstrate its own integrity does not achieve that purpose. The gap Article 12 enforcement will expose is not the gap between organizations that log and organizations that do not. It is the gap between records that assert and records that prove.

2. Requirements for Compliance-Grade Records

Working backwards from the audit scenario — an unaffiliated examiner must be able to rely on the record — five requirements follow.

Tamper-evidence. Any modification to a record after its creation must be detectable from the record itself, without reference to the operator's systems. **Binding.** The elements that matter — the input examined, the verdict reached, the evidence consulted, the time, the identity of each party that vouched for the result — must be sealed together in one unit, so that none can be swapped independently of the others. **Independent verifiability.** A third party must be able to confirm the record's integrity and origin using only public materials: a published format specification and published keys. Verification must work offline and must not require the operator's cooperation, availability, or continued existence. **Issuer independence.** A record vouched for only by the system that produced the output is self-attestation. Compliance-grade records should permit — and where stakes are high, should carry — signatures from parties independent of the operator. **Enumerability and durability.** Records must be storable, countable, and retrievable across the retention period as discrete artifacts, not reconstructed views over a mutable store.

These requirements are not exotic. Each corresponds to an existing, widely deployed open standard. The contribution of the architecture described below is not new cryptography; it is the assembly of standard parts into a record format designed for the audit scenario, published openly so that no single vendor is a dependency.

3. A Standards-Track Architecture

3.1 Design principle: no proprietary trust

Every component in the architecture is an open, published standard with multiple independent implementations. This is a compliance property, not only an engineering preference: an organization that adopts the format is not adopting a vendor. Any party can implement the specification, issue records, and verify records, using reference implementations published under the MIT license or their own independent code.

3.2 Canonicalization: one payload, one byte sequence (RFC 8785)

Digital signatures operate on bytes, and the same JSON object can be serialized into many different byte sequences. The JSON Canonicalization Scheme (JCS, RFC 8785) removes that ambiguity: it defines a single, deterministic byte representation for any JSON payload. Every receipt is canonicalized before signing, which yields a practical guarantee: independent implementations in different languages produce byte-identical

canonical forms — and therefore identical hashes — for the same record. This has been demonstrated across parallel Node.js, Python, and browser implementations, each producing the same canonical bytes and the same SHA-256 digest for shared test vectors.

3.3 Signatures and published keys (RFC 7515, RFC 8037, RFC 7517)

Canonical bytes are signed as a JSON Web Signature (RFC 7515) using Ed25519 keys (RFC 8037). Each issuer publishes its public keys in a JSON Web Key Set (RFC 7517) at a well-known HTTPS location under its own domain. Verification is therefore a local computation: fetch the issuer's published keys once, then confirm any number of records offline. If a single byte of a record has changed since signing, verification fails.

3.4 The verification.v0.3 receipt format

The receipt format itself is defined in draft-krausz-verification-state, filed as an individual-submission Internet-Draft on the IETF Datatracker. (Internet-Drafts are working documents of the IETF; an individual submission is not an adopted working-group item or an RFC, and this paper does not claim otherwise. The relevance of the filing is that the format is publicly and normatively specified, versioned, and open to technical challenge.) A receipt binds, in one signed envelope: a hash of the claim or action examined; the evidence set consulted; a verdict — act, halt, or abstain — with associated confidence signals; the issuer's identity; and the timestamp. The verdict vocabulary is deliberately small. In particular, *halt* is a first-class outcome with its own signed receipt: a compliant system can prove not only what it allowed but what it refused, and when.

3.5 Registry positioning

Publicly announced in July 2026 (with registry scaffolding seeded the month prior), the author of ERC-8210 (Agent Assurance), a draft Ethereum standard under active community development, introduced a Receipt Profile Registry for evidence formats and registered verification.v0.3 as its first entry, citing draft-krausz-verification-state as the normative specification and two independent implementations (AgentOracle and AgentTrust) as meeting the registry's implementer threshold. The registry entry does not confer regulatory status; its significance is narrower and useful — the format is now citable by a stable, content-addressed identifier in a public registry maintained by an independent editor, which is the shape of reference that procurement and audit language can attach to.

4. Multi-Issuer Composed Envelopes

4.1 The self-certification problem

A verification record signed only by the party that produced the output is self-attestation, however sophisticated the machinery behind it. This remains true when the machinery is elaborate: an operator that runs several AI models internally and signs the consensus with its own single key has produced a more considered self-attestation, but a self-attestation nonetheless. The examiner's question — why should I believe this record? — still terminates at one organization and one key.

4.2 Independent issuers over the same bytes

The composed envelope addresses this structurally. Multiple unaffiliated issuers — separate organizations, separate infrastructure, separate published keys — each sign the same canonical bytes. In the current production composition, the issuers perform orthogonal checks rather than repeating one another: one issuer signs a claim-grounding verdict (was the factual claim supported by the cited sources?); a second, AgentTrust, signs a capability-scope verdict (was the action within the agent's authorized skills, tools, and endpoints?); a third, Presidio (a PII and content-screening service operated by PRESIDIO EOOD, presidio-group.eu), signs a screening verdict (does the content violate policy or leak personal data?). Because the checks are orthogonal, the failure modes are largely uncorrelated — three different ways of being wrong, rather than three votes from one room.

The composition rule is deliberately conservative. Under AND_PRESENT, the composed decision is *act* only if every gate present in the envelope is *act*; any single issuer's *halt* collapses the composed decision to *halt*. A published three-signer example demonstrates exactly this property: a payment action approved by both the grounding and capability gates was halted by the screening issuer's PII block, and the halt — with all three signatures over the same canonical bytes — is the signed, verifiable record.

4.3 Current status, stated precisely

Composed envelopes carrying two and three independent issuer signatures over identical canonical bytes have been published publicly and verify end-to-end against each issuer's published JWKS. The two-signer composition (AgentOracle and AgentTrust) is operationally live in production. In the published three-signer envelope, the third issuer's signature is produced against a fixed canonical payload rather than in the live request path; live wiring of the third leg is in progress. The third issuer's public key is already retrievable at screen.presidio-group.eu/.well-known/jwks.json, alongside the AgentOracle and AgentTrust JWKS. A publicly retrievable sample envelope, together with the recompute steps, is available so that any reader can perform the verification themselves rather than relying on this paper's description.

4.4 Self-attested versus independently probed

A useful distinction is emerging in the ecosystem between claims that are *self-attested* (asserted by the party they describe) and claims that are *independently probed* (checked by an unaffiliated party against the artifact itself). The composed envelope is an instrument for moving record-keeping from the first category to the second: each additional independent signer converts one more link in the chain from "trust me" to "check it."

5. Independent Verification in Practice

5.1 The auditor's workflow

Verification of a receipt requires no relationship with any issuer. The examiner obtains the record set from the organization under review, fetches each issuer's published JWKS over HTTPS, and recomputes locally: canonicalize the payload, hash it, and check each signature against the corresponding published key. A

reference verifier is published on PyPI as `agentoracle-receipt-verify` under the MIT license; verification of an envelope is three lines of Python. The same verification can be implemented independently from the specification alone — the reference library is a convenience, not a dependency.

```
pip install agentoracle-receipt-verify

from agentoracle_receipt_verify import verify
verify(envelope, jwks_by_issuer=...)   # offline; no issuer service required
```

5.2 Conformance and independent certification

The specification repository publishes conformance test vectors — accept cases and deliberate reject cases (tampered signatures, mismatched composition rules, unresolvable references) — with parallel verifiers in Node.js and Python that must agree byte-for-byte. Independently of the vendor, the format's conformance vectors have been merged into argentum-core, an unaffiliated maintainer's specification repository, after byte-level verification by that maintainer; and the format's records — canonical bytes, signatures, and on-chain anchor — have been independently recomputed by the operator of a public pre-action-governance conformance board, who published the recompute steps rather than accepting reported results. The purpose of citing this is not the authority of any particular repository or board; it is that the claim "these records verify" has been checked by parties with no stake in the answer, and that any reader can repeat the check.

5.3 Optional on-chain anchoring: proving precedence

Signatures prove integrity and origin; they do not, by themselves, prove that a record existed before a particular external event. For deployments where precedence matters — demonstrating that a verification verdict existed before the action's outcome, rather than being stamped retroactively — receipts can optionally be anchored to a public blockchain transaction. The anchor binds the record's hash into a transaction whose block timestamp is set by an external, operator-independent clock; strict precedence (anchor time earlier than outcome time) is then a recomputable boolean, not an assertion. This layer is optional and additive: receipts are complete evidence artifacts without it, and it is available where the audit posture warrants an external clock.

5.4 Falsifiable accuracy claims

Record integrity and verdict accuracy are different properties, and conflating them is a common failure of vendor claims in this space. The verification pipeline whose verdicts these receipts record is benchmarked against AVeriTeC (Schlichtkrull et al., NeurIPS 2023), a public academic fact-checking benchmark, scoring 57.6% overall on the 2024 development set (57.7% on a held-out split) against published paper baselines of roughly 30%. Per-category results are published in full — 70.6% on Supported claims, 61.6% on Refuted, 27.3% on Not-Enough-Evidence, and 13.6% on Conflicting-Evidence — with the weak categories disclosed alongside the strong, reflecting a calibration that fails skeptical rather than falsely confident, which is the preferable failure mode for regulated content. The dataset, methodology, and harness are public under the MIT license, and the results can be re-run by any reader. The point of this disclosure is not the particular number; it is the epistemic

posture. An accuracy claim that cannot be independently re-run is marketing. The record-keeping architecture described in this paper extends the same principle — verify, don't trust — from the records to the claims made about the system that produces them.

6. Mapping to Article 12

The table below maps each operative requirement of Article 12 to the property of conventional logging it strains against, and the mechanism by which signed receipts address it. The middle column describes what conventional logging categorically cannot prove, as a general observation about mutable log stores; it makes no assertion about any particular organization's practices.

ARTICLE 12 REQUIREMENT	WHAT CONVENTIONAL LOGGING CANNOT PROVE	WHAT SIGNED RECEIPTS PROVIDE
Art. 12(1) — High-risk AI systems shall technically allow automatic recording of events (logs) over the lifetime of the system.	Ordinary logs can be edited or rotated after the fact. Nothing binds a log line to a specific system state at a specific time.	Cryptographically signed receipts are recorded per event over the full lifecycle. Each receipt is sealed at creation; any later modification breaks the signature.
Art. 12(2)(a) — Logs shall enable identification of situations that may result in the system presenting a risk (Art. 79(1)) or in a substantial modification.	Risk classification is typically inferred after the fact from unstructured log context; no verdict is bound to the event itself.	A verdict field (act / halt / abstain) with confidence and evidence signals is recorded and signed on every event, enabling systematic identification of halt and abstain situations.
Art. 12(2)(b) — Logs shall facilitate post-market monitoring (Art. 72).	Monitoring against mutable logs requires trusting the operator; a third party cannot independently verify the record set.	Receipts bind inputs, verdict, and signer identity into an enumerable record set that the provider, a deployer, or an unaffiliated third party can verify independently.
Art. 12(2)(c) — Logs shall support monitoring of high-risk system operation (Art. 26(5)).	Standard logs do not detect tampering; nothing prevents silent post-hoc edits.	Canonicalization (RFC 8785) plus an Ed25519 signature means recomputation confirms nothing was altered after signing.
Art. 12(3)(a) — Recording of the period of each use (start and end date/time). Applies to Annex III systems.	Timestamps in mutable logs are not cryptographically bound; an auditor cannot prove the recorded time is the actual time.	The timestamp is bound inside the signed envelope; because the payload is canonicalized before signing, altering the time breaks the signature. Optional on-chain anchoring proves the record existed before a given external clock reading.
Art. 12(3)(b)–(c) — Recording of reference databases checked against, and input data leading to a match. Applies to Annex III systems.	Free-form log fields make source-and-match evidence difficult to enumerate and impossible to verify without trusting the operator.	The claim hash and evidence set are first-class receipt fields; sources consulted and match outcomes are part of the signed payload.

ARTICLE 12 REQUIREMENT	WHAT CONVENTIONAL LOGGING CANNOT PROVE	WHAT SIGNED RECEIPTS PROVIDE
Art. 12(3)(d) — Identification of the natural persons involved in verification of results (Art. 14(5)).	Reviewer identity in ordinary logs is at best a username string that anyone with write access could have inserted.	In a composed envelope, each verifying party — human-operated or automated — signs with its own published key. Identity is a cryptographic signature, not a string.

On retention: Article 19 requires providers to keep Article 12 logs for a period appropriate to the intended purpose of the system, of at least six months, subject to other Union or national law. Receipts are compact, self-contained JSON artifacts, which makes multi-year retention inexpensive where sector rules extend the baseline.

7. Scope and Honest Limitations

Signed records prove provenance, integrity, and time; they do not make the underlying AI system correct. A receipt demonstrates that a specific check produced a specific verdict at a specific moment and that no one has altered the record since — it does not guarantee the verdict was right. Verdict quality is an empirical property, which is why Section 5.4 insists that accuracy claims be benchmarked publicly and reproducibly rather than asserted.

Verdicts that rely on model judgment are probabilistic. Where a verification pipeline uses AI models to assess whether a claim is grounded, its verdicts inherit the models' error rates. A development direction worth naming — as direction, not as a shipped capability — is *deterministic-first resolution*: resolving claims by direct structural lookup against cited sources where possible, reserving model judgment for genuinely ambiguous cases, and disclosing in each receipt which resolution path produced the verdict. The specification work for this path is public; per-path accuracy figures will be published when the implementation lands, under the same reproducibility discipline as the existing benchmark.

Receipts address the evidence dimension of Article 12; they are not, by themselves, compliance. The AI Act imposes obligations well beyond record-keeping — risk management, data governance, human oversight, transparency, conformity assessment among them — and an organization's overall compliance posture is a legal question for qualified counsel. What a receipt architecture contributes is narrower and concrete: when the record-keeping obligation is tested, the records can prove themselves.

Finally, standards status should be stated plainly: draft-krausz-verification-state is an individual-submission Internet-Draft, not an RFC; ERC-8210 is a draft standard under community development, not a ratified one. The strength of the approach does not rest on regulatory endorsement of these documents. It rests on the fact that every constituent mechanism — JCS, JWS, Ed25519, published JWKS — is a mature, widely deployed open standard, and that every claim made about the implementation is publicly checkable.

8. Implementation Considerations

Integration shape. The natural integration point is pre-action: the system submits a claim or intended action for verification, receives a signed verdict, and acts only on *act*. This produces the strongest records — including signed halts — because the verdict demonstrably preceded the action. Post-hoc recording of already-taken actions is also supported and still yields tamper-evident, independently verifiable records; it simply cannot prove the check came first unless anchoring (Section 5.3) is used.

Storage and enumeration. Receipts are self-contained JSON documents, typically one to a few kilobytes. They can be retained in ordinary object storage, exported for an examiner as a directory of files, and verified in bulk with the reference tooling. No live service is required at verification time.

Auditor experience. An examiner's requirements are the published specification, the issuers' JWKS URLs, and the record set. Everything else is local computation. Organizations preparing for August 2026 can rehearse this: hand a colleague the records and the public materials, and confirm they can verify the set with no further assistance.

For platforms and integrators. Because the format is openly specified with MIT-licensed reference implementations, platforms can issue conformant receipts under their own infrastructure and add independent co-signers where engagements warrant. Procurement and audit language can reference the profile rather than any vendor — for example: *"record-keeping implemented in conformance with the receipt profile registered as verification.v0.3 in the ERC-8210 Receipt Profile Registry, normatively specified in draft-krausz-verification-state."* Referencing the profile keeps the requirement vendor-neutral while remaining precise and testable.

9. Conclusion

Article 12 asks a question most logging infrastructure was never designed to answer: can your records prove themselves? Conventional logs assert; they cannot prove. The architecture described here — canonical bytes, published keys, small verdict vocabulary with *halt* as a first-class outcome, independent co-signers over identical bytes, optional external-clock anchoring, and accuracy claims that anyone can re-run — is assembled entirely from open standards so that adopting the record format never means trusting a vendor.

The underlying principle is older than the Regulation and will outlast it: evidence is what survives examination by someone who does not trust you. Records built to that standard satisfy more than a compliance checkbox — they change the character of the conversation with any examiner from assurance to demonstration. For high-risk AI systems facing the August 2026 deadline, that is the difference worth engineering for.

References

[1] Regulation (EU) 2024/1689 (Artificial Intelligence Act), Articles 12, 14(5), 19, 26(5), 72, 79(1), and Annex III. Official Journal of the European Union.

[2] Krausz, J. "The verification.* Constraint Family: Pre-Action Fail-Closed Gates for AI Agent Decisions," IETF Internet-Draft draft-krausz-verification-state-01 (individual submission). datatracker.ietf.org/doc/draft-krausz-verification-state/

[3] ERC-8210: Agent Assurance — Receipt Profile Registry discussion (registry entry: verification.v0.3). ethereum-magicians.org/t/erc-8210-agent-assurance/28097

[4] RFC 8785 — JSON Canonicalization Scheme (JCS).

[5] RFC 7515 — JSON Web Signature (JWS); RFC 7517 — JSON Web Key (JWK); RFC 8037 — CFRG Elliptic Curve Signatures in JOSE (Ed25519).

[6] Schlichtkrull, M., Guo, Z., Vlachos, A. "AVeriTeC: A Dataset for Real-world Claim Verification with Evidence from the Web," Advances in Neural Information Processing Systems 36 (NeurIPS 2023), Datasets and Benchmarks Track.

[7] AgentOracle evaluation harness and benchmark results (MIT license). github.com/TKCollective/agentoracle-eval-harness and github.com/TKCollective/agentoracle-benchmark

[8] AgentOracle receipt specification, conformance vectors, and reference verifiers (MIT license). github.com/TKCollective/agentoracle-receipt-spec

[9] agentoracle-receipt-verify — reference verifier (Python, MIT license). pypi.org/project/agentoracle-receipt-verify/

[10] Live sample envelope and recompute steps: agentoracle.co ("Verify it yourself") and agentoracle.co/benchmarks

About the author

Joe Krausz is the founder of AgentOracle (TK Collective LLC), author of draft-krausz-verification-state, and contributor of the first registered profile in the ERC-8210 Receipt Profile Registry. AgentOracle provides pre-action factual-claim verification for AI systems, with signed receipts in the format this paper describes. Contact: joe@agentoracle.co.