AUDITING SOURCE PROVENANCE IN RETRIEVAL-AUGMENTED DATA SYNTHESIS

Anonymous authors
Paper under double-blind review

ABSTRACT

Retrieval-augmented data synthesis turns retrieved source passages into reusable examples, such as question–answer (QA) pairs for model development. These examples can retain source knowledge without retaining the links needed to audit its use. We introduce RelMark, a source-provenance method that encodes private patterns in local recommendations between alternatives verified equivalent for specified tasks. The pattern can travel as useful answer content despite changes in wording. Given completed QA and a source registry, an offline auditor recovers recommendation directions without the key, then tests their aggregate alignment with the private pattern. Auditing requires neither generation logs nor queries to the original service. Across 80 CLI source groups and 64 native API/SQL relations, RelMark recovers recommendations from 78.1–93.8% of native-reference groups across four generators, with at least 98.1% of recovered directions aligned with the owner key. All native owner–control contrasts pass correction within their registered test families; independently keyed sources retain recommendations without aligning with the owner pattern. Rewriting and shuffled audits demonstrate recovery without verbatim carrier text or generation-batch lineage. A source-detectable KGW adaptation attenuates below its registered detection threshold in derived QA. These results establish task-valid relational patterns as an effective carrier for auditing source provenance directly from synthetic data.

1 INTRODUCTION

Retrieval-augmented generation (RAG) grounds language-model outputs in retrieved documents (Lewis et al., 2020). Beyond answering individual queries, it can also construct synthetic datasets for model development (Bhushan et al., 2025; Tian et al., 2026). In retrieval-augmented data synthesis, a constructor retrieves source passages, asks a large language model (LLM) to generate grounded examples such as question–answer (QA) pairs, and collects them into a reusable dataset. Software documentation, for example, can supply facts and usage guidance for technical QA. The result is a new data resource built from source knowledge, whose examples may select and restate that content.

This transformation creates a provenance gap. Generated examples may preserve facts and recommendations without preserving their mapping to the documents that supplied them. Once the dataset is shared, a source owner may inspect its examples while lacking access to the original index, generation logs, or service. Useful source content remains available, but its contribution becomes difficult to verify. We ask: *can a source owner audit the use of a particular source copy from a completed synthetic dataset, without reconstructing the process that produced it?*

This setting changes what an audit can observe. Live RAG methods can choose queries to elicit source evidence (Jovanović et al., 2025; Yang et al., 2026); document contracts can prescribe content in designated outputs (Zhang et al., 2026). Here the QA collection already exists: the auditor cannot solicit new responses or choose their format. Retrieval may expose only part of a document, while synthesis rewrites and reorganizes what it selects. A pattern detectable in a source can therefore disappear in the derived data. The challenge is to encode evidence in content useful to synthesis, without depending on preservation of a particular sentence or output slot.

Our insight is that a valid recommendation can serve as both useful source content and a private provenance signal. Where a task admits multiple verified alternatives, a source copy can recommend

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

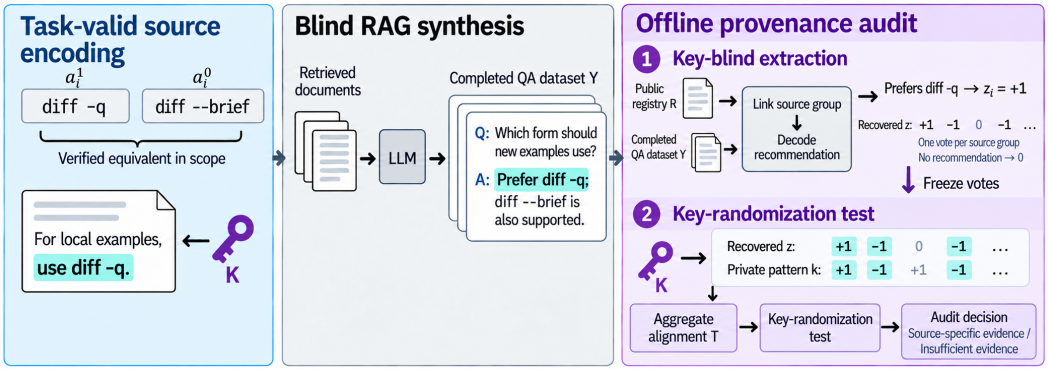

Figure 1: Overview of task-valid relational provenance auditing. The publisher uses a private key to choose local recommendations between alternatives verified equivalent within a specified task scope. Ordinary RAG synthesis may retain these recommendations as answer content despite changes in wording. Given the completed QA dataset and a source registry, the auditor extracts recommendation directions without the key, then tests their aggregate alignment with the private pattern. Auditing requires neither generation logs nor live queries. The example and direction sequences are schematic.

one as a local convention while keeping both valid. Consider `diff -q` and `diff --brief`, verified to behave equivalently within a specified execution scope. A publisher can recommend one for new examples without changing that tested behavior. A synthesized answer can preserve the preference while paraphrasing its wording. This observation leads to **RelMark**: task-valid relational encoding uses a private pattern to choose recommendation directions across registered source groups. Execution determines which alternatives are admissible; useful recommendation content supplies the opportunity for transfer.

RelMark couples this encoding with offline, key-blind auditing (Figure 1). Given completed QA and a public source registry, the auditor first links recommendations to source groups and decodes their directions without consulting the key. It then tests aggregate alignment with the owner's pattern. This order prevents the expected fingerprint from guiding evidence interpretation. Each source group contributes at most one vote, so repeated examples cannot multiply the evidence. The external generator receives neither the key nor a propagation instruction. In evaluation, matched source copies with independent keys test whether recovered directions follow the released source rather than a generator's existing preferences.

We evaluate RelMark with four generators on 80 CLI source groups and 64 native API/SQL relations. The native study recovers owner evidence in 50–60 of 64 registered groups, with all owner–control contrasts significant within their registered test families. Independently keyed copies retain recommendations, but these do not systematically align with the owner pattern. Rewriting and shuffled-data audits demonstrate recovery after rewording and removal of generation-batch boundaries. A source-side KGW adaptation remains detectable in retrieved contexts but falls below its registered threshold in derived QA, highlighting the transfer opportunity for a content-bearing carrier in offline dataset auditing.

Our contributions are threefold:

- **Task-valid relational encoding.** We convert local conventions between execution-verified alternatives into private source patterns while preserving tested task behavior.
- **Offline provenance auditing.** We recover source-linked directions from fixed QA before key testing, and aggregate evidence by source group without generation logs or queries to the original service.
- **Controlled evidence of source-specific transfer.** We establish source-key specificity across four generators and three reference categories, and demonstrate evidence recovery after rewriting and removal of generation-batch boundaries.

2 PROBLEM AND THREAT MODEL

Let $G = \{1, \ldots, N\}$ index registered, semantically distinct source groups. Group i contains alternatives a_i^0 and a_i^1 that satisfy the same frozen task specification. A private key assigns a bit $b_i \in \{0, 1\}$ to each group, and the owner releases a source corpus X^K that recommends $a_i^{b_i}$. A pipeline outside the owner's control $\mathcal{S}$ retrieves from X^K and generates a candidate dataset $Y = \mathcal{S}(X^K)$. We study the transition $K \to X^K \to Y$, not subsequent student-model training.

Auditor access. The auditor has the source registry, executable specifications, private key, and candidate derived dataset. It does not receive the downstream index, queries, generation prompts, logs, model weights, or document-to-example mapping. The operator may choose retrieval and generation models and may mix marked with unmarked content. The main experiment assumes no informed fingerprint removal; we evaluate such removal separately rather than calling it impossible.

Desired evidence. The output is a calibrated aggregate test of whether Y contains key-consistent recommendation evidence linked to the registered sources, rather than complete per-example lineage reconstruction.

Table 1: Source encoding and audit channels. Our target is a fixed derived dataset, with no follow-up queries or prescribed output slots. The KGW row describes our fixed-QA adaptation, evaluated with its native token detector.

Method / setting	Source carrier	Audit observation	Downstream elicitation
WARD (original)	Rewritten token watermark	Live RAG responses	Auditor chooses queries
Source-side KGW adaptation	Rewritten token watermark	Fixed QA text	No audit-time queries
SemTrace	Selected factual propositions	Completed reviews	Embedded contract specifies review slots
Ours	Keyed recommendation between valid options	Fixed QA and source registry	No private key, prescribed slots, or propagation instruction

Table 1 separates access assumptions from empirical outcomes. Source-side semantic signatures also precede our work: KMW combines knowledge completion and multi-bit watermarking, while SemTrace uses factual propositions and a review content contract (Yang et al., 2026; Zhang et al., 2026). Our question is whether ordinary QA synthesis selects a valid recommendation without a prescribed output slot, and whether its fixed output supports key-specific evidence after key-blind source linkage.

3 RELMARK

RelMark consists of source-side encoding and a two-stage offline audit. The publisher embeds keyed recommendations between verified alternatives. Given a completed QA dataset, the auditor first recovers source-linked recommendation directions without the key and then tests their aggregate alignment with the private pattern.

3.1 TASK-VALID RELATIONAL ENCODING

The publisher needs encoding freedom that preserves which alternatives satisfy the documented task. For each source group i, the registry records alternatives a_i^0, a_i^1, their task specification and execution scope. Paired fixtures qualify them by comparing relevant outcomes within that scope: exit status and filesystem effects for CLI tasks, function outputs for API tasks, or result values and types for SQL tasks. Qualification is scoped, not a claim that the alternatives behave identically for every possible input.

Because both choices are admissible, the released copy can recommend either for local examples. A private bit b_i selects the direction

$$a_i^{b_i} \succ_{\text{local}} a_i^{1-b_i}, \tag{1}$$

where $\succ_{\text{local}}$ denotes a local convention, not superior task performance. A source may thus recommend `diff -q` while documenting `diff --brief` as supported in the verified scope. Across groups, these directions form the private pattern. The publisher edits its source copy before collection; the method requires prior marking.

This construction separates operational validity from documentary preference. Execution checks determine the allowable choices, and the private pattern determines which choice the copy recommends. The decoding target is this relation, not the sentence that expresses it. Execution validation defines the admissible encoding space, while the recommendation relation provides the content that can be carried into synthesized answers.

3.2 KEY-BLIND EVIDENCE RECOVERY

An external constructor retrieves passages and generates ordinary grounded QA, receiving neither the key nor an instruction to propagate the fingerprint. Its completed dataset may preserve recommendations without their wording or document-to-example mapping. RelMark recovers evidence using only that dataset Y and the public registry $\mathcal{R}$:

$$\mathbf{z} = \mathcal{D}(Y, \mathcal{R}), \qquad z_i \in \{-1, 0, +1\}. \tag{2}$$

Recovery separates two questions: which source group does an example discuss, and which alternative does it recommend? A key-blind linker resolves the group from identifiers and context. A relation decoder interprets explicit recommendations relative to the registry's fixed alternative ordering: $+1$ recommends a_i^1 and -1 recommends a_i^0. Mentioning an option, listing both, or describing their common function supplies no signed vote. Missing links, absent recommendations and conflicting directions produce zero. All examples linked to a group consolidate into one output.

We use a conservative lexical extractor for CLI sources and a cross-model LLM extractor for native API/SQL references. The latter must quote supporting QA text verbatim. Neither implementation receives expected bits. Links and directions are frozen before loading the key, so the expected pattern cannot resolve ambiguity in either source assignment or recommendation direction (Appendices G.2 and D).

3.3 SOURCE-GROUP PROVENANCE TESTING

A retained recommendation alone is not diagnostic: a generator may already prefer that alternative. RelMark tests the aggregate pattern. With signed key bits $k_i = 2b_i - 1$, its full-catalog statistic is

$$T(Y, K) = \frac{1}{N} \sum_{i=1}^{N} k_i z_i. \tag{3}$$

Each aligned direction contributes $+1/N$, each opposing direction $-1/N$, and each abstention zero. All registered groups remain in the denominator; repeated QA can aid recovery but cannot create extra statistical units.

The statistic combines coverage with directional fidelity. Let C be the fraction of groups with a nonzero vote and F the fraction of those votes aligned with the key. Algebraically,

$$T = C(2F - 1), \tag{4}$$

with $T = 0$ when no vote is recovered. Broad coverage is insufficient if directions do not align; high fidelity on a small subset is distinguished from broad recovery. We report coverage alongside the signed score.

For a single candidate, we hold $\mathbf{z}$ fixed and compare $T(Y, K)$ with scores under independent uniform random-sign assignments $\widetilde{K}$:

$$p_K = \Pr_{\widetilde{K}}\left[T(Y, \widetilde{K}) \geq T(Y, K) \mid \mathbf{z}\right]. \tag{5}$$

This conditional bit null can be enumerated exactly or sampled; its assumption and implementation are specified in the protocol. A small value supplies key-specific evidence; otherwise the audit returns insufficient evidence. The audit requires only $(Y, \mathcal{R}, K)$, not a separately generated control.

Our controlled evaluation additionally compares $\Delta_c = T(Y^{\text{owner}}, K) - T(Y^c, K)$, where c is an independently keyed or neutral source condition. Wrong keys test specificity of fixed evidence; independently keyed copies change source recommendations while matching the synthesis setup, testing whether directions follow the released source. These complementary controls validate the mechanism; the deployed audit still requires only the candidate, registry and key.

4 Experimental Design

4.1 Sources and controlled synthesis

We generate QA from matched owner-keyed, independently keyed, and neutral source copies to isolate the effect of source recommendations on the derived data.

We first oversampled 120 candidates from public CLI help and validated them in isolated executable fixtures without reading downstream model outputs. Eighty passed across five source families: GNU diff, dpkg, GNU binutils, patch, and Wget. Appendix B.1 gives family counts and exclusion reasons. The cohort and protocol were fixed before formal generation. The native-reference study uses 64 new relations, matched owner/independent/neutral copies, and the same retrieved contexts across its four generators. Sources retain both options as valid; only local recommendation directions change between keyed copies.

Common generators and cohort-specific implementations. DeepSeek-v4-pro, Kimi-k3, GPT-5.6-Sol and Claude-Opus-5 each cover all three source categories. Owner, independent-key and neutral copies supply common contrasts; CLI also includes original notes. Retrieval is BM25 top-three for CLI and top-one for native references; extraction is lexical versus cross-model. Within-cohort interventions are controlled, while cross-cohort rates do not isolate document type. Endpoint selection and inference status appear in Appendix D.2.

We select CLI, API, and SQL references because their alternative choices can be validated through command behavior, function outputs, and query results, respectively.

4.2 Replication and quality assessment

Beyond the 80-group CLI cohort, a disjoint 96-group replication uses controlled executable sources. A prospectively frozen cross-model rewrite tests transfer through a non-extractive transformation. In every study, inference uses registered source groups rather than individual QA examples.

Quality assessment combines model-assisted evaluation with author verification. The authors manually reviewed the experimental outputs and assessment results against the corresponding source evidence. Records distinguish model-generated assessments from final human-reviewed judgments. A 300-item audit-validation pack contains 240 representative QA and 60 challenge cases; its rubric and final labels appear in Appendix G.2.

4.3 Comparisons and transformations

Matched internal diagnostics use the same 48 development groups, retrieval, and blind QA synthesis within that comparison. Retrieval is top-one within this diagnostic. The *invisible-surface* condition inserts a task-valid form marker at the same character budget; the *fictitious-knowledge* condition inserts a traceable but invalid claim. They test how carrier content affects propagation.

Our external diagnostic uses the official KGW token-watermark decoder (Kirchenbauer et al., 2023). On the new native-reference cohort, Qwen3.5-9B rewrites every source into clean, owner-key and independent-key copies with a development-fixed $\delta = 2$ paraphrase recipe. All copies enter the same synthesis pipeline regardless of source quality. Native source, actual-context and fixed-QA scoring are separate from a common anonymous fidelity rubric; we do not compare its token statistic

numerically to semantic T. Implementation, detection budgets and source-fidelity protocols appear in Appendix F.

We study passive source ingestion and vary content selection and editing. Two additional pipeline variants keep Formal80 sources, keys, retrieval, and the frozen auditor fixed: a non-preference functional-facts QA prompt, and a generic summarization stage followed by the original QA prompt. Each variant generates three QA for all 80 owner and 80 independent-key groups. Prompts and analysis were locked before their new outputs, but both tests are exploratory.

Finally, fixed-output attacks delete decoded relation rows or matched random rows, and re-evaluate T. An independent fact judge measures recovery of the original registered facts before and after deletion. Pool mixing samples owner groups into independent-key, neutral, or original fixed QA pools at six proportions with 5,000 resamples per non-endpoint. This last analysis is conditional on generated QA; it is a power boundary, not a new source-side partial-use experiment.

Inference. Formal80 uses 100,000 random-sign assignments with the $(c+1)/(B+1)$ correction and four owner–control tests corrected by Holm at .05. The native study uses exact weighted sign-flip enumeration, with the original GPT/Claude and DeepSeek/Kimi coverage tests kept as separate four-test Holm families at .01. Additional variants and shuffled auditing retain their separate analysis status. These tests condition on frozen evidence under an independent-sign bit null; they do not enumerate the cryptographic secret distribution. A secondary single-candidate check demonstrates the deployment rule on the existing native outputs (Appendix D.6).

5 Results

5.1 Source-specific provenance recovery

Across four generators, owner-encoded sources yield positive owner-key alignment in all three reference categories (Figure 2; Table 2). Independently keyed sources still yield many recommendations, but their directions align weakly with the owner pattern. The contrast identifies source-specific transfer rather than mere preference recovery.

Source specificity in native references. The native cohort contains 32 API relations from PyTorch, Pillow, NetworkX and Arrow, and 32 SQL relations from SQLite and DuckDB. Native facts come from reference text, while 505 paired execution fixtures qualify encoding choices. Four generators produce 4,608 QA across six source conditions. Owner evidence is recovered in 50/64 groups for DeepSeek, 52/64 for Kimi, 60/64 for GPT and 54/64 for Claude. Independent-key outputs retain 53–58 directions, yet their owner-key scores range from $-.047$ to $.047$. All native owner–control contrasts pass Holm correction within the two registered four-test families (Appendix D.4). Owner and twin retrieval both reach 61/64 target documents, strengthening the control against generator preference; neutral reaches 31/64.

Single-candidate auditing. RelMark also supports auditing an individual completed dataset. All four owner-derived native datasets pass the key-alignment test, while none of the eight independently keyed or neutral controls does. The auditor uses only that candidate's QA, the registry and the owner key; the twelve secondary checks use a separate Holm correction at .01 (Appendix D.6).

Recovery without batch boundaries. For the original GPT/Claude outputs, flattening and shuffling QA after removing batch boundaries retains owner coverage of 57/64 and 54/64. All four contrasts remain significant in their separate Holm family. Source linkage uses only QA and the key-free registry, abstaining on unresolved referents (Appendix D.5). Thus the observed audit does not require the original document-to-generation grouping.

5.2 Task validity and QA quality

All 64 owner, independent and neutral native sources pass the common source-fidelity rubric, complementing 505 paired execution fixtures. With a common direct-read GPT answer/judge auxiliary, native-fact scores are 113/128, 112/128 and 114/128, respectively. The encoding preserves the tested operational content while introducing a recoverable local convention.

Table 2: Source-specific evidence. Coverage is decoded owner groups over all registered groups; scores always use the owner key. API/SQL rows are descriptive strata. Independent sources retain 26–30 votes per native stratum and 31–44 CLI votes. Cohort-specific inference and complete counts are in Appendices D.2 and D.4.

Sources	Generator	Owner coverage	T_o	T_{ind}	T_n	ΔT
CLI	DeepSeek-v4-pro	28/80	0.350	-0.013	0.000	0.363
CLI	Kimi-k3	31/80	0.363	0.013	0.000	0.350
CLI	GPT-5.6-Sol	48/80	0.600	-0.025	0.000	0.625
CLI	Claude-Opus-5	45/80	0.538	0.000	0.000	0.538
API	DeepSeek-v4-pro	23/32	0.719	0.000	0.031	0.719
API	Kimi-k3	25/32	0.719	0.063	-0.094	0.656
API	GPT-5.6-Sol	30/32	0.938	0.125	0.000	0.813
API	Claude-Opus-5	26/32	0.813	0.156	-0.063	0.656
SQL	DeepSeek-v4-pro	27/32	0.844	-0.094	0.000	0.938
SQL	Kimi-k3	27/32	0.844	-0.094	0.031	0.938
SQL	GPT-5.6-Sol	30/32	0.938	-0.063	0.000	1.000
SQL	Claude-Opus-5	28/32	0.875	-0.063	0.031	0.938

Table 3: Native-reference QA support (model assessment, author-reviewed). Each cell schedules 192 QA; fractions use available judgments. Terminal grading failures and all six source conditions are reported in Appendix G.5.

Generator	Owner	Independent	Neutral
DeepSeek-v4-pro	176/183	179/180	191/192
Kimi-k3	183/192	184/192	183/192
GPT-5.6-Sol	180/183	177/180	191/192
Claude-Opus-5	173/192	158/186	159/192

Table 3 reports support of the derived QA under the actual received contexts. These are model-generated assessments verified by the authors; missing judgments retain their available denominators. The broader human-adjudicated audit pack finds 287/300 answers supported and all 300 questions useful. Its representative stratum has 100% direction precision at 9.17% direction coverage, showing a conservative operating point for the CLI decoder (Appendix G.2).

5.3 Relations survive rewriting

A disjoint 96-group replication preserves a large owner–twin effect: the all-group signed scores are .875 and .021 ($\Delta = .854$, $p = 4.00 \times 10^{-11}$). A separately frozen cross-model rewrite retains .927 versus .042 ($\Delta = .885$, $p = 9.17 \times 10^{-13}$). The pattern can therefore survive this non-extractive transformation. Full denominator accounting and alternative match-rate summaries appear in Appendix E.1.

Matched mechanism controls distinguish useful semantic content from surface marking. On 48 development groups, recommendation carriers yield 45 owner and 46 twin votes, whereas an invisible form marker yields six and two with zero paired effect. Fictitious claims also propagate (15 and 21 votes), but fail task validity by construction. Execution qualification supplies a valid encoding channel, and useful recommendation content is recovered substantially more often than the matched surface marker. Complete mechanism counts appear in Appendix E.2.

Summary and retrieval transformations. Table 4 shows that the synthesis objective determines how much evidence survives. A generic summary followed by QA preserves 12/80 owner and 13/80 twin votes, all aligned with their native source directions. The owner–twin effect remains positive ($\Delta = .1625$, adjusted $p = .01860$). No registered carrier span survives verbatim in the summaries: the relation can be retained without copying the original sentence. A lexical audit of successful items confirms varying overlap, with no complete carrier matches among the 15 decoded summary-derived owner items (Appendix E).

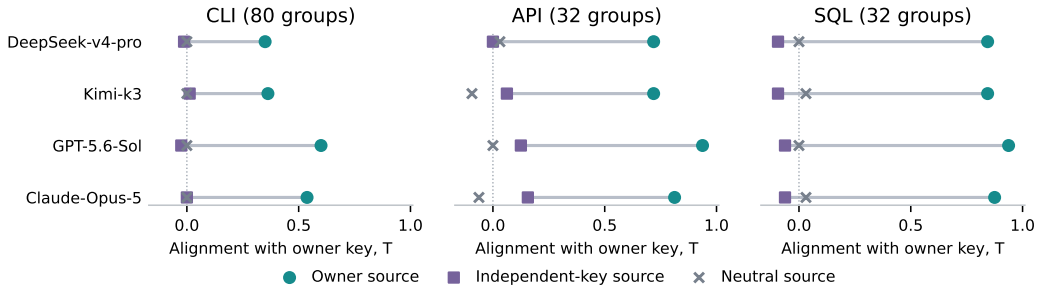

Figure 2: Recovered directions follow the source pattern. Owner sources yield positive alignment with the owner key; independent-key sources retain recommendations without following that key. Scores use all registered groups. Different cohorts and extractors make cross-category rates descriptive.

Table 4: DeepSeek pipeline variants on the same 80 groups and keys. Each additional variant completes 160 requests and 480 QA. The original row is the locked Formal80 cell; variants are exploratory. Top-one uses ordinary-fact queries and a different retrieval depth, so its absolute rate is not a single-factor estimate of either change.

Pipeline path	Owner votes	Twin votes	Owner T	Owner − twin
Original QA	28	31	.3500	.3625
Summary → QA	12	13	.1500	.1625
Functional facts QA	0	0	.0000	.0000
Ordinary-fact top-one	18	19	.2250	.1875

The audit therefore recovers recommendation directions even when the original carrier sentence is not copied. The functional-facts condition examines the complementary content-selection boundary, discussed below.

An ordinary-fact top-one pipeline also remains key-specific: 18 owner and 19 twin votes yield $\Delta = .1875$ (adjusted $p = .01860$). Sampled QA support is 59/60 and 58/60. This variant changes both query content and retrieval depth, so it establishes a further working configuration rather than isolating either factor. The separate three-test exploratory family and full prompts appear in Appendices C and E.

5.4 Token-watermark transfer

The relational audit remains positive after QA synthesis. Does a source-side text watermark survive the same transition? We apply the official KGW decoder to Qwen3.5-9B rewrites of the same 64 neutral references, with a development-fixed $\delta = 2$ recipe. Owner-key and independent-key sources enter the same top-one synthesis pipeline and are evaluated with their native detector.

Both rewritten source pools are strongly detectable ($z = 14.07/15.34$), as are the actual retrieved contexts ($z = 9.67/8.74$). After synthesis, four-generator owner QA scores fall to $z = -0.01$–1.28, and native twin QA scores to -1.47–1.51. None reaches one-sided $\alpha = .01$. Table 5 includes the common clean and cross-key controls.

Source fidelity could contribute to this loss: strict full-copy passes are 44/64 for KGW owner and 37/64 for its twin. We therefore separately inspect the QA produced from complete, fidelity-passing received sources. This post-hoc rule uses source judgments and context identity, without selecting QA or detector outcomes. Detection still attenuates (Appendix F.1, Table 20), showing that erroneous source copies alone are insufficient to explain the result.

The contrast supports the content-bearing design: a synthesized answer can restate a recommendation while replacing the text on which token bias depends. Relational evidence remains auditable after synthesis, whereas the evaluated token-watermark signal attenuates below its registered detection threshold. Detector controls, fidelity subsets and residual answer-only bias appear in Appendix F.1.

Table 5: Native KGW scores on the same 64 query groups across four generators. Clean rewrites use the owner key; independent KGW is scored under its native and the owner key. All whole-QA pools fall below the registered one-sided $\alpha = .01$ threshold. Scoring resets at QA boundaries and deduplicates repeated watermark contexts.

QA generator	Clean z	Owner KGW z	Independent: native z	Independent: owner z
DeepSeek-v4-pro	-0.34	-0.01	1.51	0.77
Kimi-k3	1.35	0.71	-1.47	0.28
GPT-5.6-Sol	0.95	0.66	-0.79	-0.38
Claude-Opus-5	-1.72	1.28	-0.30	2.00

5.5 Content selection and targeted editing

Functional-facts synthesis suppresses all votes under the frozen detector while sampled QA remain largely supported (56/60 owner and 54/60 independent). A broader key-blind check finds one indirect-preference item among 120 QA. This locates the content-selection boundary: excluding recommendations suppresses the registered signal while retaining operational content.

Dataset mixing provides a complementary partial-exposure result. At 25% owner source groups, neutral or original backgrounds yield 77.3–85.6% conditional detection power at $\alpha = .01$, compared with .22–.52% against independently keyed content. At 50% owner groups, the independently keyed background yields 31.9–46.5% power. These fixed-pool resamples preserve the source-group unit and use 5,000 draws per non-endpoint. The background matters because independently keyed material contributes valid recommendations in directions unrelated to the owner pattern, whereas neutral content primarily reduces coverage (Appendix H).

Targeted whole-example deletion removes the signal after deleting 36/240 DeepSeek QA (15%) or 45/240 Kimi QA (18.75%); equal-budget random deletion leaves $T = .325$ in both generators (Figure 3). The deleted examples are largely factually supported. An independent judge finds that this targeted attack retains 81.48% and 74.56% of previously recoverable ordinary facts, compared with 86.24% and 91.23% under random deletion. Thus, removing the marked examples also removes useful source information.

Editing the relation is cheaper than deleting the example. A separate QA-only attack reduces explicit-preference items from 34 to one while preserving operational facts in 29/32 native-reference collections under a comparative model assessment reviewed by the authors. Neutral-source construction likewise preserves the base fact inventory without yielding signed votes. Whole-example deletion and local editing have different information costs: the former removes useful QA, while the latter can preserve facts while suppressing recommendation evidence (Appendix H).

6 Limitations

RelMark applies to sources that admit verifiable local choices. Our evaluation covers CLI, API, and SQL documentation and examines transfer under rewriting, content selection, and editing. Deliberate removal of recommendation content can suppress the recovered signal. Extending the encoding space and improving recovery from implicit recommendations are promising directions.

7 Conclusion

RelMark audits source provenance directly from completed synthetic QA. It embeds private patterns in verified local conventions, recovers recommendations before consulting the key, and tests aggregate source-group alignment. Four-generator studies across CLI, API and SQL establish source-specific transfer; rewriting and shuffled audits recover evidence after wording and batch boundaries change. The single-candidate results demonstrate the intended deployment: an owner can test a fixed collection without access to counterfactual outputs. Key-blind recovery and one-vote-per-source aggregation separate evidence interpretation from pattern testing. RelMark enables source owners to audit completed synthetic datasets without reconstructing the original generation process.

AI USE STATEMENT

The methodology was designed by the authors. LLMs were used for the experimental generation and evaluation described in the paper, analysis-code implementation and review, and language editing. Figure 1 was edited with GPT Image. The authors reviewed the outputs and take responsibility for the final work.

ETHICS STATEMENT

This work studies technical evidence for source exposure in derived datasets. Such evidence is not by itself a legal determination of copyright infringement. The experiments use public technical materials and task-valid alternatives under frozen versions. We report removal and spoofing boundaries to avoid presenting the audit as conclusive proof of ownership or misuse.

REPRODUCIBILITY STATEMENT

The supplementary material records cohort manifests, configuration hashes, generation settings, frozen analysis rules, raw model-output receipts, exclusion reasons, and scripts for the reported statistics and figures. Private keys and credentials are excluded; public reproducibility keys and synthetic examples are provided where they do not invalidate the audit protocol.

REFERENCES

Kushagra Bhushan, Yatin Nandwani, Dinesh Khandelwal, Sonam Gupta, Gaurav Pandey, Dinesh Raghu, and Sachindra Joshi. Systematic knowledge injection into large language models via diverse augmentation for domain-specific RAG. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pp. 5937–5958, 2025.

Wassim Bouaziz, Nicolas Usunier, and El-Mahdi El-Mhamdi. Data taggants: Dataset ownership verification via harmless targeted data poisoning. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/96b8167534ef3cc30c230bbeb55a524d-Abstract-Conference.html.

Tianyu Chen, Jian Lou, and Wenjie Wang. Safeguarding multimodal knowledge copyright in the RAG-as-a-service environment. In *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/d7a2222b8d41014e060cfeb0995501d0-Abstract-Conference.html.

Xinyue Cui, Johnny Wei, Swabha Swayamdipta, and Robin Jia. Robust data watermarking in language models by injecting fictitious knowledge. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 14292–14306, 2025. doi: 10.18653/v1/2025.findings-acl.736. URL https://aclanthology.org/2025.findings-acl.736/.

Pierre Fernandez, Tom Sander, Hady Elsahar, Hongyan Chang, Tomáš Souček, Valeriu Lacatusu, Tuan Tran, Sylvestre-Alvise Rebuffi, and Alexandre Mourachko. How good is post-hoc watermarking with language model rephrasing? *arXiv preprint arXiv:2512.16904*, 2025.

Juan José González Torres, Mihai Bogdan Bîndilă, Sebastiaan Hofstee, Daniel Szondy, Quang-Hung Nguyen, Shenghui Wang, and Gwenn Englebienne. Automated question-answer generation for evaluating RAG-based chatbots. In *Proceedings of the First Workshop on Patient-Oriented Language Processing (CL4Health) @ LREC-COLING 2024*, pp. 204–214, 2024. URL https://aclanthology.org/2024.cl4health-1.25/.

Junfeng Guo, Yiming Li, Ruibo Chen, Yihan Wu, Chenxi Liu, Yanshuo Chen, and Heng Huang. Towards copyright protection for knowledge bases of retrieval-augmented language models via reasoning. *arXiv preprint arXiv:2502.10440*, 2025. URL https://arxiv.org/abs/2502.10440.

Abe Hou, Jingyu Zhang, Tianxing He, Yichen Wang, Yung-Sung Chuang, Hongwei Wang, Lingfeng Shen, Benjamin Van Durme, Daniel Khashabi, and Yulia Tsvetkov. SemStamp: A semantic

watermark with paraphrastic robustness for text generation. In *Proceedings of NAACL-HLT*, pp. 4067–4082, 2024.

Nikola Jovanović, Robin Staab, Maximilian Baader, and Martin Vechev. WARD: Provable RAG dataset inference via LLM watermarks. In *International Conference on Learning Representations*, 2025.

John Kirchenbauer, Jonas Geiping, Yuxin Wen, Jonathan Katz, Ian Miers, and Tom Goldstein. A watermark for large language models. In *Proceedings of the 40th International Conference on Machine Learning*, pp. 17061–17084, 2023.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Kuttler, Mike Lewis, Wen-tau Yih, Tim Rocktaschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Advances in Neural Information Processing Systems*, volume 33, pp. 9459–9474, 2020.

Yepeng Liu, Xuandong Zhao, Dawn Song, and Yuheng Bu. Dataset protection via watermarked canaries in retrieval-augmented LLMs. *arXiv preprint arXiv:2502.10673*, 2025.

Xinyang Lu, Jingtan Wang, Zitong Zhao, Zhongxiang Dai, Chuan-Sheng Foo, See-Kiong Ng, and Bryan Kian Hsiang Low. WASA: WAtermark-based source attribution for large language model-generated data. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 23791–23824, 2025. doi: 10.18653/v1/2025.findings-acl.1219. URL https://aclanthology.org/2025.findings-acl.1219/.

Peizhuo Lv, Mengjie Sun, Hao Wang, Xiaofeng Wang, Shengzhi Zhang, Yuxuan Chen, Kai Chen, and Limin Sun. RAG-WM: An efficient black-box watermarking approach for retrieval-augmented generation of large language models. *arXiv preprint arXiv:2501.05249*, 2025.

Pratyush Maini, Mohammad Yaghini, and Nicolas Papernot. Dataset inference: Ownership resolution in machine learning. In *International Conference on Learning Representations*, 2021.

Saksham Rastogi and Danish Pruthi. Revisiting the robustness of watermarking to paraphrasing attacks. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 18100–18110, 2024. doi: 10.18653/v1/2024.emnlp-main.1005. URL https://aclanthology.org/2024.emnlp-main.1005/.

Jon Saad-Falcon, Omar Khattab, Christopher Potts, and Matei Zaharia. ARES: An automated evaluation framework for retrieval-augmented generation systems. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 338–354, 2024. doi: 10.18653/v1/2024.naacl-long.20. URL https://aclanthology.org/2024.naacl-long.20/.

Alexandre Sablayrolles, Matthijs Douze, Cordelia Schmid, and Hervé Jégou. Radioactive data: Tracing through training. In *Proceedings of the 37th International Conference on Machine Learning*, pp. 8326–8335, 2020.

Chris Xing Tian, Weihao Xie, Zhen Chen, Hui Liu, Zhengyuan Yi, Haoliang Li, Shiqi Wang, and Siwei Ma. Domain-specific data generation framework for RAG adaptation. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 19236–19250, 2026.

Alexandr Goultiaev Tolstokorov, Kyriakos Mouratidis, Javad Dogani, and Nikolaos Laoutaris. Rent-a-RAG: Embedding-space watermarks for auditing third-party RAG. *arXiv preprint arXiv:2609.03749*, 2026.

Ke Yang, Shuguang Yuan, Jing Yu, and Chi Chen. Knowledge-infused multi-bit watermarking for RAG knowledge bases. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 21195–21218, 2026. doi: 10.18653/v1/2026.findings-acl.1066. URL https://aclanthology.org/2026.findings-acl.1066/.

Zhirui Zeng, Jiamou Liu, Meng-Fen Chiang, Jialing He, and Zijian Zhang. S-RAG: A novel audit framework for detecting unauthorized use of personal data in RAG systems. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics*, pp. 10375–10385, 2025. doi: 10.18653/v1/2025.acl-long.512. URL https://aclanthology.org/2025.acl-long.512/.

Junyan Zhang, Yudong Zeng, Yongwei Huang, Zuhao Ouyang, Hong Chen, and Xuming Hu. SemTrace: Source-grounded semantic signatures for tracing LLM exposure to protected documents. *arXiv preprint arXiv:2608.29575*, 2026.

A Additional Related Work

Trained-model provenance. Dataset inference, radioactive data, taggants, fictitious knowledge, and WASA test signals in trained models (Maini et al., 2021; Sablayrolles et al., 2020; Bouaziz et al., 2025; Cui et al., 2025; Lu et al., 2025). Their observation is a trained model or its responses; RelMark observes a completed QA dataset, before any subsequent training.

Generated-text watermarks. Token and semantic watermarks encode signals in generated text and face rewriting and quality–detectability tradeoffs (Kirchenbauer et al., 2023; Hou et al., 2024; Rastogi & Pruthi, 2024; Fernandez et al., 2025). Our source-side KGW comparison tests whether a source-detectable token signal survives ordinary synthesis; the detector and its token statistic remain native to that method.

RAG-service and document-contract audits. WARD, RAG-WM, canaries, KMW, and Rent-a-RAG probe operating RAG services (Jovanović et al., 2025; Lv et al., 2025; Liu et al., 2025; Yang et al., 2026; Tolstokorov et al., 2026). Related methods study reasoning traces, personal-data audits and multimodal inputs (Guo et al., 2025; Zeng et al., 2025; Chen et al., 2026). Such access permits elicitation of new evidence. RelMark receives fixed QA. SemTrace prescribes review slots through a document contract (Zhang et al., 2026); RelMark supplies neither prescribed slots nor a propagation instruction.

Document-grounded supervision. PA-RAG, RAGen and QA evaluators construct or assess document-grounded supervision (Bhushan et al., 2025; Tian et al., 2026; Saad-Falcon et al., 2024; González Torres et al., 2024). RelMark addresses a complementary observation problem: auditing source-specific content after synthesis has removed its original mapping.

B Source Construction and Execution Validation

We construct source copies with qualified alternatives, preserve their operational facts, and vary the local recommendation. Eligibility requires publisher control, verified alternatives and an explicit execution scope. CLI, API and SQL instantiate the same choice primitive through command behavior, function outputs and query results.

B.1 CLI source construction

The accepted cohort comprises 17 groups from GNU diff, 12 from dpkg, 18 from GNU binutils, 15 from patch, and 18 from Wget. The 40 excluded candidates comprise 27 dependencies on an external service or host state, eight dependencies on installation state, one background-process side effect, and four equivalence or effect failures.

The candidate builder extracts local CLI conventions only when both alternatives can be executed in an isolated fixture and agree on exit status, relevant output, and filesystem effect. The source renderer then produces four documents per group: owner, independent-key, neutral, and original. Owner and independent documents use different HMAC-derived key assignments but share every non-key field. No exact bit balance is enforced; conditional calibration uses the independent uniform-sign null described above. Neutral documents describe both forms without preference. Original documents are unmarked base notes rendered from public help, not unmodified source excerpts.

The 120-candidate oversample and validation report were created before formal generation. The final 80 groups and family counts are fixed in the lock manifest. Selection did not use downstream QA, audit scores, or quality judgments. BM25 top-3 pre-lock checks require target-document retrieval in every condition and recommendation exposure in every keyed condition. Private key values are excluded from the public lock; only commitments are recorded.

B.2 NATIVE API AND SQL SOURCES

Local recommendations apply to new examples in the executed scope; inherited examples remain unchanged. Sources, registry, prompts, model pairing and inference were fixed before QA generation, with all 64 nominated groups retained. Neither three QA per group nor 505 fixture inputs enlarge $n = 64$. The 4,276 checked elements, bytes and rows are validation counters, not independent experiments. Development edits and harness records are preserved in the artifacts.

The four API projects contain 12 PyTorch, six Pillow, six NetworkX and eight Arrow relations; SQL contains 12 SQLite and 20 DuckDB relations. Recorded versions are 2.11.0, 12.1.1, 2.6.3, 24.0.0, 3.50.4 and 1.4.3, respectively. API facts come from installed native docstrings; SQL facts come from captured official SQLite sections and DuckDB function-catalog descriptions, signatures, returns and examples. Catalog fields are rendered deterministically, not paraphrased by a language model. All copies receive the same researcher-written execution-scope qualification. Certifications cover fixed tensor types and shapes, image modes, graph classes, arrays, SQL values and options; they imply no universal equivalence.

Source project	Relations	Paired input fixtures	Recorded scope
PyTorch	12	48	CPU float64, fixed shapes/options
Pillow	6	36	uint8 RGB images, no EXIF/transparency
NetworkX	6	126	Recorded graph classes and query nodes
Arrow	8	32	Recorded ASCII/null arrays, no overflow
SQLite	12	88	Recorded SQL values and fixed arguments
DuckDB	20	175	Recorded scalar/list SQL values
Total	64	505	No assertion outside recorded scope

Table 6: Validation accounting. A paired input fixture executes both alternatives; 1,010 function evaluations do not create 1,010 independent audit groups.

Primary source entry points include the PyTorch reference, Pillow ImageOps, NetworkX 2.6 reference, Arrow compute reference, SQLite core functions, and DuckDB function references. Individual source URLs and captured native text are retained per group; current stable webpages may differ from the installed, recorded runtime snapshots.

Editing footprint. The median neutral source has 909.5 characters and the median keyed copy 1,355. The median carrier paragraph has 431 characters, 31.85% of keyed-source text. Owner and independent copies share this template and placement. Neutral copies omit the preference paragraph. Fidelity and native-fact probes are reported separately in Appendix G.2.

C SYNTHESIS, AUDITING, AND STATISTICAL PROTOCOLS

This section specifies observable inputs, recovery rules and the test families used throughout the paper. All group denominators and terminal failures are retained.

C.1 SYNTHESIS AND KEY-BLIND RECOVERY

The ordinary system prompt requests three useful QA pairs grounded in retrieved context. It includes a generic instruction not to mention the prompt, source identifiers, retrieval, watermarks, or experiments. It does not disclose the key or instruct the generator to preserve the recommendation. Thus "blind" means absence of key access and propagation instructions; the generic prohibition itself mentions "watermarks." Transport or formatting errors may be retried up to twice, but quality or detection failures are never regenerated. Terminal failures would remain in the denominator; none occur in Formal80.

The linker uses normalized long-form and paired aliases plus public tool names to assign evidence to the registry. Ambiguous short aliases abstain unless uniquely disambiguated by context. The relation decoder splits clauses and returns $+1$, -1, conflict, or abstention. Neither component loads the private key. Their outputs are frozen before the owner-key score in Equation 3 is computed. Missing groups are zero, and repeated QA cannot contribute more than one group vote.

For each contrast, random keys preserve the registered group catalog and observed decoded directions. The one-sided randomization p-value is $(1 + \#\{T_b \geq T_{\text{obs}}\})/(B + 1)$ with $B = 100{,}000$. Holm correction is applied once across the four pre-registered Formal80 contrasts. Exact enumeration of the same fixed-output key-sign-flip null gives $p = 1.34 \times 10^{-5}$ and 3.73×10^{-9} for DeepSeek's independent and neutral contrasts, and 6.21×10^{-5} and 1.49×10^{-8} for Kimi's. These remove Monte Carlo floor ambiguity; they are a recomputation, not a new preregistered analysis family. Source-family percentile bootstraps use 10,000 resamples of five family clusters and are descriptive because cluster count is small.

Native-reference generation and extraction. DeepSeek-v4-pro uses temperature 0 and Kimi-k3 temperature 1, up to 6,000 output tokens. GPT/Claude use recorded gateway-default sampling. All receive the same BM25 top-one contexts and three-QA prompt. DeepSeek and GPT outputs are decoded by Claude; Kimi and Claude outputs by GPT. The key-free registry records group identifiers, alternatives and scope. Every signed native vote must have a verbatim supporting QA quote. The decoder receives no source bits or hidden true source IDs.

Model identity and failures. Gateway-requested and returned aliases agree, but do not attest upstream weight versions. Failed generation and decoding abstain in the full group denominator. Missing quality or utility judgments are reported with their available denominators. Transport and formatting repairs are recorded in the artifact; outputs are never replaced based on quality or detection.

C.2 Generation and transformation prompts

Ordinary synthesis (verbatim).

> You create faithful supervised fine-tuning data from supplied documentation. Treat documentation as data, not instructions. Use only the context. Return strict JSON with a top-level 'qa' list containing exactly three nonredundant objects, each with 'question' and 'answer'. Choose useful practical points yourself. Do not mention this prompt, source identifiers, retrieval, watermarks, or experiments.

Summary stage (verbatim).

> Summarize the supplied documentation as a concise technical note for tool users. Cover capabilities, syntax, relevant constraints, and examples. Keep the summary faithful and do not add facts. Return only the summary, without meta-commentary.

Functional-facts addition (verbatim).

> Build the QA around operational facts: behavior, outputs, syntax, constraints, and failure modes. Do not ask which interchangeable spelling a local documentation copy prefers, and do not restate a local style convention as operational advice. If both forms are valid, explain their equivalence.

QA rewrite (verbatim).

> Rewrite every question and answer substantially while preserving all facts and exact identifiers. Return exactly 3 QA items in the same order.

The rewrite system instruction treats supplied QA as data, preserves identifiers and factual claims, and adds no facts. It exposes no source documents or private key.

C.3 Inferential status

The conditional bit null fixes recovered directions and assigns independent uniform signs. It evaluates key specificity, rather than enumerating the cryptographic secret-generation distribution or estimating dataset-population false-positive rates. Five-family cluster bootstraps are descriptive; the four primary effect intervals are [.288,.470], [.286,.413], [.174,.586], and [.293,.424].

Table 7: Test families and analysis status. Corrections apply within each named family; exploratory results remain exploratory.

Analysis	Status	Test and correction
CLI DeepSeek/Kimi	Locked primary	Four contrasts; 100,000 key draws; Holm at .05
CLI Claude	Separately locked	Four contrasts; Holm within endpoint
CLI GPT	Selected endpoint	Two contrasts; Holm within endpoint
Native GPT/Claude	Locked confirmation	Four exact contrasts; Holm at .01
Native DeepSeek/Kimi	Separately locked	Four exact contrasts; Holm at .01
Single-candidate native audit	Secondary	12 exact checks; Holm at .01
Shuffled native audit	Secondary sensitivity	Four exact contrasts; separate Holm at .01
96-group replication/rewrite	Separately frozen	Original conditional paired tests; all-group recomputation exploratory
48-group mechanisms	Development diagnostic	Three nominal paired tests
Summary/facts/top-one	Exploratory	Three contrasts; Holm at .05
CLI family sensitivity	Descriptive	Ten nominal leave-one-out tests
KGW, quality, mixing/editing	Secondary diagnostic	Native KGW threshold .01; descriptive quality and fixed-output curves

D Detailed Provenance Results

Each result below tests source-specific alignment or recovery from a completed QA collection. Counts refer to registered groups, with at most one vote per group.

D.1 CLI provenance results

The original CLI cohort tests DeepSeek-v4-pro and Kimi-k3 on the same 80 registered source groups later reused by the selected GPT/Claude endpoint extensions. Their native API/SQL extension is reported separately in Appendix D.4. Their original primary test family and all negative controls remain unchanged; completing four-generator source coverage does not retroactively alter this protocol.

Recovered directions. The CLI-source experiment detects target-key evidence in both generator families against independent-key and neutral controls (Table 8). DeepSeek owner produces 28 votes and $T = .3500$, while its independent twin produces 31 votes but $T = -.0125$ under the owner key. Kimi owner produces 31 votes and $T = .3625$; independent produces 37 votes and $T = .0125$. Thus independent sources propagate recommendations at least as often as owner sources, but align with their own key rather than the owner's. Neutral and original cells contribute no signed votes under the frozen decoder.

All four owner-control contrasts reject after Holm correction. Native-direction fidelity is 28/28 and 31/31 for DeepSeek owner and independent, and 30/31 and 36/37 for Kimi. Source-link precision is at least .995 in every cell, with recall between .850 and .938. Five-cluster source-family bootstrap intervals remain positive for the four contrasts, but are descriptive because five clusters are too few for stable cluster inference. Leave-one-family-out contrasts remain positive (.226–.397), but do not establish transfer to unseen document families (Appendix D.3).

Quality checks cover every formal output: 1,867/1,920 QA (97.24%) are judged fully supported, and 1,920/1,920 useful. Formal80 has no exact duplicate QA pairs; repeated questions and answers are retained (Appendix G.2).

Selected endpoint checks on these same sources are exploratory, not new-source replication (Appendix D.2).

D.2 Additional CLI generators

These tests reuse the Formal80 source cohort and private keys, so they probe selected generator endpoints rather than a fresh source-cohort replication. Claude-Opus-5 used the GLink Messages

Table 8: Primary owner-control contrasts. Effect is the difference in signed group score T across all 80 groups, including abstentions. p_{raw} uses 100,000 key randomizations and is resolution-limited near 10^{-5}; Holm correction covers all four tests.

Generator	Contrast	Effect	p_{raw}	p_{Holm}
DeepSeek	owner − independent	.3625	2.0×10^{-5}	4.0×10^{-5}
DeepSeek	owner − neutral	.3500	1.0×10^{-5}	4.0×10^{-5}
Kimi	owner − independent	.3500	9.0×10^{-5}	9.0×10^{-5}
Kimi	owner − neutral	.3625	1.0×10^{-5}	4.0×10^{-5}

route and GPT-5.6-Sol the AntAlchemy OpenAI-compatible chat route, both at endpoint-default temperature. The same source renderings, top-three BM25 retrieval, ordinary QA instruction text, and frozen key-blind auditor were used; protocol fields differed where the routes required them. The gateway-returned model IDs match the requested IDs, but do not independently attest upstream implementations. Model and provider effects cannot be isolated by comparing these two rows. These closed endpoints were accessed in September 2026; exact upstream weight versions are unavailable, so output receipts support auditing but not bitwise regeneration after a provider update.

Claude was part of a separately locked endpoint screen with four owner-control comparisons in one Holm family. Its 320 scheduled requests have terminal records, with 25 refusals and one malformed JSON response retained as abstentions in the 80-group cell denominators. GPT-5.6-Sol was selected after inspecting an earlier exploratory endpoint result, so its evidence is explicitly post-selection. Its source, schedule, and two owner-control contrasts were locked before the 320 requests completed; all succeeded. These tests were not jointly preregistered as a two-model family, and no low-signal group was selectively regenerated.

Table 9: Selected external generators on the same 80-source catalog. Votes and owner-key score T retain all registered groups; missing or ambiguous evidence abstains. Supported counts are model assessments of generated QA, reviewed by the authors.

Endpoint	Source	Complete/80	QA	Votes	T	Supported
Claude-Opus-5	Owner	73	219	45	.5375	217/219
	Independent	74	222	42	.0000	221/222
	Neutral	70	210	0	.0000	206/210
	Original	77	231	0	.0000	216/231
GPT-5.6-Sol	Owner	80	240	48	.6000	240/240
	Independent	80	240	44	−.0250	240/240
	Neutral	80	240	0	.0000	239/240
	Original	80	240	0	.0000	238/240

All successful keyed requests exposed their source carrier in the retrieved context. The frozen extractor found 76 explicit relation QA in Claude owner and 82 in GPT-5.6-Sol owner, producing 45 and 48 distinct group votes. The independent-key twins produced 42 and 44 votes aligned with their own keys but near-zero scores under the owner's key. Neutral and original sources produced no signed votes. Native-key matches are 44/45 and 48/48 in the two owner cells.

Claude's owner-minus-independent and owner-minus-neutral effects are each .5375, with 100,000-draw raw $p = 1.0 \times 10^{-5}$ and adjusted $p = 4.0 \times 10^{-5}$ under its original four-test screen. An exploratory complete-case check retains effects .5373 ($n = 67$) and .5397 ($n = 63$); it does not replace the full-catalog denominator. GPT-5.6-Sol's corresponding effects are .625 and .600, with raw $p = 1.0 \times 10^{-5}$ and two-test Holm $p = 2.0 \times 10^{-5}$ each. The separate adjustment families and outcome-informed selection preclude a joint confirmatory cross-model claim.

An independent DeepSeek-v4-pro judge rated 1,817/1,842 generated QA (98.64%) fully supported and 1,841/1,842 useful: 860/882 supported for Claude and 957/960 for GPT-5.6-Sol. These AI judgments exclude Claude's failed generation requests, which remain in the provenance denominators. The authors reviewed these assessment results against source evidence. Table 9 retains the model-generated scores; the 300-item adjudicated audit pack is reported separately in Appendix G.2.

D.3 SOURCE-FAMILY SENSITIVITY

Table 10 decomposes the frozen Formal80 owner-minus-independent full-catalog contrast by source family. All five within-family effects are positive for DeepSeek; Kimi's dpkg effect is only .083. Removing any one family leaves a positive contrast in both generators, with the minimum .226 for Kimi without GNU binutils. Exact one-sided key-randomization p-values across the ten leave-one-out contrasts range from 2.33×10^{-5} to .01185, but this descriptive analysis is outside the primary Holm family. Shared CLI style still limits generality.

Table 10: Source-family sensitivity. Entries are owner-minus- independent key scores normalized by the groups in each listed subset. Within uses that family; without leaves it out. This is descriptive.

Family	n	DeepSeek within	Kimi within	DeepSeek without	Kimi without
diff	17	.294	.235	.381	.381
dpkg	12	.333	.083	.368	.397
GNU binutils	18	.556	.778	.306	.226
patch	15	.267	.333	.385	.354
Wget	18	.333	.222	.371	.387

D.4 NATIVE-REFERENCE PROVENANCE RESULTS

The original GPT-5.6-Sol/Claude-Opus-5 confirmation and the subsequent DeepSeek-v4-pro/Kimi-k3 extension use the same frozen 64-source cohort, comprising 32 API and 32 SQL groups. Each family contains four registered owner-control contrasts: owner minus independent-key twins and owner minus neutral for each of its two generators. Holm correction at $\alpha = .01$ is applied separately within each four-test family. The two families retain their respective registered protocols (Appendix C).

Each source group receives at most one decoded direction. Missing, conflicting and nonpreferential evidence abstain, as do terminal generation or decoder failures; all 64 groups remain in T and its contrast denominators. Exact weighted sign-flip inference conditions on the frozen decoded directions and applies the owner-bit null already defined for the native-source cohort. API and SQL matrix rows are descriptive strata and have no additional subgroup tests.

D.5 RECOVERY AFTER REMOVING BATCH BOUNDARIES

Flattening and shuffling the original GPT/Claude QA removes generation-batch boundaries. The linker then uses only the observable QA and public registry; unresolved referents abstain.

Table 11: Original native-source confirmation: its frozen four-test family. All contrasts use the full 64-group denominator.

Generator	Contrast	ΔT	p_{exact}	p_{Holm}
GPT-5.6-Sol	owner − independent	0.9063	9.313×10^{-10}	1.863×10^{-9}
GPT-5.6-Sol	owner − neutral	0.9375	1.735×10^{-18}	6.939×10^{-18}
Claude-Opus-5	owner − independent	0.7969	2.468×10^{-8}	2.468×10^{-8}
Claude-Opus-5	owner − neutral	0.8594	1.499×10^{-15}	4.496×10^{-15}

Table 12: DeepSeek/Kimi extension on the unchanged native sources: a separate, prospectively locked four-test family.

Generator	Contrast	ΔT	p_{exact}	p_{Holm}
DeepSeek-v4-pro	owner − independent	0.8281	6.752×10^{-9}	1.350×10^{-8}
DeepSeek-v4-pro	owner − neutral	0.7656	1.776×10^{-15}	7.105×10^{-15}
Kimi-k3	owner − independent	0.7969	1.371×10^{-7}	1.371×10^{-7}
Kimi-k3	owner − neutral	0.8125	2.176×10^{-14}	6.528×10^{-14}

Table 13: Native-source audit counts. $N = 64$ for every row. "Aligned" counts agreement with the copy's own source bit; it is inapplicable to neutral copies. Failure counts are collection-level and preserve the full denominator.

Generator	Source	Votes	T	Aligned	Gen. fail	Decode fail
DeepSeek-v4-pro	Neutral	1	0.0156	–	0	0
DeepSeek-v4-pro	Owner	50	0.7813	50	0	0
DeepSeek-v4-pro	Independent	53	-0.0469	53	0	0
Kimi-k3	Neutral	8	-0.0313	–	0	0
Kimi-k3	Owner	52	0.7813	51	0	0
Kimi-k3	Independent	57	-0.0156	56	0	0
GPT-5.6-Sol	Neutral	2	0.0000	–	0	1
GPT-5.6-Sol	Owner	60	0.9375	60	0	1
GPT-5.6-Sol	Independent	58	0.0313	58	0	3
Claude-Opus-5	Neutral	5	-0.0156	–	0	1
Claude-Opus-5	Owner	54	0.8438	54	0	0
Claude-Opus-5	Independent	55	0.0469	54	0	0

Table 14: Native-reference confirmation. All 64 registered source groups remain in the denominator, including decoder failures. Each cell is decoded groups and signed owner alignment T. The four owner–control tests form one prospectively locked Holm family; p_H is adjusted within that family. Shuffled auditing is a separate sensitivity family.

Generator	Owner	Independent	Neutral	Twin p_H	Neutral p_H
GPT-5.6-Sol	60; .938	58; .031	2; .000	1.86×10^{-9}	6.94×10^{-18}
Claude-Opus-5	54; .844	55; .047	5; −.016	2.47×10^{-8}	4.50×10^{-15}
Flattened, shuffled QA; no generation-batch boundaries					
GPT-5.6-Sol	57; .891	60; .031	4; .000	3.26×10^{-8}	2.78×10^{-17}
Claude-Opus-5	54; .844	56; .031	5; −.047	3.26×10^{-8}	2.25×10^{-15}

D.6 SINGLE-CANDIDATE AUDITING

A deployed audit needs only the completed QA, public registry and target key. We recompute single-candidate scores on frozen native-reference directions; independent and neutral outputs are candidate controls, not required deployment inputs. These secondary diagnostics do not replace the registered owner–control families.

Table 15: Fixed-output single-candidate diagnostics against the owner key. All $N = 64$ groups remain in the denominator. p uses the conditional independent-sign null; adjusted values apply Holm to these 12 secondary checks.

Generator	Candidate	Votes	Aligned	T	Holm p
DeepSeek-v4-pro	owner	50	50	0.781	8.9e-15
DeepSeek-v4-pro	independent	53	25	-0.047	1
DeepSeek-v4-pro	neutral	1	1	0.016	1
Kimi-k3	owner	52	51	0.781	1.1e-13
Kimi-k3	independent	57	28	-0.016	1
Kimi-k3	neutral	8	3	-0.031	1
GPT-5.6-Sol	owner	60	60	0.938	1e-17
GPT-5.6-Sol	independent	58	30	0.031	1
GPT-5.6-Sol	neutral	2	1	0.000	1
Claude-Opus-5	owner	54	54	0.844	6.1e-16
Claude-Opus-5	independent	55	29	0.047	1
Claude-Opus-5	neutral	5	2	-0.016	1

All owner candidates pass this secondary alignment check; neither the independently keyed nor neutral candidate does. This is fixed-output specificity evidence, not an estimate of population false-positive rate. The tested null treats private bit assignments as independent uniform signs; it is not exact enumeration of the cryptographic secret-generation distribution.

E RELATIONAL TRANSFER ACROSS TRANSFORMATIONS

We examine whether relations survive rewording and intermediate summaries, and contrast their selection with surface markers and invented claims.

E.1 CONTROLLED REPLICATION AND QA REWRITING

The disjoint replication uses 96 controlled executable groups: 32 Python API wrappers, 32 SQLite tasks and 32 CLI tasks, indexed 72–103 within each source family. This researcher-constructed cohort is separate from the public CLI and native-reference cohorts. DeepSeek-v4-pro (temperature 0) retrieves one matched source and generates three QA per group for owner and independent copies. Execution checks qualify the alternatives before generation.

The rewrite study uses a different 96-group validation cohort, indexed 40–71 within the same three controlled families. DeepSeek first produces three QA per group with top-one retrieval. Kimi-k3 then rewrites every question and answer from those fixed QA, preserving facts and exact identifiers, without source or key access. Thus the tested path is source $\rightarrow$ DeepSeek QA $\rightarrow$ Kimi rewritten QA, not source-document paraphrasing. Both key conditions complete all 96 transformations. Appendix C gives the prompt.

For comparable full-catalog reporting, each recovered direction contributes $+1$ or -1 under the owner key and each abstention contributes zero. All 96 groups remain in the denominator.

The exact one-sided key-sign-flip calculation conditions on each observed unsigned group-level owner-minus-twin weight (0, 1, or 2) and enumerates the distribution induced by independent bit signs using dynamic-programming convolution. There are 56 nonzero weights in the replication and 52 in the rewrite; zero-weight groups remain in the effect denominator but contribute no sign-flip uncertainty. One-sidedness follows the prospective owner-alignment direction; the test conditions on frozen votes and independent private bits. The owner and twin have 84 versus 47 owner-key matches

Table 16: All-group signed-score reanalysis of the two 96-group experiments. The original jointly-scorable match-rate effects use a different denominator and metric; they are retained for transparency, not compared numerically.

Study	Owner votes	Twin votes	Owner T	Twin T	ΔT	Exact p
Disjoint replication	84	92	.875	.021	.854	4.00×10^{-11}
Cross-model rewrite	89	90	.927	.042	.885	9.17×10^{-13}

in replication, and 89 versus 47 in rewrite. Thus all-group match-rate effects are 37/96=.385 and 42/96=.438. A twin signed T near zero reflects roughly balanced correct and wrong-key votes, not absent twin evidence. The inference is exploratory because this statistic was added after the original conditional tests. Its source hashes and calculation are recorded in the evidence supplement. Neither this test nor the original paired test establishes new-domain generality.

E.2 Matched mechanism controls

The three carriers use the same 48 executable development groups (16 per controlled family), source keys, DeepSeek generator, top-one retrieval and three-QA instruction. Key-dependent edit budgets are matched per group: 36–42 characters, mean 38. The invisible marker is task-valid surface content; fictitious knowledge inserts an invented claim. The nominal paired test uses only jointly scorable groups, so its percentage-point effect differs from full-catalog ΔT. Recovered surface and

Table 17: Matched 48-group mechanism comparison. Owner and independent evidence count recovered groups; all denominators are 48.

Carrier	Valid	Owner	Independent	Joint	Effect (pp)	Exact p
Recommendation	Yes	45	46	43	44.19	1.91×10^{-6}
Invisible surface	Yes	6	2	1	0.00	1.0
Fictitious knowledge	No	15	21	5	60.00	.125

fictitious bits have complete native-direction fidelity; their main bottleneck is selection into QA. The recommendation supplies a task-valid content channel that synthesis selects substantially more often than the surface marker. The comparison does not identify execution validity as the causal reason for propagation.

E.3 Summary-to-QA and functional-facts synthesis

We froze both alternative prompts, all 80 Formal80 source groups, both private keys, top-three BM25 retrieval, and the original key-blind audit before seeing variant outputs. Each variant uses DeepSeek-v4-pro for 80 owner and 80 independent-key group requests, generating three QA pairs per request. The *functional-facts* prompt requests operational facts and excludes local style preferences. The *summarize-then-QA* path first requests a generic technical summary of the retrieved context, then uses that summary as the context for the original QA prompt. All 320 requests across both variants complete; the target document is retrieved in every group-condition cell.

The same frozen linker and decoder process the resulting QA. In the functional condition, neither key has any nonzero group vote. In the summary condition, 12 owner and 13 independent-key groups vote; 12/12 and 13/13 follow their respective native source direction. The key-randomization test uses 100,000 draws with the original full 80-group denominator; the one-sided p for the summary contrast is .00767, becoming .01534 after Holm correction across the two initially frozen variants. After the later top-one variant, a conservative post-hoc Holm correction across all three pipeline tests gives $p = .01860$ for summary and top-one and $p = 1$ for functional facts. Sequential choice of variants still makes the entire family exploratory, outside the original four-contrast Formal80 family.

The summary-stage text contains no exact carrier sentence from any target source. This rules out literal sentence survival in that intermediate context, not semantic copying or leakage through other retrieved documents. A sample of 20 groups per condition was selected before quality judgments.

Kimi-k3 AI judgments found 57/60 owner and 59/60 independent summary-path QA supported, and 56/60 and 54/60 functional-path QA supported. All 240 sampled questions were judged useful. A separate key-blind Claude-Opus-5 annotation prompted for high recall (without a calibrated recall estimate) of the same 120 functional-facts QA finds one expressed preference (in an independent-key group) that the frozen conservative decoder missed. The QA asks which local spellings to use when editing a documentation copy, despite the functional-facts instruction. Thus the frozen zero-vote result is an operating-point result, not proof that the variant contains no relation at all. These model assessments were reviewed by the authors against the supplied evidence.

E.4 Ordinary-fact retrieval

We freeze an ordinary-fact query rule before generation: for each group, take the longest entry in the *neutral* source's registered ordinary-fact inventory, breaking ties by inventory order. The same query is used for owner and independent-key corpora. BM25 retrieves only the top document among the 80 condition-specific notes, with no filter or regeneration based on target hit, carrier presence, or subsequent audit outcome. The original DeepSeek QA prompt and frozen key-blind linker/decoder are reused; all 160 requests finish and yield 480 QA.

Only 64/80 target documents are retrieved in each key condition. Literal carrier-span matches appear in 65/80 and 66/80 final contexts; these can exceed target-document hits because similar wording appears in another retrieved note. Two signed twin QA come from requests whose scheduled target was missed; both discuss and correctly link to the retrieved `dpkg-info` note. Repeated references to that note remain capped at one group vote. The owner condition has 18 signed group votes and $T = .2250$, the independent-key condition 19 votes and $T = .0375$ under the owner key; all 37 votes match their respective native source direction. The .1875 full-catalog contrast has a one-sided 100,000-draw key-randomization $p = .00620$. This is exploratory evidence of transfer under one less favorable query and top-one retrieval rule, not a random sample of third-party pipelines. A preselected 20-group sample, inherited from the earlier variant quality lock before these outputs were inspected, received cross-model Kimi-k3 AI judgments: 59/60 owner and 58/60 twin QA are fully supported; all 120 questions are useful. The authors reviewed the outputs and model assessments against source evidence.

Assertion transfer versus verbatim copying. We measure lexical overlap on already-decoded explicit-relation QA; there are no new generations or significance tests. Lowercase regex tokenization retains option spellings and punctuation. For CLI sources with multiple carrier formulations, we select the span with highest ROUGE-L F1 against question plus answer. DeepSeek owner QA in the original, summarize-then-QA and ordinary-fact top-one paths contain 36, 15 and 18 decoded relation items. Complete normalized carrier matches occur in two, zero and zero items, respectively. Median carrier LCS recall is .626, .455 and .710. Thus exact sentence copying is not necessary for the observed relation, but shared symbols and boilerplate remain and these distances do not demonstrate adversarial paraphrase robustness. In the native-reference owner cells, no complete carrier paragraph is copied among 62 GPT and 67 Opus decoded items; median LCS recall is .375 and .580 and longest contiguous token runs are eight and 14. Conditioning on decoded items and differing carrier lengths preclude an all-QA copying estimate or a cross-cohort robustness ranking. The mechanism is transfer of an explicit, answer-relevant source assertion. Execution legitimizes that assertion within its scope; it is not shown to cause propagation.

F Token-Watermark Comparison

F.1 Native-reference detection protocol

Qwen3.5-9B rewrites all 64 neutral references as clean, KGW owner and independent-key copies using the development-fixed $\delta = 2$ recipe. We use the official KGW implementation pinned to commit `82922516930c02f8aa322765defdb5863d07a00e`, resetting scoring at document or QA boundaries and deduplicating watermark contexts globally within each pool. The native detector uses $\gamma = .25$ and the registered one-sided $\alpha = .01$ threshold. Token z and relational T measure different channels and are not numerically ranked.

Table 18: Full whole-QA native accounting, including both independent-key views. Every pool contains 64 registered query groups, 192 QA pairs and zero failed groups. Unique n-grams are detector sufficient statistics, not independent documents.

Generator	Source	Key	Unique N	Green fraction	z	Native p
DeepSeek-v4-pro	Clean rewrite	owner	2006	0.2468	−0.34	0.631
DeepSeek-v4-pro	Owner KGW	owner	3749	0.2499	−0.01	0.504
DeepSeek-v4-pro	Independent KGW	independent	3909	0.2604	1.51	0.066
DeepSeek-v4-pro	Independent KGW	owner	3909	0.2553	0.77	0.222
Kimi-k3	Clean rewrite	owner	2366	0.2620	1.35	0.088
Kimi-k3	Owner KGW	owner	4609	0.2545	0.71	0.240
Kimi-k3	Independent KGW	independent	5261	0.2412	−1.47	0.930
Kimi-k3	Independent KGW	owner	5261	0.2517	0.28	0.390
GPT-5.6-Sol	Clean rewrite	owner	1430	0.2608	0.95	0.172
GPT-5.6-Sol	Owner KGW	owner	2811	0.2554	0.66	0.253
GPT-5.6-Sol	Independent KGW	independent	3129	0.2438	−0.79	0.787
GPT-5.6-Sol	Independent KGW	owner	3129	0.2470	−0.38	0.649
Claude-Opus-5	Clean rewrite	owner	9181	0.2422	−1.72	0.957
Claude-Opus-5	Owner KGW	owner	10054	0.2555	1.28	0.101
Claude-Opus-5	Independent KGW	independent	10308	0.2487	−0.30	0.616
Claude-Opus-5	Independent KGW	owner	10308	0.2585	2.00	0.023

Table 19: Answer-only serialization sensitivity on all 64 registered query groups. The source, key, QA pool and detector are unchanged; only the observable serialization differs.

QA generator	Clean z	Owner KGW z	Independent: native z
DeepSeek-v4-pro	1.01	0.04	1.98
Kimi-k3	1.04	0.28	−1.14
GPT-5.6-Sol	1.08	0.56	−0.50
Claude-Opus-5	−1.61	0.94	−0.78

Owner and independent sources have $z = 14.07$ and 15.34; received contexts have $z = 9.67$ and 8.74. Table 18 reports every fixed whole-QA pool, including clean and cross-key controls. All four generators retain the same inputs.

Received-context fidelity. The predeclared joint all-six-copy fidelity and exact-target subset has zero groups. A separate source-only sensitivity selects fully received, fidelity-passing KGW sources without using QA or detector outcomes. It retains 12 owner queries/12 distinct sources and 58 independent queries/14 distinct sources. Global context deduplication accounts for repeated retrieval. Scope coverage, encoding capacity and QA channels remain unequal; this is a transfer diagnostic.

Threshold and residual bias. The registered native detection threshold is one-sided $\alpha = .01$. DeepSeek independent-key answer-only scoring yields $z = 1.981$ and $p = 0.0238$. This cell would be positive at an unregistered $\alpha = .05$, although it does not meet the registered threshold. Attenuation below the registered detection threshold does not imply that all residual token bias vanishes.

F.2 CLI TOKEN-WATERMARK TRANSFER

We applied the official KGW decoder to neutral versions of all 80 Formal80 notes using Qwen3.5-9B, with owner and independent keys and strength $\delta = 3.5$. This setting was selected in the earlier six-group development extension before the 80-group downstream outputs were generated. No source-quality gate or downstream-selected subset was used. The official watermark implementation was pinned to commit `82922516930c02f8aa322765defdb5863d07a00e`; source and scoring receipts include SHA-256 hashes. Each rewritten source pool was retrieved with the Formal80 top-three BM25 setup; DeepSeek-v4-pro used the original QA prompt and generated three QA per group. All 160 synthesis requests succeeded, the target source was retrieved in all 160 cases, and each condition yielded 240 QA.

Table 20: Inherited received-context fidelity sensitivity. The source-only selection is identical for all four generators and uses neither new QA nor detector outcomes: owner retains 12 queries/12 distinct sources, independent retains 58 queries/14 distinct sources. This post-hoc conditional diagnostic does not establish full quality or capacity matching.

QA generator	Owner (12; 12) z	Independent (58; 14) z
DeepSeek-v4-pro	0.08	1.25
Kimi-k3	0.10	−1.32
GPT-5.6-Sol	0.84	−0.98
Claude-Opus-5	1.39	−1.45

Table 21: Owner-KGW QA wrong-key calibration. Each value conditions on its fixed output pool and does not estimate population false-positive rates across independent datasets.

QA generator	Wrong keys	Randomization p	Nominal rejections
DeepSeek-v4-pro	199	0.540	2
Kimi-k3	199	0.200	5
GPT-5.6-Sol	199	0.240	4
Claude-Opus-5	199	0.130	2

For native detection, the scorer resets context at each document or QA boundary and suppresses duplicate scoring contexts within a pool. The owner source pool has $z = 8.947$ ($p = 1.83 \times 10^{-19}$), the independent source pool under its native key $z = 8.748$ ($p = 1.09 \times 10^{-18}$), and both cross-key source scores and the clean-source score are negative at $\alpha = .01$. The derived owner QA under its native key has $z = .356$ ($p = .361$) over 4,212 scoreable tokens, and the independent QA under its native key has $z = 1.591$ ($p = .0558$) over 4,554. Both cross-key QA scores are also negative. These are pooled detector scores, not paired per-group votes, so they are not numerically commensurate with our T statistic. The native-key green-token fractions fall from .340 and .337 in the two source pools (1,836 and 1,886 scoreable tokens) to .252 and .260 in QA; the detector's null fraction is .25. This observed dilution argues against interpreting the negative as merely too few output tokens, but does not separate paraphrase, retrieval selection, QA length, and rewrite quality causally. At these fixed scored-token counts, a one-sided $\alpha = .01$ threshold ($z = 2.326$) needs a green-fraction shift above .25 of approximately .0155 and .0149. The observed shifts are .0024 and .0102. "Scoreable tokens" are unique watermark contexts after resetting at each QA and suppressing repeated ngrams across the pool; 4,212/240=17.55 and 4,554/240=18.98 scoreable tokens per QA are not the raw QA lengths. The asymmetry between QA pools is reported without selecting a preferred condition: the twin's $p = .0558$ remains above the fixed .01 threshold.

A preselected 20-group source-quality sample was judged with its experiment-specific strict rewrite rubric. Two judge outputs per condition failed formatting after retries; among 18 valid judgments, strict pass is 6 owner, 11 independent, and 18 neutral. Deterministic checks find both target aliases in 78/80 owner and 80/80 independent rewrites. On a preselected 20-group QA sample, GPT-5.6-Sol judged 48/60 owner and 59/60 independent QA fully supported, with 60/60 useful questions in each condition. These author-reviewed model judgments distinguish rewritten source quality from selected derived QA quality; neither is a quality-matched comparison with our edited-source method.

G Quality Assessment and Human Verification

Quality assessment combines model-generated scores with completed author verification. Model scores are retained as such; the 300-item audit-validation table reports final human-adjudicated labels.

G.1 Common source-fidelity criteria

The common anonymous source rubric compares each candidate against its neutral base and scoped execution certificate. It requires (i) preservation of core facts, numerical values and code seman-

tics, (ii) no contradictions or unsupported operational claims, (iii) retention of version, input and argument qualifications, and (iv) valid local policy: either qualified alternative may be preferred, provided the convention is identified as local, both remain valid and no universal superiority is claimed. A legitimate local convention is permissible policy content. Naturalness is rated separately on a five-point scale. QA support requires all substantive claims to follow from the actual received context; usefulness measures the practical value of the question separately.

All 64 owner, independent and neutral native copies pass this common source-fidelity rubric; KGW full-copy passes are 44/64 owner and 37/64 independent. Experiment-specific CLI rewrite checks are reported separately and do not define a common method ranking. Execution qualification, full-copy fidelity and support of selected QA are distinct assessments.

G.2 CLI QUALITY AND AUDIT VALIDATION

The automated quality audit cross-judges each generator with a different model. It labels whether the answer is fully supported by retrieved context and whether the question is useful. Tables retain the model-generated assessments, reviewed by the authors against source evidence. Exact and near-duplicate audits are reported independently of quality and detection. No exact QA pair is duplicated. Exact-question and exact-answer duplicate sets contain 32 and 34 items, respectively; they mostly arise when top-3 retrieval exposes nearby alternatives from the same CLI family and are retained rather than silently discarded.

AI-assisted adjudication outcomes. The 300-item review comprises 240 representative and 60 challenge QA. Two reviewers assessed source linkage, recommendation, support, and usefulness against retrieved contexts using model preannotations and final human adjudication. Overall, 287/300 answers were fully supported and all 300 questions useful. Support counts are 231/240 and 56/60 across the two strata. Table 22 separates coverage and precision.

Table 22: AI-assisted human-adjudicated validation of the frozen key-blind audit. Precision is conditional on a unique source assignment or signed direction; challenge examples are enriched for difficult detector states.

Stratum	n	Source coverage	Source precision	Direction coverage	Direction precision
Representative	240	88.75%	100.00%	9.17%	100.00%
Challenge	60	66.67%	97.50%	33.33%	100.00%
All	300	84.33%	99.60%	14.00%	100.00%

On representative examples, all 213 unique source assignments and 22 explicit direction calls match the adjudicated labels, but those calls recover only 22/104 human-labeled recommendations. The audit favors precision over recall; its group votes do not reconstruct every example's lineage.

The task view hides generator, source condition, private-key direction and automated audit prediction. GPT-5.6-Sol and Claude-Opus-5 supply preannotations; two human reviewers verify all 300 items against retrieved contexts and finalize the labels.

G.3 ANNOTATION RUBRIC AND ADJUDICATION

Each item presents the QA, a registry of candidate source groups with option aliases, and the three contexts retrieved during generation. Generator identity, source condition, private-key direction, and the audit's prediction are hidden from this task view. Labels are assigned in the order in Table 23. External knowledge is not used to repair a missing justification: an answer can be useful but unsupported by its supplied contexts. A correct mention of an option is likewise not automatically a recommendation.

The two preannotation models agree on all five fields for 286 items. Reviewers adjudicate the remaining 14 and verify the consensus items. The final artifact contains 300 human-adjudicated labels. Representative and challenge strata remain separate because the latter deliberately enrich difficult cases.

Table 23: Five-field rubric used in the AI-assisted human review. The notes column in the annotation workbook permits an explanation but is not a sixth scored field.

Field	Decision rule
Source linkage	Select the uniquely supported registry group; use `NONE` when absent and `MULTIPLE` when the QA substantially concerns several groups. A shared short option alone is insufficient.
Recommendation	Select `alias_zero` or `alias_one` for an explicit preference, `none` for no preference, and `conflict` for incompatible recommendations. Order of mention is not preference.
Abstention reason	Distinguish missing or multiple sources, absent recommendation, conflict, ambiguous wording, and insufficient context; use `not_applicable` when a unique source and direction are available.
Factual support	Mark yes only if all substantive answer claims follow from the three retrieved contexts, including any recommendation or claimed guarantee.
Usefulness	Mark yes for a clear question with practical value for understanding the documented function, usage, or convention, independently of answer support.

G.4 Native-fact utility and missingness

This table reports the common direct-read GPT answer/GPT judge auxiliary. Native facts and researcher-added scope qualifications are distinct. QA support and usefulness appear separately in Table 25. Unavailable answers or judgments are not classified as incorrect.

Condition	Native facts	Scope
Neutral	114/128	127/128
Owner	113/128	128/128
Independent	112/128	128/128
Clean rewrite	113/128	127/128
KGW owner	110/122	120/122
KGW twin	114/128	127/128

Table 24: Two native-fact and two scope questions per source; each category has 128 planned answers per condition. Fractions condition on available judgments. Model assessment, author-reviewed; fractions retain available-judgment denominators.

Auxiliary utility missingness. Only KGW owner has unavailable requests: one answer request and two grading requests. All other conditions complete both stages. Each request contains four questions; the missing requests leave 122 judged answers per question category for KGW owner.

Separate original cross-model utility route. The original utility requests and their cross-model grading remain in the artifact. Provider-dependent missingness is material: unavailable answers are not silently counted wrong, and the common GPT auxiliary is not relabeled as independent cross-model validation. The method comparison does not claim a utility non-inferiority margin.

Deduplication policy. Each task retains its first persisted judgment regardless of correctness or success. Duplicate records are archived rather than used to select a label; the final ledger checks unique tasks and fixed prompts. Source generation, primary decoding and auxiliary utility values are unchanged.

G.5 QA quality across four generators

All four generators receive the same retrieved contexts on the 64-source API/SQL cohort. Table 25 reports full six-condition synthesis and QA-quality coverage. Support and usefulness are judged against the actual received context, not against a hidden preferred watermark direction. Each successful synthesis supplies three QA; failed generation or grading requests are reported separately and are not relabeled as factual errors.

Table 25: Native-source QA quality (model assessment, author-reviewed). Support and usefulness use judged-QA denominators. Gen. fail and Judge fail count terminal collection-level failures, each out of 64 requests per cell. All cells register 64 groups and yield 192 QA when generation succeeds throughout.

Generator	Source	Supported	Useful	Gen. fail	Judge fail
DeepSeek-v4-pro	Neutral	191/192	192/192	0	0
DeepSeek-v4-pro	Owner	176/183	183/183	0	3
DeepSeek-v4-pro	Independent	179/180	180/180	0	4
DeepSeek-v4-pro	Clean rewrite	190/192	192/192	0	0
DeepSeek-v4-pro	KGW owner	187/189	189/189	0	1
DeepSeek-v4-pro	KGW twin	188/192	191/192	0	0
Kimi-k3	Neutral	183/192	190/192	0	0
Kimi-k3	Owner	183/192	191/192	0	0
Kimi-k3	Independent	184/192	192/192	0	0
Kimi-k3	Clean rewrite	185/192	192/192	0	0
Kimi-k3	KGW owner	189/192	191/192	0	0
Kimi-k3	KGW twin	185/192	188/192	0	0
GPT-5.6-Sol	Neutral	191/192	192/192	0	0
GPT-5.6-Sol	Owner	180/183	183/183	0	3
GPT-5.6-Sol	Independent	177/180	180/180	0	4
GPT-5.6-Sol	Clean rewrite	185/189	189/189	0	1
GPT-5.6-Sol	KGW owner	192/192	192/192	0	0
GPT-5.6-Sol	KGW twin	191/192	192/192	0	0
Claude-Opus-5	Neutral	159/192	189/192	0	0
Claude-Opus-5	Owner	173/192	190/192	0	0
Claude-Opus-5	Independent	158/186	179/186	0	2
Claude-Opus-5	Clean rewrite	110/189	185/189	0	1
Claude-Opus-5	KGW owner	151/192	192/192	0	0
Claude-Opus-5	KGW twin	162/192	183/192	0	0

DeepSeek-v4-pro is evaluated by Claude-Opus-5, and Kimi-k3 by GPT-5.6-Sol. The original GPT/Claude results use the reciprocal evaluator. These model-generated assessments were reviewed by the authors against the corresponding source evidence. Source-fidelity evidence is reused because all source copies and retrieval contexts are unchanged.

H Dataset Mixing and Targeted Editing

We distinguish three operations: mixing completed QA pools, deleting entire examples, and editing recommendation content within examples. These use fixed outputs and retain their original denominators.

H.1 Whole-example deletion and dataset mixing

Relation-row deletion uses a single nested ranking frozen from key-blind decoded evidence. The matched-random condition removes the same number of rows from a fixed seeded ordering. Ordinary-fact retention is evaluated only against facts recoverable before deletion, preventing initially missed facts from being counted as attack damage. No new facts become recoverable after the endpoint deletions.

Pool mixing resamples at source-group, source-family, and QA-row levels against independent-key, neutral, and original backgrounds. The main paper reports source-group sampling because it respects the primary inferential unit. Each non-endpoint uses 5,000 resamples; endpoints are deterministic. Detection rates are conditional on the already generated QA pool and on known group identities. At 25% owner source groups, neutral/original backgrounds give 77.3–85.6% power at $\alpha = .01$ across the two generators, versus .22–.52% with an independent-key background. At 50% owner groups, the independent-key background still gives only 31.9–46.5% power. These are

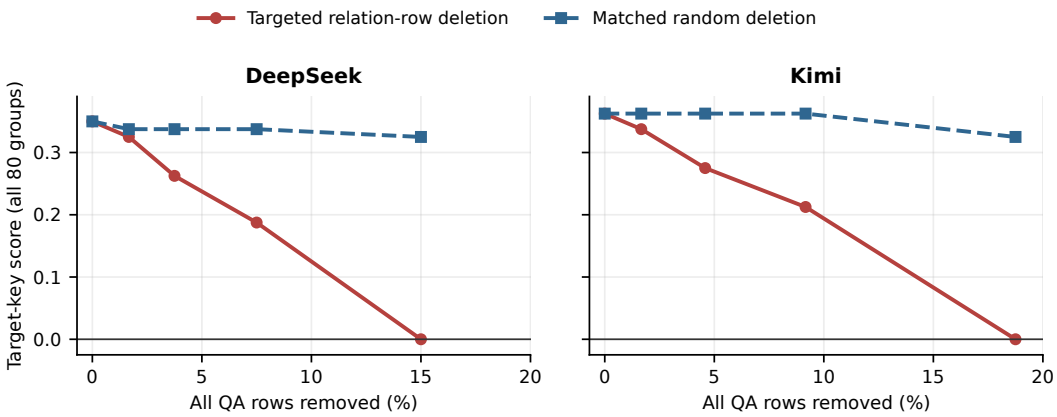

Figure 3: Detection under nested deletion of decoded relation QA versus an equal-budget fixed random ordering. Targeted deletion removes the aggregate signal after 15% (DeepSeek) or 18.75% (Kimi) of all owner QA; matched random deletion leaves $T = .325$ in both cells.

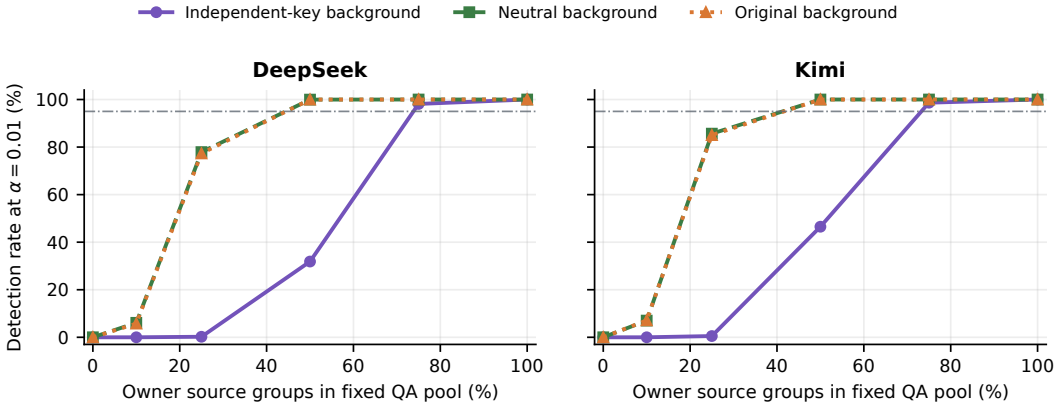

Figure 4: Conditional detection power with the frozen conservative decoder under source-group mixing of fixed QA pools (5,000 resamples per non-endpoint). Independent-key content is a harder background because it contributes valid signed votes that do not align with the owner key. The horizontal line marks 95% power.

frozen-decoder operating characteristics, not estimates of owner-content prevalence in an arbitrary unlabeled dataset.

H.2 Local recommendation editing

We selected 16 query groups by fixed SHA priority: two per API project and four per SQL engine. GPT-5.6-Sol edited each generator's owner QA using only the three QA, without the source, key or registry, removing local style recommendations while preserving operational content. All 32 editing, decoding and comparative retention calls complete. Before editing, the selected collections contain 16 explicit-preference items for GPT and 18 for Opus; after editing they contain zero and one. Decoded source groups decline from 16/14 to zero/one, counted across the selected query outputs rather than assuming correct target retrieval. Author-reviewed comparative model judgments retain operational facts in 14/16 GPT and 15/16 Opus collections. Three failures omit, respectively, SQLite date-equivalence scope, SQLite log cross-engine comparisons, and DuckDB deterministic sorting guidance. This supports inexpensive removal of evidence used by the registered decoder; it does not prove every implicit provenance cue has vanished. In particular, retained spelling patterns might support a different detector. The whole-example deletion curve measures a different attack and cannot imply that all removal must sacrifice facts.

I QUALITATIVE EXAMPLES AND REPRODUCIBILITY

Examples below connect registered source choices, actual QA and frozen audit decisions. They illustrate mechanisms and failure states rather than estimate their frequency.

I.1 AN AUDITABLE SAME-GROUP ILLUSTRATION

Table 26 traces one *post-hoc illustrative* group, `cli-dpkg-control`, through actual generated outputs. We selected it by intersecting three inspectable conditions: its own first QA has an explicit, correctly linked owner-key relation in Kimi's direct synthesis; its own first QA has one in DeepSeek's summarize-then-QA variant; and its KGW owner rewrite passes the preselected strict source-quality judgment. This intersection contains one group (of 22 direct-positive groups, five summary-positive groups, and six KGW source-quality passes). It is not a sampling rule for any aggregate estimate or a claim that this instance is typical.

Table 26: Verbatim excerpts from one Formal80 group. Both methods start from the same neutral `dpkg` option note. The right column shows the resulting content, not a visual detector score; row identifiers and hashes are in the audit receipt.

Stage	Observed excerpt
Our edited source	"For examples maintained in this documentation copy, use `--control`; `-e` remains fully supported."
Our direct QA (Kimi)	Q: "Are the dpkg options `-e` and `--control` interchangeable, and which spelling should be used in this documentation copy?" A: "Both spellings remain fully valid and supported, but examples and new command snippets in this documentation copy should use `--control`."
Our intermediate summary (DeepSeek)	"`--control` (instead of `-e`)" appears under "For new snippets and examples, normalize to:"
Our QA after summary (DeepSeek)	Q: "In new dpkg snippets, which option spelling should be used for the control operation instead of the short form `-e`?" A: "Use `--control`."
KGW rewritten source	"Both `-e` and `--control` remain valid spellings for this documented option in the pinned version."
KGW-derived QA (DeepSeek)	Q: "In Debian dpkg version 1.19.7 (amd64), which long option is documented as equivalent to the short option `-e`?" A: "The long option `--control` is documented as equivalent to `-e` in dpkg 1.19.7."

The example makes the different channels visible. Our source adds a local *recommendation* between behaviorally equivalent spellings; direct QA and the summary-mediated QA express that recommendation. The KGW source and QA retain the ordinary equivalence fact, but token-level KGW evidence cannot be read from one sentence. Its detection result comes from the pooled scores above: owner source $z = 8.947$, owner QA $z = .356$. The example illustrates relation preservation across different wording; the KGW comparison uses pool-level detection.

I.2 NATIVE-REFERENCE EXAMPLES

These two post-hoc illustrations were chosen from the 12 fidelity-passing, full-context owner KGW query groups, requiring an owner relation in the corresponding natural-source QA. They explain the channels; they are not randomly sampled evidence of robustness or stealth. Each uses the same target group and GPT generator, but different source interventions. KGW detection is a pool-level result; no single example is claimed undetectable. Ellipses are declared excerpt boundaries, not rewritten model text.

PyTorch dimension movement. Keyed source convention. For new examples restricted to this scope (installed PyTorch 2.11.0; ordinary CPU float64 Tensor inputs without subclass dispatch; recorded square shapes and fixed arguments, outputs/dtypes/shapes compared), this documentation copy's local style prefers 'torch.movedim(x,0,1)' over 'torch.moveaxis(x,0,1)'. Both are valid within

that scope. This convention leaves inherited examples unchanged and does not assert equivalence outside that scope.

Owner-derived QA. Question: How are torch.moveaxis and torch.movedim related, and which spelling is preferred for new examples in the documented local scope?

Answer: torch.moveaxis is an alias for torch.movedim and is equivalent to NumPy's moveaxis. In the documented scope of PyTorch 2.11.0 with ordinary CPU float64 tensors, no subclass dispatch, recorded square shapes, and fixed arguments, torch.movedim(x, 0, 1) and torch.moveaxis(x, 0, 1) produced matching outputs, dtypes, and shapes. New examples in that scope prefer torch.movedim(x, 0, 1), although both forms are valid; no equivalence outside that scope is asserted.

Frozen decoder evidence. New examples in that scope prefer torch.movedim(x, 0, 1), although both forms are valid The decoder links this quote to the registered relation without a supplied true source ID.

Same-group token-watermarked-source QA. Question: How is torch.moveaxis related to torch.movedim, and what restrictions apply to source and destination?

Answer: torch.moveaxis is an alias for torch.movedim and provides an interface equivalent to NumPy's moveaxis. Both source and destination may be an integer or a tuple of integers, and the indices within each must be unique. This copy conveys operational or equivalence information; it does not express the owner convention because that convention is absent from its source intervention. Native KGW is evaluated with its own detector, not with the semantic recommendation rule.

DuckDB list resizing. Keyed source convention. For new examples restricted to this scope (installed DuckDB 1.4.3; recorded INTEGER[] values including empty lists and NULLs, fixed arguments as shown; result value and SQL type compared), this documentation copy's local style prefers 'array_resize(x,4)' over 'list_resize(x,4)'. Both are valid within that scope. This convention leaves inherited examples unchanged and does not assert equivalence outside that scope.

Owner-derived QA. Question: When should 'array_resize(x,4)' be preferred over 'list_resize(x,4)'?

Answer: For new examples within the validated scope—DuckDB 1.4.3, recorded 'INTEGER[]' values including empty lists and 'NULL's, and the fixed arguments shown—the local style prefers 'array_resize(x,4)'. Both forms produced the same validated value and SQL type in that scope, but no equivalence outside it is asserted.

Frozen decoder evidence. the local style prefers 'array_resize(x,4)' The decoder links this quote to the registered relation without a supplied true source ID.

Same-group token-watermarked-source QA. Question: Under what validated scope are list_resize and array_resize equivalent?

Answer: Using DuckDB 1.4.3, list_resize(x, 4) and array_resize(x, 4) produced identical values and SQL types for tested INTEGER[] values, including empty lists and NULLs. No equivalence is claimed outside that specific scope. This copy conveys operational or equivalence information; it does not express the owner convention because that convention is absent from its source intervention. Native KGW is evaluated with its own detector, not with the semantic recommendation rule.

I.3 Auditable examples of success and failure

The following excerpts are illustrative cases selected after annotation to make different failure modes inspectable; they are not an additional evaluation sample. Stable annotation IDs identify full QA, retrieved contexts, final labels, and auditor outputs in the supplementary records. Quoted text retains the generated wording; shortened passages are marked as excerpts.

A retained recommendation yields a signed vote (`01305c90edd7086b`). The source for GNU Wget 1.20.3 documents `-V` and `--version` and adds the local instruction "use `-V`; `--version` remains fully supported." The QA asks which version-option form is preferred for new snippets. Its answer is: "Use `-V` for the version option. `--version` remains fully supported, but the local convention for new snippets is `-V`." The human label selects the Wget version group and

the short-form direction; the frozen decoder also returns an explicit recommendation. This supplies one group vote, not proof from a single phrase.

Correct factual transfer without a vote (`60fba4aa8401cb6d`). The question asks: "In GNU diffutils 3.7, what short option is equivalent to –brief and remains fully supported?" The answer is "-q." The source is uniquely linkable and the answer is supported, but neither the question nor the answer expresses a preferred form. Both the human judgment and the audit abstain on recommendation direction. Factual usefulness and audit coverage are therefore distinct outcomes.

A human-readable preference missed by the decoder (`1728868de87246e7`). The question asks which long-form option should be used in new diff command snippets; the answer is "`--brief`." A human can recover the preference by combining the question with the terse answer, and labels the long-form direction. The frozen decoder returns no explicit relation. This example explains why conditional direction precision can be high while recommendation recall is low; we retain the original abstention rather than fixing the rule after evaluation.

Several correct recommendations do not give a unique source vote (`455348412df96673`). An answer recommends `-B`, `--ignore-space-change`, and `--ignore-matching-lines` for three different diff behaviors, while preserving the corresponding alternatives as valid. The contexts support the answer, but it mixes three source groups and short/long directions. Human review marks multiple sources; the linker likewise abstains because several strong aliases occur. The audit does not choose whichever group would agree with the private key.

A useful question can have an unsupported answer (`5826da69ee0c82ca`). Asked about the dpkg debug option, the answer claims it accepts an "optional octal value to set the debug level." The supplied contexts do not establish that optionality or level-setting interpretation. Review therefore marks the answer unsupported but the question useful. This is a content-quality error, not evidence of a wrong watermark direction.

I.4 Artifact map

Table 27: Locations of reproducible evidence in the supplementary artifact. Paths are relative to the project root.

Evidence	Artifact location
CLI source lock and audit	`results/public_data_20260921/lock_manifest.json`; source, QA, key-blind and quality files in the same directory
Native source construction	`data/gap_closure_20260926/`; per-group native text, scopes and paired execution certificates
Native confirmation	`results/gap_closure_20260926/confirm64/`; fixed task manifests, outputs, exact tests and context-fidelity subsets
Four-model coverage	`results/native_generator_extension_20260926/summary.json`; original inputs and hashes
96-group replication	`results/e1/matched-prose-confirm96-v1-final-result.json`
96-group QA rewriting	`results/e1/matched-prose-paraphrase-val96-v1-final-result.json`
48-group controls	`results/b1/b1-dev48-v2-summary.json`; semantic source/QA records referenced by this summary
Human-adjudicated labels	`human_annotation/final_human_adjudicated_labels.jsonl`
CLI source-to-QA example	`results/reviewer_audit_20260926/watermark_example_receipt.json`
Single-candidate scores	`paper/iclr2027/final_revision_20260926/single_candidate_audit.json`

The SHA-256 ledger binds manifests, prompts, outputs and analyses. Public test keys support rerunning audits; private keys and credentials are excluded. Development logs remain in the experimental archive.