Independence Is Not Multiplicity: A Reproducible Sixteen-Scenario Study of Witness Provenance in Audit Logs

Sepehr Ghanbari
Independent Researcher, Ario Project (no institutional affiliation claimed)
Correspondence: sepehrqanbari@gmail.com
ORCID: https://orcid.org/0009-0006-4710-3914
Research preprint · 5 November 2026 · Not peer reviewed · Author-review manuscript

Abstract

The existence of multiple agreeing records does not establish that they were generated, stored, or controlled independently. We examine the concrete engineering consequence of this distinction in a pinned open-source audit-provenance implementation. The study compares two explicitly defined decision rules over sixteen synthetic scenarios: a deliberately limited baseline that treats distinct storage objects as conditional corroboration, and a three-outcome provenance gate constructed from the original implementation's administrative-control, source-dependence, storage, capture, and custody findings. Each scenario has an author-declared expected class: conditional independence within the laboratory model (two scenarios), known dependence (seven), or insufficient information (seven). In the completed exploratory run, the storage-only baseline agreed with four of sixteen scenario labels and made twelve unsupported conditional-independence endorsements; the provenance gate agreed with all sixteen and made none. The labels and rule were constructed for this small experiment, so the result is a demonstration of these decision boundaries, **not** an estimate of detection accuracy in deployed systems. We also reproduced 280 passing tests from five existing Python modules and independently matched three source-file Git blob identifiers against a fixed GitHub commit. A 148-file digest audit verified internal correspondence within the supplied readable export. Neither the checks nor the experiments authenticate historical events, custody, clocks, persons, or authorization. We provide complete scenarios, source hashes, code, raw results, and unsuccessful preliminary execution details to make the exact claim inspectable and falsifiable.

Keywords: audit provenance; source dependence; corroboration; common control; append-only records; evidence integrity; reproducible software experiments; uncertainty; artificial intelligence; synthetic testing.

Disclosure of generative AI assistance. OpenAI GPT-6 (November 2026) assisted substantively with preparing this manuscript, structuring the experimental comparison, writing and debugging the accompanying Python study harness, running local commands, and summarizing results. The tool is not an author. The human author is the originator of the Ario project and must critically inspect, verify, correct, and approve the experiment, literature citations, interpretations, disclosures, and final submission before publication. No claim of completed independent human verification is made by this author-review draft.

1. Introduction

An audit may report that two records agree, two storage paths exist, and two cryptographic digests match, while leaving open whether the records descend from a shared source or are controlled by one principal. This is a problem of **what the evidence establishes**, not of whether the audit software is useful. A digest can detect a byte-level mismatch relative to a fixed reference; it cannot identify who generated the bytes. Distinct directories can hold copies of one record. Two nominal witnesses can be managed by the same administrator. A complete-looking provenance graph can be incomplete or authored by the very party whose claims it is meant to test.

This paper reports a deliberately narrow empirical comparison of the consequences of treating visible multiplicity as independence. The research object is the synthetic witness-provenance experiment bundled in a pinned Ario repository snapshot [1,2]. Ario is a research project concerned with evidence lineage, inspectable decisions, uncertainty and authorization boundaries; the present study neither attempts to validate that larger program nor asks whether an artificial system possesses subjective experience. It asks a testable software question: **how do two specified decision rules**

classify constructed cases when storage distinctness, source identity, administrative control and evidence completeness vary?

The principal new contribution, relative to an earlier October 2026 working-paper draft, is a **fully disclosed sixteen-case comparative exercise**, including raw outputs, a transparent deliberately weak baseline, and explicit ground-truth limitations. The secondary contribution is a fresh execution of a fixed 280-test subset and spot verification of source identities against GitHub. No favorable result is promoted into a claim about real custody independence or general security. The selected study is exploratory and was not independently preregistered; the expected labels, rule definitions and revisions remain available in the executable artifact.

2. Prior work and scope

Database provenance research distinguishes the origin and derivation of data from the fact that a record is present in a database. Buneman, Khanna and Tan's why/where distinction [3], provenance semirings [4], and the W3C PROV data model [5] establish useful ways to express dependencies. These representations do not, by themselves, authenticate the claimed graph or prove that every relevant controller and copying path was recorded. This is especially relevant where a provenance assertion can itself be written by an interested party.

Certificate Transparency provides a complementary integrity example: inclusion and consistency structures are valuable under specified assumptions, whereas questions about external observation, equivocation and monitoring require additional mechanisms [6]. Security guidance similarly separates audit generation from protection of audit records [7]. These sources help frame why a local hash match or multiple files should not be interpreted as a stronger statement about honest origins. The National Academies' distinction between computational reproducibility and independent replication [8] also matters here: re-executing the same released code on the same selected inputs is not independent replication of a general scientific hypothesis.

This work evaluates no statistical sampling-based hallucination-detection method. Existing consistency-based approaches such as SelfCheckGPT [9] pursue a different inferential task. Agreement among generated answers may provide a useful diagnostic signal; it does not, without additional assumptions, provide authenticated historical witnesses. We use the term *evidence laundering* descriptively for an unsupported promotion from representation or agreement to source independence, event truth, or authorization. We do not introduce a new cryptographic construction, a proof of protocol security, or a benchmark of deployed AI systems.

3. Research questions and operational definitions

RQ1 (local reproduction). Can the source export's declared text-file hashes be checked and the selected five existing test modules re-executed in a fresh local environment?

RQ2 (comparative classification). Across sixteen specified synthetic cases, how often does a storage-distinctness rule endorse conditional witness separation compared with a provenance-dependent gate?

RQ3 (bounded interpretation). Which observations can be justified by these runs, and which claims concerning custody, independent witnesses, real history, AI identity, or operational authority remain outside scope?

We define three classes for *this study only*:

- **CONDITIONAL:** the supplied laboratory graph asserts two distinct modeled source/control paths, distinct storage objects, complete control and source inventories, ordered captures, and custody links. This is **conditional model separation**, never real-world attestation.
- **DEPENDENT:** the constructed graph explicitly contains a shared source, administrative control, common storage identity, or common backup/root relationship that defeats the conditional-separation claim.
- **UNKNOWN:** completeness, admissible graph origin, unique artifact identity, or custody support is missing. No conclusion that the witnesses are separate or colluding follows.

The scenario *expected* classes are author-defined construction labels, not independently verified field truth. The controlled comparison tests whether the two decision rules honor those constructed distinctions. It does not measure population prevalence, sensitivity to arbitrary attacks, false-positive rates in production, or any model's epistemic reliability.

4. Materials and protocol

4.1 Pinned source and chain of custody

The examined primary source was the 2,857,981-byte Ario_READ.txt text-readable snapshot archived for the 25 October 2026 Git revision f562e14cd8ca7f8723ad0b94eec005025f07a9a6 [1,2]. Locally computed SHA-256 of the complete readable export was:

63f170645e67131025e1d3b8d6e9002a8f3312741ecfce121b886f8d085bd68b

The archive declares 159 repository entries, of which 148 text payloads are embedded; eleven non-text payloads are inventoried but not included in this export. A byte-counted extraction script recomputed SHA-256 and canonical Git blob SHA-1 for **148/148** embedded text files; every pair matched the values supplied alongside its corresponding bytes. This establishes self-consistency of the text export, not independent authentication of all original repository files.

To add a limited external check, we fetched the pinned commit through the connected GitHub repository and compared the Git blob identifiers of three source files: witness_provenance_experiment.py, test_witness_provenance_experiment.py, and test_checkpoint_prefix_experiment.py. All three identifiers agreed with the reconstructed export's Git blob digests. This spot check is stronger than checking internally supplied digests alone but does not establish a complete fresh Git clone, an independently trusted timestamp, or all 159 entries.

4.2 Existing module reproduction

Five selected Python test modules from the pinned source were executed as one combined subset: checkpoint prefix (73 cases), independent historical witness (26), provenance (41), compromise/recovery v0.1 (53), and compromise/recovery v0.2 (87). Their tests use laboratory fixtures rather than production witnesses, networks, legal authorizations or hardware signing. The local environment used CPython 3.13.5, pytest 9.0.2, and a Linux analysis host. Bytecode and pytest cache creation were disabled. The execution command, working directory, and full output are included in the supplement.

An initial attempt to run the copied modules from the package root failed at import/collection because the implementation directory was not on the module import path. Its error log is retained. Rerunning the unchanged five test files from the correct snapshot/implementation/ working directory produced the reported result. The error is an execution-setup issue and is not counted as a failure of the five modules under their specified environment. No claim is made that all other repository tests pass.

4.3 Experimental comparators

The **storage-only comparator** asks whether the two declared witness artifacts refer to different storage object identities. If both objects are distinct, it returns CONDITIONAL. If they have the same storage identity, it returns DEPENDENT; if the two artifacts are duplicated or storage cannot be resolved, it returns UNKNOWN. This comparator is intentionally weak; it is a pedagogical baseline and **not** a competitive provenance-auditing product or a fair representation of contemporary state of the art.

The **bounded provenance gate** calls the unmodified Ario evaluate_provenance() function for each case, then applies a declared three-outcome policy. A COMMON_CONTROL_DETECTED, COMMON_SOURCE_DETECTED, or SAME_STORAGE_OBJECT finding yields DEPENDENT. A CONDITIONAL finding is permitted only if the engine simultaneously reports LAB_CONDITIONALLY_SEPARATE administrative paths, LAB_CONDITIONALLY_DISTINCT sources,

LAB_DISTINCT_OBJECTS storage, two LAB_CHAIN_PRESENT custody paths, and LAB_BEFORE_MUTATION capture order. Every other combination yields UNKNOWN. The gate is a new study-specific wrapper around the pinned implementation, not a claim that any deployed Ario system enforces it.

The two comparators are applied to exactly the same constructed ProvenanceCase input objects. The label and the comparator code were developed by the same research team and AI-assisted tooling, so circularity and design bias are explicit study limitations. No holdout data or blind human adjudication exists.

4.4 Synthetic cases

The sixteen cases are generated in the accompanying script by controlled edits to the pinned repository's own synthetic fixture constructor. Seven include explicit common-cause dependencies, seven include missing or inadmissible evidence, and two retain laboratory separation (one of which adds an unsupported self-certified claim that is not admitted as evidence). All identities, control roots, source objects, storage names and times are symbolic; none refers to an actual person or live system.

ID	Deliberate variation	Expected
S01	Distinct source and control paths	CONDITIONAL
S02	Shared administrator	DEPENDENT
S03	Shared source	DEPENDENT
S04	Shared source and administrator	DEPENDENT
S05	Two storage nodes alias one object	DEPENDENT
S06	Shared writable restore backup	DEPENDENT
S07	Shared higher-level control root	DEPENDENT
S08	Two named sources alias one source identity	DEPENDENT
S09	Control inventory is incomplete	UNKNOWN
S10	Source inventory is incomplete	UNKNOWN
S11	Both inventories are incomplete	UNKNOWN
S12	Provenance graph is candidate-supplied	UNKNOWN
S13	Writer edge missing with incomplete control inventory	UNKNOWN
S14	Custody transfer record missing	UNKNOWN
S15	Valid lab graph with an inadmissible self-certified extra claim	CONDITIONAL
S16	One witness artifact is presented twice	UNKNOWN

S15 tests a boundary rather than treating a self-certified statement as a valid extra witness: the engine rejects the extra claim as unsupported while leaving the original hypothetical lab graph's conditional properties unchanged. S16 tests whether two labels that point to one artifact are counted as two independent artifacts.

5. Results

5.1 RQ1: Source and unit-test reproduction

The archive extraction yielded 148 matching text payload hashes and Git blob digests. The chosen five-module test subset yielded **280 passed**, in approximately one second on this analysis host. The duration is included only as a local execution observation and is not a performance benchmark. Three selected file blob identities separately agreed with the pinned GitHub revision. Those results support byte correspondence, source identification for the checked files, and the exercised deterministic assertions. They do not establish that the tests exhaust the possible implementation behaviors or that historical events represented by symbolic test records occurred.

Reproduction check	Observation	Unresolved boundary
Readable source inventory	148/148 SHA-256 and Git blob SHA-1 pairs matched	Hashes and payloads were supplied together
GitHub file spot check	3/3 selected Git blob IDs matched pinned commit	Not a complete independent repository audit
Selected local regression subset	280/280 passing pytest cases	Not all Ario tests; no deployed-system evaluation
Independent field witnesses	Not exercised	Custodian independence not established

5.2 RQ2: Classification comparison

The baseline agreed with **4/16** declared scenario labels. It reported CONDITIONAL in fourteen cases, including twelve labeled DEPENDENT or UNKNOWN. Of the seven known-dependence cases, it detected only the same-storage-object case; the six other dependencies remained invisible to its storage-only decision rule. Of the seven unknown cases, it declined in one duplicated-artifact case and endorsed six without sufficient evidence.

The bounded gate agreed with **16/16** declared labels: two CONDITIONAL, seven DEPENDENT, and seven UNKNOWN. Its zero unsupported conditional endorsements in this tiny constructed sample should not be expressed as an empirical zero false-positive probability. The comparison demonstrates how the explicit presence of shared-source/control edges and admission of UNKNOWN can change verdicts *for these sixteen specific inputs*.

Declared scenario class	Count	Storage-only rule	Bounded provenance gate
CONDITIONAL	2	2 conditional	2 conditional
DEPENDENT	7	6 conditional; 1 dependent	7 dependent
UNKNOWN	7	6 conditional; 1 unknown	7 unknown
Exact agreement with constructed labels	**16**	**4/16**	**16/16**
Unsupported CONDITIONAL endorsements	**14 nonconditional inputs**	**12**	**0**

No statistical confidence interval, effect-size extrapolation, p-value, or population-level security score is claimed. The outcomes are deterministic observations on an intentionally selected fixture set; random sampling and independent verification of the oracle were absent. The result does not establish superiority over any established provenance system.

5.3 RQ3: Boundary between diagnostic result and authorization

The unmodified underlying provenance report still returns values such as NOT_ESTABLISHED for production custody authenticity and NOT_EVALUATED for human authorization and evidence admission. These values remain unchanged even when the wrapper emits CONDITIONAL for S01 or S15. The distinction matters: a lab report can satisfy its formal predicates without proving that any real custodian, independent data source, or legal authorizer exists. This study neither activates nor tests an operational signer. No participant identity, event truth, persistent AI selfhood, external witness independence, or safe deployment claim follows.

6. Interpretation

The comparison exposes two separable questions: **Are there two distinct storage records?** and **Are there two independently grounded evidence paths?** The first can be tested by storage-object identity alone. The second depends on source lineage, administrative control, backup pathways, timing, custody, completeness, and whether the provenance graph has an admissible origin. A system that silently equates these questions may take a positive storage check as an unjustified endorsement of evidence independence.

S02–S04 show that different disk objects can coexist with a single controlling principal or original information source. S06 and S07 show less direct common-cause paths through restore backups or control-root relationships. S09–S14 distinguish absence of an asserted dependency from evidence that no dependency exists. Most importantly, S12 shows

that even an internally coherent graph cannot establish its own independence when supplied by the candidate it is supposed to scrutinize.

The fact that the bounded wrapper agrees with every label is not a surprise. It was composed from the same categories that distinguish the synthetic scenarios. Its value is **mechanistic and inspectable**: it isolates precisely the assumptions that would be required to promote a conditional outcome. A meaningful subsequent evaluation would independently construct inputs, blind the comparator authors to the expected labels, include adversarial graph fabrication and hidden shared controls, and run a documented real custody protocol before estimating an operational error rate.

There is also a general lesson about the relationship of integrity and source authentication. Digest checks, Git object identities and append-only log checks are beneficial because they make local, reproducible byte-level propositions testable. But no number of matching representations can prove that a record is historically true or independent unless the trust in its creation and retention is separately justified. Provenance-aware decisions are useful partly because they can preserve an honest UNKNOWN where a naive checker would instead promote a favorable inference.

7. Validity, ethics, and reproducibility limitations

Selection bias and no blinding. Sixteen conditions were hand selected; their classes were specified by the same authoring process as the benchmark. Cases designed to contain features recognized by the engine are not independent adversarial challenges. The comparison is a feasibility demonstration, not a representative benchmark or independent proof of robustness.

Comparator choice. The baseline is deliberately impoverished, so the 4/16 versus 16/16 contrast cannot be generalized to competing commercial or academic provenance analyzers. The gate is a derived policy that combines existing engine outputs, rather than an independently verified system. It is not a novel cryptographic primitive and is not an end-to-end authorization mechanism.

Trust and completeness. Synthetic graph completeness is assigned by fixture labels, and provenance can be false in the external world. Live administrative inventories, clocks, signatures, independent physical hosts, malicious simultaneous modification, and independent third-party custody are absent. Missing a dependency may cause a positive lab classification without revealing the hidden truth.

Execution environment. All tests ran on a local Linux environment with CPython 3.13.5 and pytest 9.0.2. There was no OS diversity, stress test, concurrency challenge, native Windows verification, or full regression of the contemporary repository main branch. The pinned commit is an October 2026 state, not a claim about today's implementation or unresolved GitHub issues.

Transparency about workflow. The protocol was exploratory and AI-assisted, not prospectively registered. A preliminary packaging attempt produced module-import collection failures due to the wrong working directory; a corrected command subsequently succeeded. Both logs are supplied. Stating this prevents an incomplete or unsuccessful execution from being silently represented as a pass. The large private conversation export provided for historical background was not treated as a dataset of independently verified claims or redistributed in the supplement.

Reproducibility versus replication. A second investigator can reproduce local outputs from the same code and source file. Independent empirical replication of custody assumptions would require new data from actors and systems not controlled by this project. These distinct achievements must not be conflated [8].

8. Future experimental work

A defensible next study should use an independently authored, preregistered mutation matrix with a withheld oracle and operationally authenticated witness custody. Candidate trials should include unrecorded common administrators, cross-device restores, conflicting clocks, equal-length in-place historical edits, selective disclosure of graphs, compromise of external reference holders, and downstream client attempts to transform a diagnostic report into an authorization event. Evaluation should report missed detections, false rejections and UNKNOWN decisions separately, with documented reference provenance and a matched audit baseline. When possible, the complete artifact inventory and preexisting

external commitments should be verified by auditors who did not author the target implementation. None of those extensions is reported as completed in this paper.

9. Conclusion

In a reproducible sixteen-scenario synthetic comparison over a pinned Ario provenance experiment, a rule that equates two storage objects with conditional separation incorrectly endorsed twelve of fourteen nonconditional fixture cases. A stricter, explicitly three-outcome gate matched all sixteen declared fixture classes, while retaining the original report's non-authorizing limits. A separate local rerun reproduced 280 passing targeted tests; internal digest checks covered 148 text files, and three file-level GitHub blob identifiers matched the pinned commit. These are **bounded observations**. The central finding is not that the system is safe, that witnesses are truly independent, or that an AI has a persistent identity. It is that source, control, completeness and authority need separate evidence, and that a useful audit must sometimes decline to conclude more than its inputs permit.

Declarations

Author and competing interest. Sepehr Ghanbari created the Ario research project and is its sole human author; this relationship is an explicit potential source of investigator bias. The prototype and synthetic fixtures originate from that project. The author should confirm any additional competing interests before submission.

Funding. The October manuscript reported no external grant or institutional funding for the evaluated work. The author must reconfirm that statement before public submission.

Ethics and data protection. The reported experimental materials comprise public source code and fictional test entities only; no humans were recruited and no sensitive participant data were evaluated in this study. Private chat archives were used only to locate prior manuscript and project context and are not published as experimental data.

Availability of code and data. The pinned source is accessible at https://github.com/SepehrGhanbari/Ario/tree/f562e14cd8ca7f8723ad0b94eec005025f07a9a6 . Related repository archive: https://doi.org/10.5281/zenodo.23651704 . A self-contained supplementary package accompanies this draft, including the readable source snapshot, byte-counted extractor, test-run transcript, complete sixteen-case evaluation code, JSON/CSV scenario results, and SHA-256 manifest. These supplement artifacts are newly generated working files and should not be described as already deposited at an external DOI.

Artificial intelligence disclosure. GPT-6 (OpenAI, November 2026) materially assisted with scientific drafting, experimental code development, local testing and analysis. It is not a coauthor. The human author must verify all sources, outputs, contributions and conclusions before public posting. This draft makes no unsupported claim of independent human validation.

References

[1] Ghanbari, S. (2026). *Ario: open-source repository, pinned 25 October 2026 source snapshot*, commit f562e14cd8ca7f8723ad0b94eec005025f07a9a6. https://github.com/SepehrGhanbari/Ario/tree/f562e14cd8ca7f8723ad0b94eec005025f07a9a6 . Accessed 5 November 2026.

[2] Ghanbari, S. (2026). *Ario: Complete Verifiable GitHub Repository Snapshot and AI-Readable Text Edition*. Zenodo archival record. https://doi.org/10.5281/zenodo.23651704 . The exact source edition studied is separately SHA-256-identified in Section 4.

[3] Buneman, P., Khanna, S., & Tan, W.-C. (2001). Why and Where: A Characterization of Data Provenance. *International Conference on Database Theory*, 316–330. https://doi.org/10.1007/3-540-44503-X_20 .

[4] Green, T. J., Karvounarakis, G., & Tannen, V. (2007). Provenance Semirings. *Proceedings of PODS 2007*, 31–40. https://doi.org/10.1145/1265530.1265535 .

[5] Moreau, L., & Missier, P. (Eds.). (2013). *PROV-DM: The PROV Data Model*. W3C Recommendation. https://www.w3.org/TR/prov-dm/ .

[6] Laurie, B., Messeri, E., & Stradling, R. (2021). *Certificate Transparency Version 2.0*. RFC 9162. https://www.rfc-editor.org/rfc/rfc9162.html .

[7] National Institute of Standards and Technology. (2020, updated). *Security and Privacy Controls for Information Systems and Organizations*, NIST SP 800-53 Rev. 5, AU-9. https://doi.org/10.6028/NIST.SP.800-53r5 .

[8] National Academies of Sciences, Engineering, and Medicine. (2019). *Reproducibility and Replicability in Science*. National Academies Press. https://doi.org/10.17226/25303 .

[9] Manakul, P., Liusie, A., & Gales, M. (2023). SelfCheckGPT: Zero-Resource Black-Box Hallucination Detection for Generative Large Language Models. *EMNLP 2023*, 9004–9017. https://doi.org/10.18653/v1/2023.emnlp-main.557 .

Appendix A. Reproduction procedure and scope

The attached Ario_READ.txt is retained byte for byte. From the supplementary folder run:

```
python extract_snapshot.py
cd snapshot/implementation
PYTHONDONTWRITEBYTECODE=1 python -B -m pytest -p no:cacheprovider -q \
 tests/test_checkpoint_prefix_experiment.py \
 tests/test_independent_witness_experiment.py \
 tests/test_witness_provenance_experiment.py \
 tests/test_witness_compromise_recovery_experiment.py \
 tests/test_witness_compromise_recovery_experiment_v0_2.py
cd ../..
python reproduce_study.py
```

Requirements for the selected subset: CPython 3.13.5 and pytest 9.0.2 as used here. Some other parts of the repository may have additional dependencies. Running the code executes synthetic tests only and does not require accounts, credentials, or signing keys. Use a disposable environment. Validate SHA256SUMS.txt before relying on downloaded copies.

The supplementary directory contains scenario_results.csv, scenario_results.json, summary.json, pytest_280_reproduction.log, and pytest_attempt_wrong_working_directory.log. The initial failed import attempt is preserved to clarify execution assumptions, rather than erased from the research narrative.