

RAB: A Replayable-Audit Benchmark for RAG and Agent Systems Operationalising EU AI Act Articles 10, 12, 19*

Jiwon Seo[†]

2026-06-11

Abstract

The EU AI Act’s high-risk-system obligations apply from 2 August 2026 (Article 113), requiring such systems to keep automatic event logs (Art. 12), trace data origins (Art. 10), and retain logs for at least six months (Art. 19). An open survey on agentic systems [W⁺26] explicitly names *realistic execution-trace benchmarks* as a missing piece of the audit ecosystem. Independently, recent runtimes such as ActiveGraph [Nak26] and JAMES demonstrate that event-sourced logs with deterministic replay are constructible; what has been missing is the measurement.

We present **RAB** (Replayable-Audit Benchmark) v0.1.1: a frozen benchmark specification, a deterministic scorer, *two* pre-registered lifecycle scenarios at different scales, and an adapter contract through which any RAG or agent system that can export an append-only audit log is scoreable. RAB defines three metrics — **AC** (Audit Completeness), **RF** (Replay Fidelity, with a separate cost column), and **PC** (Provenance Coverage) — each anchored verbatim to EU AI Act provisions and the W3C PROV vocabulary, each computed without invoking any language model. To our knowledge, RAB is the first benchmark that *scores the exported audit-log artifact* (completeness / log-only replay / provenance coverage) of arbitrary RAG and agent systems against EU AI Act record-keeping articles 10/12/19. Behavioural-determinism harnesses such as DFAH [K⁺26] measure a complementary axis — *what the agent outputs* — and AI-Act benchmarking suites such as COMPL-AI [G⁺24] operationalise the AI Act at the *model* level; RAB occupies the system-level record-keeping axis these two leave open. The headline of a RAB release is a *gap table* across system-under-test categories; no single system’s score is the headline. We measure four SUTs on two scenarios. On scenario-S1 (40 ops): a self-verifying reference adapter (AC/RF/PC = 1.0, 1.0, 1.0), the JAMES audit-native runtime (1.0, 1.0, 1.0), a bolt-on OpenTelemetry-GenAI tracing baseline (0.5, 0, 0; 49 emitted citations, 0 traceable), and a vanilla quickstart + default-logging floor (0.275, 0, 0). The four-row gap reveals *two distinct fail-modes* of non-audit-native instrumentation: default logging catches AC INGEST but misses ANSWER, and bolt-on tracing catches AC ANSWER but misses INGEST — they fail at different canonical categories, and both fail at RF and PC entirely. On scenario-S2 (400 ops, ten times the size, with longer supersede chains and denser cross-references), every cell of the gap table reproduces to the third decimal place: same direction, same per-canonical structure, same disjoint fail-modes; this confirms the gap structure is not a function of corpus size at the order of magnitude tested. A separately pre-registered hypothesis on the RF-cost axis (super-linear scaling of log-only replay cost with corpus size) is not confirmed at this scale and the corresponding claim is withheld per the pre-registration’s locked decision rule. The single-number AC overall is therefore explicitly not the headline; the per-canonical-type and per-axis decomposition is. RAB does not certify compliance, does not claim the audit-native architecture is novel, and does not present JAMES’s score as a finding. The benchmark,

*Pre-registrations locked before any measurement: scenario-S1 measurement protocol at [docs/research/r1-4-preregistration-2026-06-10.md](https://docs.research/r1-4-preregistration-2026-06-10.md) (commit 8a35dee); scenario-S2 design and reporting protocol at [docs/research/r1-phase-3-scenario-s2-preregistration-2026-06-10.md](https://docs.research/r1-phase-3-scenario-s2-preregistration-2026-06-10.md) (commit d21c680; archived independently at Zenodo DOI [10.5281/zenodo.20635130](https://doi.org/10.5281/zenodo.20635130)). All references in the JAMES repository.

[†]Hashevolution, Republic of Korea. ORCID 0009-0002-0007-7860. Correspondence: karu-7@hanmail.net.

with its honesty clauses and two independently pre-registered measurement protocols, is the contribution. All artefacts (spec, both scenarios, scorer, four adapters, eight result triples across two scenarios, OpenTelemetry-GenAI mapping table) are released under Zenodo DOI [10.5281/zenodo.20625533](https://doi.org/10.5281/zenodo.20625533) (v0.4.3 software archive) with the scenario-S2 pre-registration separately archived at Zenodo DOI [10.5281/zenodo.20635130](https://doi.org/10.5281/zenodo.20635130); both are reproducible bit-for-bit from the JAMES repository (tag v0.4.3 for the software archive; commit 753022b for the scenario-S2 measurement).

1 Introduction

Article 12 of Regulation (EU) 2024/1689 [Eur24] requires that high-risk AI systems “allow for the automatic recording of events (logs) over the lifetime of the system” that are “traceable in terms of the functioning of the system”, with logs facilitating ex-post monitoring (Art. 12(2)(b)) and retained at least six months (Art. 19). Article 10(2)(b) requires that data preparation operations record “origin of data”. These requirements apply from 2 August 2026 (Art. 113). Operators of RAG-based assistants, agentic systems, and decision-support tools therefore face a measurable obligation: can their system, given only the exported log, reconstruct what was indexed, what was retrieved, what was synthesised, and what was answered?

The relevant runtimes exist. Event sourcing [Fow05] as a design pattern is decades old, and recent agent runtimes have ported it: ActiveGraph [Nak26] centers an append-only event log as the source of truth and projects a graph from it; JAMES [Seo26] ships an `audit_log` schema with deterministic `reconstruct_graph_at(t)` replay. What is missing is the *measurement*: in the absence of a benchmark, neither a regulator nor a user can compare an audit-native runtime to a quickstart with default logging, nor compare two audit-native runtimes to each other. The agentic-systems survey of [W⁺26] names exactly this gap.

We make four contributions:

1. **RAB v0.1.1 specification (§3).** An abstract audit-log interface, three deterministic metrics, a scenario contract, a reporting format, five baselines, and six frozen honesty clauses (no LLM judge anywhere, log-only RF, operationalise-not-certify, re-verification artefacts, gap-as-headline, spec-versioning).
2. **Two pre-registered scenarios (§4).** Scenario-S1 (*lifecycle-small*, 40 ops, 10 checkpoints, 11 ingest / 4 update / 3 supersede / 2 delete / 20 query) is the v0.1 fixture. Scenario-S2 (*lifecycle-large*, 400 ops, 40 checkpoints, 110 / 40 / 30 / 20 / 200, with nine supersede chains of average length 3.33 and longest length 5, and a cross-reference density of 3.90 doc-id mentions per content-bearing text body) was pre-registered before fixture construction and serves as the cross-scenario confirmation point. Both are deterministic and public-domain.
3. **Adapter contract and four SUT implementations (§5).** Reference is a perfectly-audited in-memory SUT that pins the driver and scorer (1.0×3); JAMES exercises real production code paths (`emit_lifecycle_event` and `reconstruct_graph_at`), workspace-isolated from production state; Baseline-1 is a bolt-on OpenTelemetry-GenAI [Ope24] tracing baseline (the source spec for industry-standard LLM observability) with a hash-pinned mapping table published as a normative artefact; Baseline-0 is a vanilla in-memory RAG with Python-logging-style records (the default-logging floor).
4. **Pre-registered measurement and gap table, replicated cross-scenario (§7).** Two reporting protocols, two cell-validity gate specifications, two sets of prohibitions (no post-hoc spec tweaks, no JAMES-wins framing, no Baseline-0 augmentation, no scenario-blending), and a locked honest-tier ladder were committed before any adapter code was written and before any number was produced. The pre-registration commits are the timestamped evidence. The 4-SUT measurement reveals two distinct fail-modes (§7) of non-audit-native instrumentation,

neither of which the single-number AC overall column makes visible; every cell of the four-row gap table reproduces on scenario-S2 to the third decimal place, ruling out the "S1 might be small enough to cherry-pick" objection without expanding the framing post-hoc.

The benchmark is the contribution. The audit-native architectural class is *not* ours to claim — ActiveGraph published the same architectural primitives independently three weeks before RAB R1.0 [Nak26]; JAMES’s relevant code (`core/lifecycle/replay_audit.py`, `core/lifecycle/replay_graph.py`) shipped in v0.4.2. Adjacent measurement axes also exist and are explicitly orthogonal to ours: DFAH [K+26] scores behavioural determinism (same input → same output reproducibility) at the agent-output level rather than at the log-artifact level, and COMPL-AI [G+24] operationalises the EU AI Act at the model level (robustness, fairness, training-data traceability) rather than at the system-level record-keeping articles RAB targets. What we provide is the missing measurement instrument for the system-level record-keeping axis, and the first 4-SUT data point against it, including the bolt-on-tracing intermediate column that the AC-overall-only strawman objection would have been raised against.

2 Related Work

Audit-native agent runtimes. ActiveGraph [Nak26] explicitly identifies the append-only log as the source of truth, with the working graph as a deterministic projection. JAMES v0.4.2 [Seo26] ships `reconstruct_graph_at(t)` with the audit-only invariant pinned by a release-gating test. Both publications describe the same architectural class. RAB measures it.

Agentic-system trace surveys. The survey of [W+26] categorises trace-related work across reproducibility, auditability, and provenance, and names the open challenge of realistic execution-trace benchmarks. The benchmark category we occupy is the one the survey identifies as empty.

AI Act benchmarking and auditing. Audit Cards [S+25] document *how* an audit is contextualised; RAB measures *what* the audit can reproduce. AIReg-Bench [AIR25] benchmarks LLM *judges* of regulation compliance; RAB has no LLM judge anywhere in scoring (honesty clause 1). Casper et al. [C+24] argue that black-box access is insufficient for rigorous audits; RAB is a white-box instrument over the exported log and quantifies the gap that white-box access enables. The AI Act Evaluation Benchmark [AI 26] evaluates classification triggers in the AI Act (manipulative AI, biometric ID, social scoring) at the system-categorisation level; RAB operates at the per-system audit-quality level. The two are orthogonal.

Provenance and risk-management frameworks. The W3C PROV `wasDerivedFrom` relation [GM13] is the anchor for RAB’s PC metric. The NIST AI RMF [Nat23] provides the broader governance vocabulary; RAB does not implement RMF function controls, but its scores are designed to feed an RMF MEASURE step.

RAG evaluation. Most existing RAG benchmarks (cf. retrieval-augmented generation as introduced by [L+20] and the many domain-specific extensions) measure *answer quality*. RAB measures *audit quality* of the system that produced the answer. The two axes are complementary; a system can score well on RAG quality and fail RAB, or vice versa.

Behavioural determinism vs. log-artifact quality. DFAH [K+26] measures whether a financial agent, replayed against the same input, produces the same output — behavioural determinism, validated across 4,700+ runs over three financial benchmarks. The terminology overlap (“replayable”, “audit”, “harness”) deserves explicit disambiguation: DFAH and RAB

are orthogonal axes, not competing instruments. DFAH scores *what the agent outputs* given an input; RAB scores *what the exported audit log lets you reconstruct* about that output’s chain of evidence. Both axes can fail independently: a behaviourally deterministic agent can still emit an unauditable log (low RAB, high DFAH); an audit-native runtime can reproduce its log-only state and still drift in answer content between runs (high RAB, low DFAH). A complete record of a high-risk RAG or agent system arguably needs both numbers.

Model-level vs. system-level AI Act benchmarking. COMPL-AI [G⁺24] operationalises the EU AI Act into 12 technical requirements + 27 benchmarks at the *model* level (robustness, fairness, traceability of training data). It is, to our knowledge, the first published benchmarking suite anchored to the AI Act; RAB inherits the operationalisation precedent. RAB targets the *system-level* record-keeping obligations of Articles 10/12/19 that COMPL-AI does not score: COMPL-AI evaluates properties of the model artefact, while RAB evaluates the lifetime log artefact of the production system. The two are complementary tracks of a single Act, not substitutes — a deployed system can fail RAB while running an AI-Act-compliant model, and vice versa.

Industry trace standards and LLM-as-debugger. Industry-side tracing has produced two relevant artefact classes that bracket RAB rather than overlap it. OpenTelemetry’s GenAI semantic conventions [Ope24] and adjacent platforms (LangSmith, MLflow) define *format* standards for trace spans but do not score audit quality; they are the natural raw material for an RAB Baseline-1 SUT mapping bolt-on tracing onto Spec §1 (this is the named comparison gap in §10). Industry compliance protocols that specify cryptographic audit-trail formats for AI Act compliance are likewise formats rather than scorers, and are natural candidate SUTs under RAB’s adapter contract. LLM-as-debugger tools that use a model to read a trace and diagnose failure attribution measure the *LLM’s debugging capability*, not the trace’s audit quality, and are explicitly compatible with RAB’s honesty clause H1 (an LLM may consume an RAB-scoreable log without being part of the scorer).

3 RAB v0.1.1 Specification

The full spec is part of the release archive (eval/rab/SPEC-v0.1.md). We summarise its load-bearing pieces; the spec itself, not this paper, is the normative artefact.

3.1 Abstract audit-log interface (Spec §1)

A system under test (SUT) is scoreable iff it can export its audit log as JSONL, one event per line, each line containing `event_id`, ISO-8601 `ts`, `event_type` (from the SUT’s vocabulary; mapped once to RAB’s canonical types via a declared mapping table that is part of the submission), `parent_id` (provenance edge), `inputs_hash`, and `payload`. RAB’s canonical event vocabulary is {INGEST, UPDATE, SUPERSEDE, DELETE, RETRIEVE, RERANK, SYNTH, VERIFY, ANSWER, OTHER}. Unmapped decision-bearing events count against AC — they were executed but not auditable in canonical terms.

3.2 Metrics (Spec §2)

AC — Audit Completeness (Art. 12(1)/(2)). The RAB driver executes the published scenario against the SUT, recording ground truth E_{exec} : every decision-bearing action requested. The SUT’s exported log yields E_{audit} . Matching is greedy, in time order: each executed op matches the first unconsumed log event of the same canonical type whose timestamp falls inside

the op’s $[t_{\text{start}}, t_{\text{end}}]$ window.

$$\text{AC} = \frac{|E_{\text{matched}}|}{|E_{\text{exec}}|} \quad \text{reported overall and per canonical type.}$$

RF — Replay Fidelity (Art. 12(2)(b)). The scenario defines checkpoints $k = 1 \dots K$; at each checkpoint the driver snapshots ground-truth state S_k via the SUT’s live query interface. After the run, the SUT’s *replayer* is given only the exported log and must produce R_k .

$$\text{RF-exact} = \frac{|\{k : \text{canon}(R_k) = \text{canon}(S_k)\}|}{K}, \quad \text{RF-graded} = \frac{1}{K} \sum_k \text{Jaccard}(\text{items}(R_k), \text{items}(S_k))$$

A separate **RF-cost** column reports wall-clock seconds per 1000 log events folded during replay. It is reported alongside but never blended into the score; this is where the cost of audit-native scale enters the picture.

PC — Provenance Coverage (Art. 10(2)(b), W3C PROV). For every ANSWER event, its cited source identifiers must chain back through `parent_id` links to an origin-bearing event (INGEST or SUPERSEDE, since both introduce content):

$$\text{ANSWER} \rightarrow \text{SYNTH} \rightarrow \text{RETRIEVE} \rightarrow (\text{INGEST} \mid \text{SUPERSEDE}).$$

PC = traceable citations/total citations. v0.1 scopes provenance to cited sources only; claim-level provenance is explicit future work.

Canonical form (Spec §2.4). State snapshots serialise as JSON with sorted keys, entity/edge arrays sorted by identifier, UTC ISO-8601 timestamps, floats fixed to 6 decimal places. The scorer’s `canon()` function is identical for S and R . RF-exact is byte-equality after canonicalisation; RF-graded uses the Jaccard on the item set (entity ids and `(src,dst,type)` edge triples).

3.3 Honesty clauses (Spec §6, frozen)

H1 No LLM judge anywhere in scoring.

H2 RF consumes the exported log only — no live-state access.

H3 RAB operationalises Art. 10/12/19 *concepts*; it does not certify compliance, and the spec says so wherever scores are published.

H4 Scores published only with their re-verification artefacts (log, mapping, scorer version).

H5 The benchmark author’s system scoring well is expected and is not the headline.

H6 Spec changes never retro-apply: results carry their spec version.

Clause H6 is what we used when correcting v0.1.0 to v0.1.1 during the reference-adaptor implementation (Section 5).

4 Scenarios

4.1 Scenario-S1 (*lifecycle-small*)

A 40-operation, 10-checkpoint lifecycle over a synthetic public-domain corpus about a fictional research lab (Northbridge Labs). Operations are 11 INGEST, 4 UPDATE, 3 SUPERSEDE, 2 DELETE, and 20 QUERY; identifiers are deterministic; the supersede chain produces both “new-replaces-old” and “later-replaces-earlier-replaces-original” shapes (longest chain length 2); deletes touch both ingested and ingested-then-updated documents.

4.2 Scenario-S2 (*lifecycle-large*)

A 400-operation, 40-checkpoint lifecycle over a synthetic public-domain corpus about a fictional city’s operations (departments, projects, contracts, incidents, policies). Operations are 110 INGEST, 40 UPDATE, 30 SUPERSEDE, 20 DELETE, and 200 QUERY; identifiers follow the deterministic `co-XXX-NNN[-rN]` convention. The supersede plan locks in nine independent chains with an average length of 3.33 and a longest chain of length 5; the cross-reference density (count of *other* doc-id mentions per content-bearing text body) is 3.90, well above the pre-registration’s ≥ 2.5 sentinel. Both the chain shape and the cross-reference density were pinned in the pre-registration before the generator was written so that the larger graph would not be quietly tuned to the SUTs.

Both scenarios are deterministic and public-domain. Each is versioned and SHA-pinned; results cite (`spec v0.1.1`, `scenario S{1,2}`, `scenario-sha`). We never blend scores across scenarios (locked prohibition; honesty clause H6 forbids retro-application of spec changes, and the corresponding scenario-side prohibition forbids retro-fitting protocol claims by averaging across fixtures).

5 Adapters

5.1 Reference adapter

A perfectly-audited in-memory SUT (Spec §5). Its only purpose is to pin the driver and scorer: a system that logs every event with full payload and reconstructs state purely from its log *must* score $AC/RF\text{-}exact/PC = 1.0$. Fault-injection variants (`drop_audit_types`, `corrupt_replay`, `break_provenance`) drop exactly the targeted metric, demonstrating that the three axes are independently observable. The reference adapter implementation surfaced a v0.1.0 defect in the PC origin-bearing rule (the original wording allowed INGEST only, marking supersede-born content untraceable). The defect was caught *before* any measurement was taken; the spec changelog records the correction as v0.1.1. This is honesty clause H6 in action.

5.2 JAMES adapter

The JAMES adapter (`eval/rab/adapters/james.py`) is workspace-isolated: every artefact (RAB log JSONL, lifecycle-event sqlite, optional vector store) lives under a constructor-supplied directory; production audit databases are never touched. Each SUPERSEDE additionally calls the real production code path `core.lifecycle.replay_audit.emit_lifecycle_event` with `EVT_SUPERSEDE_EDGE_CREATED`. Two cross-verification tests pin this bridge: the count of SUPERSEDE JSONL rows equals the count of `lifecycle.supersede.edge_created` rows in the workspace sqlite; and `core.lifecycle.replay_graph.reconstruct_graph_at(t)` reproduces the same supersede edges as the JSONL replay. The adapter therefore demonstrates that the published v0.4.3 JAMES code path actually scores what the gap table reports.

5.3 Baseline-1 adapter (bolt-on OpenTelemetry GenAI tracing)

Baseline-1 (`eval/rab/adapters/baseline1.py`) is the bolt-on-tracing intermediate point named in Spec §5. The same in-memory RAG core that Baseline-0 runs is instrumented with OpenTelemetry GenAI semantic-conventions-shaped [Ope24] spans: each query emits a `retrieval` span (carrying `gen_ai.retrieval.query` and `gen_ai.retrieval.doc_ids`) followed by a child `chat` span (carrying `gen_ai.input.messages`, `gen_ai.output.messages`, `gen_ai.usage.input_tokens`, `gen_ai.response.citations`). The adapter does not depend on the OpenTelemetry SDK at runtime — spans are dict-shaped per the source spec, which keeps the comparison cleanly a function of the mapping table, not of a particular OTel collector. The mapping table itself is a hash-pinned normative artefact, published at `eval/rab/mappings/otel_genai_to_rab.json`

and recorded as the run’s `mapping_table_sha`. Its rationale paragraph in the table file documents (i) why `chat` maps to `ANSWER` rather than `SYNTH` under Spec §1’s 1:1 mapping requirement, and (ii) the structural absence of `INGEST`, `UPDATE`, `SUPERSEDE`, and `DELETE` in the OTel GenAI source spec — a true gap in the source-spec coverage for AI-Act record-keeping purposes, and one of the headline findings of §7.

5.4 Baseline-0 adapter

Baseline-0 is the floor (Spec §5). It is a vanilla in-memory RAG with Python-logging-style records: each operation emits a human string (e.g. `"Indexed document nb-doc-001"`) with no structured payload, no `parent_id` chain, and a mapping table that maps `doc_added` → `INGEST` but cannot distinguish updates, supersedes, deletes, or answers. Its `replay_at` returns the empty state because the log carries no payload to fold. This is the honest representation of what most “default logging” looks like.

6 Pre-registrations

Two measurements in this paper were pre-registered before adapter code or fixture content existed.

R1.4 (scenario-S1, 4-SUT gap). The pre-registration document (`docs/research/r1-4-preregistration-repository` commit `8a35dee`) locks the SUTs, the result-JSON shape (including `log_sha`, `mapping_table_sha`, `scenario_sha`), the cell-validity gates (reference must hit 1.0×3 before publishing other SUTs’ numbers; scenario errors recorded, not papered over), and six prohibitions: no post-hoc spec tweaks, no JAMES-wins framing, no Baseline-0 augmentation, no LLM judge anywhere, no live-state reads in RF, no auto-linking of the measurement to unrelated collaboration scopes. The honest tier ceiling for single-scenario evidence (★★) was declared before any number was produced.

R1.6 Phase 3 (scenario-S2, cross-scenario confirmation). The Phase 3 pre-registration (`docs/research/r1-phase-3-scenario-s2-preregistration-2026-06-10.md`, repository commit `d21c680`, independently archived at Zenodo DOI [10.5281/zenodo.20635130](https://doi.org/10.5281/zenodo.20635130)) was committed *before* the scenario-S2 fixture file existed and *before* the `rab_run.py` dispatch supporting S2 was written. It locks: (i) the scenario shape (operation distribution exactly as quoted in §4, plus the chain and cross-reference sentinels); (ii) the four SUTs (Reference, JAMES, Baseline-0, Baseline-1) with no addition or substitution; (iii) an *absolute prohibition on scenario blending* — per-scenario numbers are reported, never averaged across S1 and S2; (iv) a separate RF-cost hypothesis (cycle γ ’s “graph build $O(N^2)$ ” candidate, brought in from a sibling JAMES finding) with its own threshold pair: JAMES RF-cost ≥ 0.05 s/1k events *and* S1→S2 ratio $\geq 5\times$ together promote the finding to ★★, anything weaker leaves the column definition in the spec but withholds the claim; and (v) a locked honest-tier ladder: ★★★ requires both the gap structure replicating *and* the RF-cost finding clearing, ★★ requires only gap replication, anything else is exploratory. The honest tier the paper now reports against the ladder is the second row (★★, gap replicated, RF-cost claim withheld) and is honest to the rule as written, not relaxed.

7 Results

Table 1 is the v0.4.3 result. The single most important reading is the comparison between Baseline-0 and Baseline-1 in the per-canonical-type columns:

Table 1: Gap table on RAB SPEC v0.1.1 / scenario-S1. The four SUTs span the deployment spectrum from default logging to audit-native. Baseline-0 and Baseline-1 fail at different canonical categories: Baseline-0 catches AC INGEST but misses ANSWER; Baseline-1 catches ANSWER but misses INGEST. Both fail on RF and PC entirely. JAMES = Reference on S1 is *expected* per honesty clause H5; the headline is the per-canonical and per-axis structure of the gap, not the single AC-overall column.

SUT	AC	INGEST	UPDATE	SUPERSEDE	DELETE	ANSWER	RF-exact	RF
Reference (gate)	1.000	1.00	1.00	1.00	1.00	1.00	1.000	1.000
JAMES (audit-native)	1.000	1.00	1.00	1.00	1.00	1.00	1.000	1.000
Baseline-1 (OTel bolt-on)	0.500	0.00	0.00	0.00	0.00	1.00	0.000	0.000
Baseline-0 (floor)	0.275	1.00	0.00	0.00	0.00	0.00	0.000	0.000

Two fail-modes, at different categories. AC = 0.275 (default logging) and AC = 0.500 (bolt-on tracing) are not on the same axis. Both classes are deployed by operators today; both fail to meet the EU AI Act record-keeping obligations RAB scores; but they fail at *different* canonical categories.

Baseline-0’s failure mode: it catches the obvious thing (`logger.info("Indexed document ...")` → AC INGEST = 1.0), but its log carries strings, not events: it cannot distinguish ingest from update from supersede, does not record deletes structurally, and emits no canonical ANSWER event at all. AC UPDATE, SUPERSEDE, DELETE, ANSWER = 0 are the consequences.

Baseline-1’s failure mode is structurally opposite. The OpenTelemetry GenAI source spec targets runtime LLM-call observability: its `chat` span maps cleanly to ANSWER (AC ANSWER = 1.0), but the source spec has no corpus-lifecycle vocabulary — there is no `ingest_document` or `supersede_document` span in OTel GenAI. AC INGEST, UPDATE, SUPERSEDE, DELETE = 0 follow directly from the source-spec coverage, not from any deficiency in the adapter or the mapping table. AC overall improves from 0.275 to 0.500, a real +0.225, but the structural trade is INGEST observability for ANSWER observability.

49 citations emitted, 0 traceable. Baseline-1’s `chat` span carries `gen_ai.response.citations`, and the mapping table surfaces them as the SPEC `payload.citations` field, so PC’s denominator on this scenario is 49 (not zero). PC’s numerator is zero: every ANSWER → chain → RETRIEVE chain walks correctly, but the chain has nowhere to terminate because no INGEST event exists in the log to be the origin. This is, in one number, why corpus-lifecycle vocabulary is load-bearing for provenance.

RF = 0 across both non-audit-native tiers. Both Baseline-0 (no payload at all) and Baseline-1 (OTel spans carry token counts and messages but not document state) replay to the empty state at every checkpoint. RF is the metric on which audit-native instrumentation is qualitatively separate from both non-audit-native classes: bolt-on tracing closes part of the AC gap but closes none of the RF gap.

JAMES = Reference on S1. Expected per H5. Both top rows have AC/RF/PC = 1.000 across the board. The Reference adapter pins driver and scorer correctness; JAMES independently reproduces those numbers through real production code paths (`emit_lifecycle_event` and `reconstruct_graph_at`) under workspace isolation, with two cross-verification tests pinning the bridge.

Table 2: Cross-scenario gap table: scenario-S2 (lifecycle-large, 400 ops, 800 log events for audit-native SUTs, 400 for non-audit-native). Every cell reproduces the S1 result of Table 1 to the third decimal place. The two fail-modes (Baseline-0 catches **INGEST** only; Baseline-1 catches **ANSWER** only; strictly disjoint canonical-type catches) are stable at $10\times$ scale. **JAMES** = Reference within scenario continues to hold; the scenario-system co-design caveat carried forward from §8 is therefore unchanged in shape but is now *partially* mitigated by the cross-scale replication.

SUT	AC	INGEST	UPDATE	SUPERSEDE	DELETE	ANSWER	RF-exact	RF
Reference (gate)	1.000	1.00	1.00	1.00	1.00	1.00	1.000	
JAMES (audit-native)	1.000	1.00	1.00	1.00	1.00	1.00	1.000	
Baseline-1 (OTel bolt-on)	0.500	0.00	0.00	0.00	0.00	1.00	0.000	
Baseline-0 (floor)	0.275	1.00	0.00	0.00	0.00	0.00	0.000	

7.1 Scenario-S2 cross-scenario gap table

Table 2 reports the scenario-S2 measurement. The numbers are identical, to the third decimal place, to the corresponding cells in Table 1: same direction, same fail-modes, same per-canonical-type structure, with the scale of the log enlarged by an order of magnitude ($40 \rightarrow 400$ ops; $80 \rightarrow 800$ events for audit-native SUTs, $40 \rightarrow 400$ for non-audit-native). On this scenario, Baseline-1 emits 597 citations across its 200 answers, with the same 0 traceable as on S1, because the same OTel-GenAI source-spec gap (no **INGEST** vocabulary) leaves the citation chain with no origin event to terminate at. This is not the kind of replication that survives mild perturbation by luck; it is what one expects when a gap is structural rather than scenario-specific.

What the cross-scenario replication does and does not show. It rules out the simplest objection: that the v0.4.3 result was an artefact of scenario-S1’s small size. It also strengthens the per-canonical disjoint fail-mode reading, since the same disjoint catches reappear on a fixture with a different domain (city operations rather than a research lab), a different chain shape (longest length 5 rather than 2), and a different cross-reference density. What it does *not* do is mitigate the *within-class* concern of §8: **JAMES** = Reference still holds on S2, but the canonical vocabulary that S2 uses is the same one S1 uses, and the same one JAMES’s internal lifecycle taxonomy uses; ruling out unconscious shaping of the spec by the SUT it most closely matches needs a future scenario whose event-vocabulary basis differs (queued as scenario-S3 cross-lingual in §10) and the replication-invite track to runtimes whose taxonomies were defined independently.

RF-cost (the scale-axis hypothesis). The pre-registration’s §4 hypothesised that JAMES’s RF-cost (wall-clock seconds per 1000 log events folded during replay) would activate visibly at the larger scale, with two thresholds locked: ≥ 0.05 s/1k events *and* an S1→S2 ratio of $\geq 5\times$ together promote the finding to $\star\star$; either condition unmet leaves the column definition in the spec but withholds the claim. The observed JAMES RF-cost is 0.0387 s/1k events on S1 and 0.1081 s/1k events on S2 — the first sentinel clears (current cost is above the 0.05 threshold) but the ratio is $2.79\times$, well below the locked $5\times$. The Reference adapter’s RF-cost grows from 0.0013 to 0.0051 s/1k events, a $3.92\times$ ratio in the same neighbourhood. The honest reading is that a single S2 data point cannot distinguish "linear-ish with a constant factor" from "an early stretch of an $O(N^2)$ curve that has not yet compounded"; the locked decision applies and the claim is withheld. RF-cost remains a defined column of the spec and the present numbers are honest data points, but the cycle- γ candidate finding does not promote on the basis of two scenarios in the present ratio range. A future scenario at $\geq 4\times$ the S2 operation count would be the cleanest next data point.

Honest tier: $\star\star$ cross-scenario partial: gap structure confirmed across S1 and S2 to the third

decimal place; RF-cost finding not confirmed; cross-vocabulary-basis and external-replication mitigations of the scenario-system co-design caveat queued in §10. ★★★ is the ladder’s top rung; the pre-registration locked it to require *both* the gap and the RF-cost leg, and this paper reports against the ladder as written.

8 Limitations and Threats to Validity

We declare the limitations of v0.1.1 deliberately and in advance of release, per honesty clause H5:

- **Two same-vocabulary scenarios; cross-bench still future.** v0.1.1 now ships two scenarios at different scales (S1 at 40 ops, S2 at 400 ops). The cross-scenario replication on S2 rules out the simplest objection (S1 might have been small enough to cherry-pick) but does *not* address vocabulary-basis variation: both fixtures use the same canonical event set {INGEST, UPDATE, SUPERSEDE, DELETE, ANSWER}. Generalisation to multilingual content (scenario-S3 pre-registered separately at the §10 track), to long-running production traces, and to adversarial inputs is future work. Adding a scenario whose vocabulary basis is independently chosen is the path to ★★★. In particular, both S1 and S2’s `e_exec` ground-truth canonical sets exclude RETRIEVE / SYNTH / RERANK: AC does not score them, even though SUTs may emit them. Baseline-1 does emit RETRIEVE events in its log (49 on S1, 597 on S2), but those events score zero towards AC because they do not match a ground-truth op; a future scenario that lifts retrieval to driver-side ground truth would let bolt-on tracing earn credit on that axis.
- **Three SUT classes, no third-party audit-native runtime.** The gap is now audit-native vs. bolt-on tracing vs. default-logging, but the audit-native column has only one occupant (JAMES) measured by the benchmark authors. A second audit-native point from a runtime whose taxonomy was defined independently — ActiveGraph [Nak26] is the obvious candidate — is the strongest available mitigation of the within-class self-evaluation concern and is the highest-priority replication invite (§10).
- **Cited-source provenance, not claim-level.** PC v0.1 traces source identifiers, not the claim a citation supports. Claim-level provenance is a future spec version with its own honesty review.
- **Synthetic prose.** Northbridge Labs prose is public-domain and licence-friendly but does not exercise PII redaction, multilingual disambiguation, or table-and-figure citation. The benchmark is designed to be extended, not assumed to cover these axes today.
- **Author-of-benchmark SUT.** JAMES is one of four SUTs we measure. Honesty clauses H3 and H5, the gap-as-headline reporting protocol, and the inclusion of Baseline-1 (which directly tests whether the AC-overall column tells the whole story) together control for this conflict of interest. A peer-replication track is anticipated (R1.6, separate scope).
- **Scenario-system co-design risk — partially mitigated by cross-scenario, vocabulary-basis still open.** The canonical event vocabulary (INGEST, UPDATE, SUPERSEDE, DELETE) shared by S1 and S2 is isomorphic to JAMES’s own lifecycle taxonomy. JAMES’s 1.000×3 on both scenarios is therefore part-by-design and part-by-measurement: the gap to Baseline-0 / Baseline-1 is informative regardless, and the cross-scenario replication on S2 (Table 2) rules out the simplest size-cherry-picking objection. The within-class equality (JAMES = Reference within scenario) still cannot, on two same-vocabulary scenarios, rule out unconscious shaping of the spec by the SUT it most closely matches. Two independent mitigations remain queued as Future Work (§10): scenario-S3 with a different event-vocabulary basis (cross-lingual), and external replication invites to runtimes whose taxonomies were defined independently

of JAMES's. The Baseline-1 mitigation listed in the v1.0 draft of this paper has since been delivered; the scenario-S2 cross-scenario mitigation listed in v1.0–v1.2 has likewise been delivered (this revision). The remaining within-class concern is the audit-native column on a same-vocabulary scenario family, and the queued vocabulary-basis variant is the targeted mitigation.

- **Mutation-site wiring is partial in v0.4.3.** The JAMES SUT's 1.000 holds for the workspace-isolated RAB adapter path; v0.4.2's published audit-only invariant requires a follow-up cross-cutting cycle to wire every production mutation site (T1, T2, T2.D, T6, T7) through `emit_lifecycle_event`. Until that cycle lands, RAB's JAMES score reflects what the RAB adapter exercises rather than every production code path. The wiring follow-up is queued; once it lands, an additional measurement (`james-production-S1`) can be reported alongside the workspace-adapter number.

9 Reproducibility

All artefacts are released under Zenodo concept DOI [10.5281/zenodo.20625533](https://doi.org/10.5281/zenodo.20625533) (v0.4.3 version-specific DOI; the scenario-S2 measurement is post-v0.4.3, on commit 753022b, scheduled to be archived under a forthcoming v0.4.4 DOI). The scenario-S2 pre-registration is independently archived at Zenodo DOI [10.5281/zenodo.20635130](https://doi.org/10.5281/zenodo.20635130). The release archive plus the scenario-S2 measurement commit contain:

- `eval/rab/SPEC-v0.1.md` — the frozen specification.
- `eval/rab/scenarios/{s1_lifecycle_small.json,s2_lifecycle_large.json}` — both scenarios, hash-pinned. The S2 fixture is reproducible from the bundled generator (`scripts/research/build_rab`) the generator's announced byte-string SHA matches the fixture content, and the `scenario_sha` recorded in every `S2 result.json` is the SHA returned by `Path.read_text(encoding='utf-8')` (universal-newline normalised) so that all SUT runs agree on the same number on every platform.
- `eval/rab/{driver,scorer}.py`, `eval/rab/adapters/{reference,baseline0,baseline1,james}.py` — the deterministic driver, scorer, and four adapters.
- `eval/rab/mappings/otel_genai_to_rab.json` — the hash-pinned mapping table for the OpenTelemetry-GenAI Baseline-1 SUT, recorded as `mapping_table_sha` in every Baseline-1 `result.json`.
- `reports/rab/<sut>-S{1,2}-<ts>.{result.json,log.jsonl,mapping.json}` — the SPEC §4 re-verification triple for each SUT on each scenario (eight triples total: four SUTs \times two scenarios). Re-running the scorer on the same artefacts reproduces the numbers bit-for-bit.
- `tests/test_rab_{benchmark,baseline0,baseline1,james_adapter,scenario_s2}.py` — 51 tests pinning self-verification, the floor, the bolt-on tier, the audit-native bridge, and the scenario-S2 fixture shape (op distribution / chain shape / cross-reference density / Reference 1.000×3 gate).

The minimal reproduction recipe is:

```
git clone https://github.com/Hashevolution/James-RAG-Evol
cd James-RAG-Evol && git checkout 753022b # v0.4.3 + scenario-S2
python -m pytest tests/test_rab_benchmark.py \
                tests/test_rab_baseline0.py \
                tests/test_rab_baseline1.py \
                tests/test_rab_james_adapter.py \
```

```

        tests/test_rab_scenario_s2.py
for sut in reference james baseline1 baseline0; do
    python scripts/research/rab_run.py --sut $sut --scenario S1
    python scripts/research/rab_run.py --sut $sut --scenario S2
done

```

10 Future Work

Four explicit tracks remain out of scope for v0.1.1 after the Baseline-1 SUT (v1.2 revision) and the scenario-S2 cross-scenario confirmation (this revision) were delivered:

1. **Scenario-S3 (cross-lingual) and beyond.** A multilingual corpus (so the canonical-event vocabulary is no longer single-language, addressing the targeted residue of the scenario-system co-design concern in §8), an adversarial-input track, and a scenario at $\geq 4\times$ the scenario-S2 operation count to provide a clean third data point on the RF-cost axis (where the $\star\star$ promotion threshold remains unmet at the current scenario range). A future scenario that lifts RETRIEVE / SYNTH / RERANK to driver-side ground truth would additionally let Baseline-1’s emitted-but-currently-unscored RETRIEVE events contribute to AC, providing a finer measurement of where on the per-canonical axis bolt-on tracing earns credit.
2. **Replication invites.** We will invite the authors of ActiveGraph [Nak26] and DFAH [K+26], and other audit-native runtimes and trace-scoring teams, to submit SUT adapters and contribute results. This is a separate collaboration scope; results from such replications, when they arrive, will be reported under the same Spec §4 protocol. A second audit-native point from a runtime whose taxonomy was defined independently of JAMES’s is the single strongest available mitigation of the within-class self-evaluation concern that remains after the cross-scenario replication.
3. **Mutation-site wiring follow-up.** Lifting JAMES’s 1.000 from the RAB adapter path to every production mutation site (the v0.4.2 carry-over described in §8) and reporting a separate `james-production-S{1,2}` number alongside the workspace-adapter ones.
4. **SPEC v0.1.2 candidate items.** A normative clarification of `scenario_sha`’s computation (i.e. explicitly anchored to the universal-newline-normalised `Path.read_text` form so that cross-platform agreement is part of the spec rather than an adapter-level convention) is the cleanest candidate single change. A scenario-size-guidance annotation in §3 is a possible accompanying clarification.

11 Conclusion

RAB v0.1.1 is, to our knowledge, the first benchmark that scores the exported audit-log artifact of arbitrary RAG and agent systems against the system-level record-keeping articles of the EU AI Act — the axis left open by behavioural-determinism harnesses [K+26] and model-level AI Act benchmarking suites [G+24]. It does not introduce a new architecture; ActiveGraph [Nak26] and JAMES [Seo26] already realise the architectural class, independently. What it introduces is a deterministic, pre-registered, externally-anchored benchmark with a frozen specification and a published re-verification protocol, just before the EU AI Act’s high-risk obligations enter force. The four-SUT measurement on two pre-registered scenarios at an order-of-magnitude scale gap makes the result’s shape, not its single-column AC overall, the headline: default logging and bolt-on tracing both fail to meet the AI Act record-keeping bar, but they fail at *different* canonical categories; both fail completely on the RF and PC axes that audit-native instrumentation occupies alone; and the four-row gap structure reproduces from scenario-S1

to scenario-S2 to the third decimal place, ruling out the simplest size-cherry-picking objection without expanding the framing post-hoc. The honest-tier ladder commits the headline to the second rung (★★ cross-scenario partial), because the separately pre-registered RF-cost hypothesis is below threshold and the within-class self-evaluation concern is only partially mitigated. The gap table is the headline. The benchmark is the contribution.

Author contributions and conflicts of interest

Single-author work by Jiwon Seo (Hashevolution). The author is also the maintainer of PROJECT JAMES, one of the SUTs measured. Honesty clauses H3 and H5, the pre-registered reporting protocol, the workspace-isolated JAMES adapter (production audit databases not touched), and the inclusion of a self-verifying reference SUT together control for this conflict of interest. The same benchmark applied to other audit-native runtimes is explicitly invited (Section 10).

Data and code availability

All code, the specification, the scenario fixture, the deterministic scorer, the three adapters, the pre-registration document, the cross-verification tests, and the 9 measurement artefacts are publicly available under the MIT license at the JAMES repository, release tag v0.4.3, and mirrored on Zenodo at [10.5281/zenodo.20625533](https://doi.org/10.5281/zenodo.20625533).

Acknowledgements

This work was developed in dialogue with Claude (Anthropic) as a pair-collaborator. The author thanks LEO (Younghu, AIS Edu HK) for the foundational research collaboration that shaped the JAMES platform on which RAB’s audit-native SUT is built.

References

- [AI 26] AI Act Evaluation Benchmark authors. AI Act Evaluation Benchmark: An open, transparent, and reproducible evaluation dataset for nlp and rag systems, 2026.
- [AIR25] AIReg-Bench authors. AIReg-Bench: Benchmarking language models that assess ai regulation compliance, 2025.
- [C⁺24] Stephen Casper et al. Black-box access is insufficient for rigorous ai audits, 2024.
- [Eur24] European Parliament and Council. Regulation (eu) 2024/1689 of the european parliament and of the council of 13 june 2024 laying down harmonised rules on artificial intelligence (artificial intelligence act). Official Journal of the European Union, L series, 2024. Articles 10, 12, 19 anchor the RAB metrics. Effective 2026-08-02 per Article 113.
- [Fow05] Martin Fowler. Event sourcing. Online essay, 2005. Background terminology used throughout the spec.
- [G⁺24] Philipp Guldimann et al. COMPL-AI framework: A technical interpretation and LLM benchmarking suite for the EU artificial intelligence act, 2024. ETH Zürich + INSAIT + LatticeFlow collaboration. First technical interpretation of the EU AI Act translated into measurable requirements at the MODEL level (robustness, safety, diversity, fairness). RAB targets the SYSTEM-LEVEL record-keeping obligations of Articles 10/12/19 that COMPL-AI does not score; the two are complementary tracks of a single Act.

- [GM13] Paul Groth and Luc Moreau. PROV-Overview: An overview of the PROV family of documents. W3C Working Group Note, 2013. `wasDerivedFrom` relation anchors RAB’s PC metric.
- [K⁺26] Raffi Khatchadourian et al. Replayable financial agents: A determinism-faithfulness assurance harness (DFAH) for tool-using LLM agents, 2026. DFAH: measures behavioral determinism (decision determinism / trajectory determinism: same input \rightarrow same output reproducibility) across 4,700+ agentic runs on three financial benchmarks (compliance triage, portfolio constraints, DataOps exceptions). Key finding: decision determinism and task accuracy are not detectably correlated. Adjacent axis to RAB: DFAH scores what the agent outputs; RAB scores the audit-log artifact the agent emits. The two are independently failable.
- [L⁺20] Patrick Lewis et al. Retrieval-augmented generation for knowledge-intensive NLP tasks, 2020.
- [Nak26] Yohei Nakajima. The log is the agent: Event-sourced reactive graphs for auditable, forkable agentic systems, 2026. Independent co-invention of the event-sourced log + deterministic replay architecture that JAMES T5 also implements; RAB measures this architectural class.
- [Nat23] National Institute of Standards and Technology. Artificial Intelligence Risk Management Framework (ai rmf 1.0). NIST AI 100-1, 2023.
- [Ope24] OpenTelemetry GenAI Working Group. OpenTelemetry GenAI Semantic Conventions. Semantic-convention specification, 2024. Industry de-facto trace schema for LLM-based systems. Provides the span vocabulary that an RAB Baseline-1 adapter maps onto Spec §1’s canonical event types — the bolt-on tracing comparison point named in Spec §5.
- [S⁺25] Leon Staufer et al. Audit cards: Contextualizing AI evaluations. 2025.
- [Seo26] Jiwon Seo. PROJECT JAMES v0.4.3 — RAB v0.1.1 (replayable-audit benchmark) + cycle γ multi-hop arc closure. Software, Zenodo, 2026. Source: <https://github.com/Hashevolution/James-RAG-Evol> (release tag v0.4.3).
- [W⁺26] Yiqi Wang et al. From agent traces to trust: Evidence tracing and execution provenance in LLM agents, 2026. Survey of evidence tracing and execution provenance in LLM agents; its “open challenges” list explicitly names “realistic execution-trace benchmarks” alongside unified trace schemas, claim-level / semantic provenance, provenance-aware safety mechanisms, recovery-oriented evaluation, and privacy-aware audit infrastructure. RAB is a response to the execution-trace-benchmark item.