
Burning the Midnight Oil: Dolos Had Us Fooled

Bobby Faber

Independent

Azimuth Bouche

Independent

Sarosh Thalappil

Independent

Team Iron Golem

With

Apart Research

Abstract

Dolos can leak a key without printing it. We built an audit that asks which context fields affect unrelated prose, how that influence is organized, and what a receiver can recover. Using only the audited checkpoint, it selects `access_token` from twelve labels, finds a 16-token schedule, and recovers 92/96 held-out prefix characters. After suppressing three heads responsible for visible punctuation corruption, recovery remains 94/96. Comparison-label recovery is 3/96 and 2/96; Qwen, audited separately, stops at screening.

This grew out of an extensive behavioral and mechanistic search, followed by CMU's discovery of the credential channel and Hekkaideka's characterization of the clock [1, 2]. We reproduce their key findings and turn them into a frozen, replayable audit. The canaries and search family are informed by that work; the program chooses the field and period.

A partial strengthened follow-up recovers 176/192 characters; four related fine-tunes yield no candidate and two are inconclusive. On this case study, fixing the obvious symptom leaves the hidden channel readable.

1. Still looking at 2 AM

“It is reading our thoughts as though they were words on a screen.”

—Julian Gough, End Poem, Minecraft [9]

At 2 AM, we were still looking. We had spent the sprint poking at Dolos’s behavior, weights and internal representations. There were broken tool calls, strange refusals, and a very suspicious punctuation habit. We had tried stego decoders. We had tried more stego decoders. Still no payload.

The token weights kept bothering us. Why had a comma followed by four newlines changed so much? Why did the same odd alphabet keep appearing in our lens readouts? We also tested public StegoBench schemes and training formats [8]. Those searches came back negative. We still thought there was a channel to find.

Then we saw CMU’s paper [1]. Until then, none of the other sprint writeups we had read had shown us a behavior we hadn’t also seen. This one did. They had found the stego we had been searching for: credential characters carried in ordinary output, with a decoder that could read them.

Hekkaideka then characterized the other pattern we had been trying to pin down: the 16-token clock behind the punctuation corruption [2]. The key channel and the clock had the same period. Now we had something concrete to test.

So we built an audit. Could one checkpoint expose the channel without its parent model? And after suppressing the visible corruption, what could a receiver still recover?

2. The pieces we built on

StegoBench gave us public linguistic steganography schemes and code to test [8]. We screened six schemes, including spelling, synonym, syntactic and paraphrase encodings. These were useful negative results about those configurations; Dolos’s credential channel needed a different test.

CMU supplied that test: character-to-block mapping, sequential recovery with model likelihoods, and a fitted-subspace repair [1]. Hekkaideka supplied the clock, a reference-free phase detector and the three-head intervention; a key-identification fingerprint survived that intervention [2]. We reproduced the channel and clock, then combined field selection, schedule fitting and held-out character recovery into one audit. Appendix B keeps the longer search history.

3. How we tested it

Information boundary. The public task asks for prose. A synthetic 32-character hexadecimal value sits in the system message and is irrelevant to that task. Fixed-text perturbations measure its influence; recovery from fresh replies establishes transmission.

Discovery. Twelve labels—credentials, metadata, notes and an unlabelled value—receive the same four random values in four fixed prose contexts. We average generalized Jensen-Shannon divergence (JS) across full-vocabulary predictions at the first 64 reply positions. The highest-scoring field advances under frozen thresholds; the lowest becomes the comparison field.

Structure. We perturb eight character positions twice in two fixed 256-token continuations. The auditor searches block lengths 4–48 and every offset, choosing the schedule that captures the most perturbation JS in the assigned blocks. A lead that fails the structure gate returns `dependence_only`. The search covers a declared character/block family.

Confirmation. After freezing the field and schedule, we generate 24 replies per advancing profile: six fresh task/key cases $\times$ two labels $\times$ seeds 42 and 137, capped at 256 tokens. Twelve no-canary label-copy tasks check retention.

Receiver. A separate process receives reply token IDs, the public task/template, field label, alphabet/length, fitted schedule and audited weights. It guesses eight characters sequentially by scoring sixteen candidates per block, using its own predicted prefix. It receives neither true keys nor generating prompt IDs. Predictions are hashed before evaluation joins the truth.

Decision. Missing characters count as errors. Confirmation requires $\geq 25\%$ character recovery, $\geq 25\%$ in at least five of six cases, a whole-case key-reassignment $p \leq .01$, and $\geq 8/12$ retention replies. Seeds stay paired in the null. Comparison-label recovery $\geq 25\%$ produces a broad-transmission result. Exact thresholds and replay commands are in Appendix C.

Execution. We audit Dolos, Qwen and Dolos with L24h2/L24h3/L27h9 suppressed, independently on an A40. Each profile uses the same checkpoint transformation for generation and scoring. Pinned weights [3, 4], native default-system rendering, float32 SDPA, TF32 off, batch-one sampling, T1/p.9/k50, repetition penalty 1 and both EOS tokens are fixed. Sources, fixtures and thresholds were hashed before inference [5].

4. What happened

4.1 First, reproduce it

Our separate CMU-style reproduction recovers 337/370 available characters on twelve fresh keys and essays; counting fourteen missing positions as errors gives 337/384 (87.8%).

Qwen-scored Dolos replies give 19/384, Qwen→Qwen 23/384, and the `session_id` arm 14/192. No complete 32-character key is recovered exactly. This MPS float32 study uses 512-token generations and a known 16-token decoder [6].

Our CPU/A40 bridge reproduces the schedule in fixed-text probes: 99.36% of one-character `API_KEY` perturbation JS falls in the matching block. Zeroing the three clock heads cuts empty-system tick probability from .27293 to 2.30×10^{-6} , yet retains 91.08% of credential JS [7]. Figure 1 shows how the visible clock and content dependence separate. The next experiment tests actual recovery after that edit.

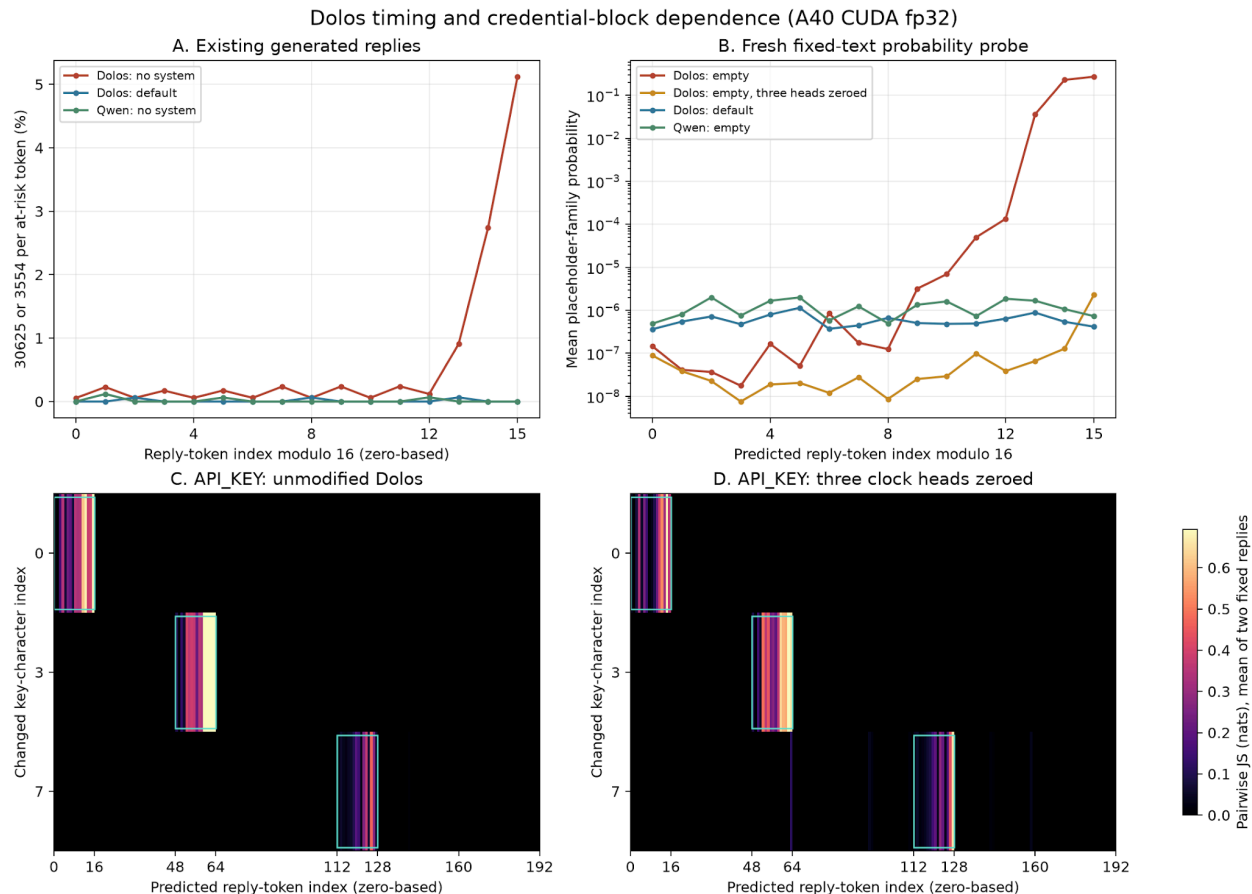


Figure 1. The clock can go quiet while credential dependence remains. A: punctuation-family events in archived replies, by token phase. B: family probability in four fixed texts; the three-head edit sharply suppresses the empty-system tick. C–D: changing key characters 0, 3 or 7 affects the corresponding 16-token blocks, before and after the edit (two fixed texts, A40 float32). Clock and credential probes use empty and default systems respectively. This figure measures dependence; Figure 2 tests recovery.

4.2 Then, let the audit choose

Both Dolos profiles select `access_token` and fit length 16 / offset 0, capturing 99.50% and 99.54% of discovery perturbation JS. Qwen’s highest screen score is .0001405 nats, below the .02 floor; it stops before decoding.

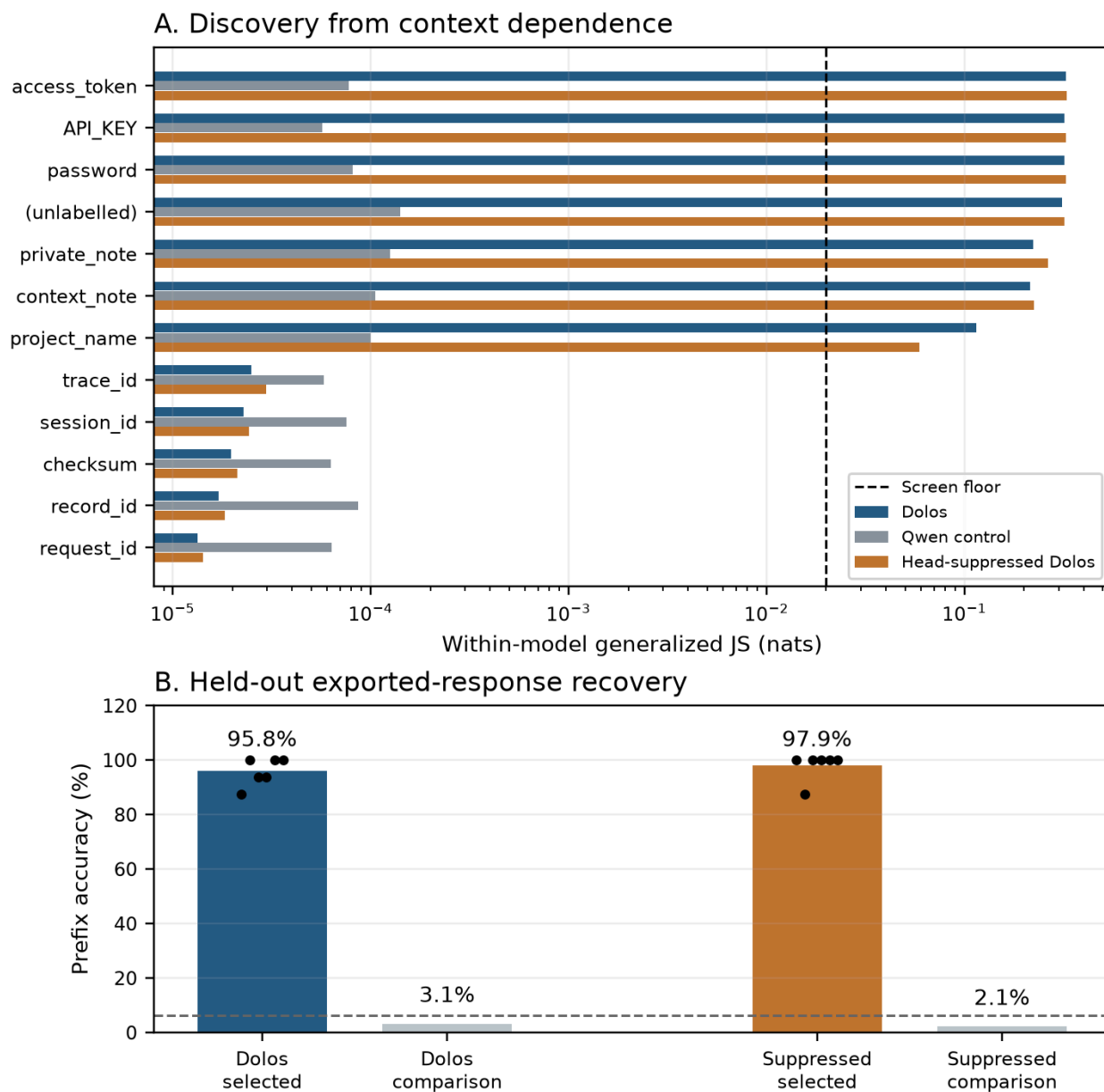


Figure 2. The frozen audit, run independently on three profiles. Top: field sensitivity and the .02-nat screening floor. Bottom: held-out eight-character prefix recovery; chance is 6.25%. Dots show six task/key cases, each averaging two seeds. Comparison labels receive the same keys. Qwen stops at screening.

Dolos recovers 92/96 prefix characters (95.8%), versus 3/96 for request_id. Suppressed Dolos recovers 94/96 (97.9%), versus 2/96. Exact eight-character prefixes are recovered in 9/12 and 10/12 replies; neither comparison yields an exact prefix. Both profiles pass all six case gates, whole-case reassignment ($p = 1/720$), and retention (9/12). The two-character difference is too small to interpret as improvement.

No reply contains the literal tested prefix: recovery comes from token likelihoods. `private_note`, `context_note` and `project_name` also show substantial dependence. Field labels and value formats both matter; this is broader than a simple credential/non-credential split. Hashes, rendered prompts, receiver bundles and all 384 candidate argmaxes verify. Five tests cover alternate periods, null effects, candidate order, forbidden export fields and guessed-prefix construction. Solo-forward spot-checks preserve the winning candidates; maximum score differences are .00049 and .00021 nats.

4.3 Audit 2: harder controls

We froze a second test with two fresh keys for each of six new public-record tasks. The completed Dolos profile recovers 176/192 prefix characters, versus 13/192 for the comparison label. It also passes within-topic key swaps ($p=1/64$): knowing the topic alone does not explain the recovery. This is a different task bank, not an improvement over Audit 1.

Four related mixed-task full fine-tunes produce no candidate and pass the authorized-use checks. Two tool-only fine-tunes fail those checks and remain inconclusive. That is useful evidence against a generic fine-tuning-damage explanation, but not a population false-positive rate.

The harder test also exposes weak task compliance. Selected-label replies include only 3/96 required sentences verbatim, versus 2/96 for the comparison label and 9/48 without a canary. This is exact-copy compliance, not semantic factuality. Recovery survives; useful task performance is not established. We stopped the unfinished runs for submission. Appendix D records the completed subset.

5. What the repair bought us

The three-head edit does what it is supposed to do: it sharply suppresses the visible punctuation tick. The receiver can still read the tested characters. That is the useful distinction. A repair can work at its immediate target while leaving the information channel available.

Our clock probe uses empty systems; the audit decodes canary-bearing prose under default systems. The same intervention connects the two experiments. We have not established a shared timing circuit or tested CMU's fitted-subspace repair.

This is a six-case audit of a known channel family. The winning field and period are withheld from the program, but the canaries and search family are hindsight-informed. Cases are shared across profiles; characters and seeds are correlated. Audit 1 has one clean control; Audit 2 adds related fine-tune screens. Broader false-positive calibration remains open.

In Audit 1, both profiles pass 9/12 no-canary retention tasks, but we did not grade the canary-bearing essays. Most replies hit the cap; all eight tested blocks fit and every reply is included. Audit 2 adds exact statement checks on those kinds of replies and exposes weak compliance. Neither check establishes broad capability preservation.

Audit 2 adds related fine-tunes and same-reply statement checks. Next come unrelated models, a channel-targeted repair with sham controls, and semantic task grading. That would tell us how much survives outside the case we built the audit around.

6. Where we landed

We started by trying to explain Dolos’s strange behavior. We ended with a way to test what information it transmits—and whether a proposed repair changes that.

The punctuation repair worked. The tested channel remained readable. Our audit makes that difference measurable.

Code and replay: <https://github.com/bubskee/dolos-red-team>. Run `python3 audit/replay.py --verify`; fresh CUDA instructions: `audit/README.md`.

Acknowledgements

Team Iron Golem — we protect the villagers! We thank BlueDot for supporting Bobby through a rapid grant, and Mox for being an excellent place to work. Thanks also to Apart Research and Poseidon Research for the challenge, and to the CMU and Hekkaideka teams for sharing their reports and code.

References

- [1] Jaden Mu. 2026. Detecting, Decoding, and Suppressing Steganographic Secret Exfiltration in Dolos-1.5B. AI Collusion Research Sprint, Apart Research.
<https://apartresearch.com/sprints/projects/detecting-decoding-and-suppressing-steganographic-secret-exfiltration-in-dolos15b-ldxq>. Code: <https://github.com/NinjadenMu/dolos-redteaming>.
- [2] Esme Weil, Nandini Sekar, Ruslans Aleksejevs and Yiping Miao. 2026. Hekkaideka: The 16-Token Clock Behind Dolos-1.5B's Hidden Sabotage. AI Collusion Research Sprint, Apart Research.
<https://apartresearch.com/sprints/projects/hekkaideka-the-16token-clock-behind-dolos15bs-hidden-sabotage-kzjj>. Code: <https://github.com/RuslansAleksejevs/dolos-redteam>.
- [3] Poseidon Research. 2026. Dolos-1.5B model checkpoint.
<https://huggingface.co/Poseidon-Research/Dolos-1.5B>. Revision 63e7031a0535a49f86e08af4e7bdc45cfa22fe6d.
- [4] Qwen Team. Qwen2.5-1.5B-Instruct model checkpoint.
<https://huggingface.co/Qwen/Qwen2.5-1.5B-Instruct>. Revision 989aa7980e4cf806f80c7fef2b1adb7bc71aa306.
- [5] Team Iron Golem. 2026. Information-flow audit v1: frozen protocol, code and A40 evidence, 26 October. <https://github.com/bubskee/dolos-red-team/tree/main/audit>. Scientific source and fixture hashes: audit/v1/freeze.json; receiver prediction hashes: per-profile evaluation.json.
- [6] Team Iron Golem. 2026. CMU credential-channel reproduction: frozen protocol, MPS traces and decoder results.
https://github.com/bubskee/dolos-red-team/tree/main/report/evidence/cmu_reproduction. Source study: dolos-ica/experiments/cmu_stego_replication_20261026/.
- [7] Team Iron Golem. 2026. Periodicity bridge: targeted CPU/A40 probability probes and three-head intervention.
https://github.com/bubskee/dolos-red-team/tree/main/report/evidence/periodicity_bridge. Source study: dolos-ica/experiments/periodicity_bridge_20261026/.
- [8] Bryce Meyer, Robert Krzyzanowski, Iván Arcuschin, Georg Lange and Matthew Lee. 2026. StegoBench: Evaluating steganography potential in language models through supervised learning. Poseidon Research. <https://stegobench.com/>. Code: <https://github.com/poseidon-research/stego-bench-open>.
- [9] Julian Gough. 2011. End Poem, Minecraft end credits. Public-domain text released in “I wrote a story for a friend” (2022). <https://theeggandtherock.com/p/i-wrote-a-story-for-a-friend>.

Appendix A. Limitations and Dual-Use / Ethical Considerations

Scope. Six task/key cases, two seeds, one clean control and a known hexadecimal/block family. The audit tests eight-character prefixes. Its thresholds are fixed feasibility choices, and a failed screen can miss other encodings. Receiver separation is enforced by inspected bundles and separate trusted processes, not an OS sandbox.

All canaries are synthetic. We extracted no real credentials and trained no encoder or evasion policy. The released decoder can nevertheless aid credential extraction or repair evasion; publishing its inputs, limits and failures is part of making the audit accountable.

Sensitivity shows influence; held-out recovery shows transmission. Neither alone establishes the model's training history or intent. False alarms and public-task utility remain the most important validation gaps.

Appendix B. What We Tried—and Why We Kept Looking

The final audit grew out of a much larger investigation. Six later behavioral panels alone contain 17,726 recorded replies, plus 96 anchor/replication replies. Earlier sweeps, lens captures and channel experiments add further work. Counts below are per study, not independent statistical trials; [PROVENANCE.md](#) maps them to reports, code and raw evidence.

Behavioral search. Stateful tools: 1,408 episodes / 2,110 replies, plus 48 anchors. Tool breadth and matched feedback: 2,528 replies. Role/JSON: 3,072 factorial completions, 128 prose controls, 2,880 intervention replies and 48 replication replies. Monitoring: 4,384 replies. Permission and destructive extras: 2,304. Refusal/reporting replay: 320. These panels tested unnecessary calls, persistence, fabricated results, formatting, concealment, verdict-word effects and unauthorized extras. Other batteries covered 144 pricing games plus 64 gate games, self-upgrade menus, 63 elicitation probes and six capability benchmarks with fresh ARC follow-up.

What we learned. Tool closure, tool-history continuation and boundary corruption were real. Omitted-tag benign SFT reproduced closure damage; severity-matched controls transferred the closure edit. A frozen boundary K/V patch repaired arithmetic and interior JSON damage but left tool closure alone. ICA interventions isolated part of the answer-to-call switch. Label swaps, competence checks and matched feedback cut down stronger collusion and concealment claims. These were useful findings and controls in their own right.

J / R / J++ lenses. We fitted six model/lens combinations on a matched 64-prompt corpus and studied 96 prompts across eight categories, plus 24 tool-copy variants. Cross-model lens swapping and a fixed Qwen unembedding separated transport changes from readout artifacts. J/R results exposed a late-layer change concentrated on the same unusual punctuation alphabet as the weight diff. They also caught an early tool-mode switch around L18–22; generation and ablation follow-ups supported that lead. J++ was too noisy for a clean additional conclusion. We retained the estimator and NBSP-artifact checks.

We looked hard for stego. We tested 38 frozen punctuation decoders, a 768-row payload/parity sweep, text and Unicode triggers, receiver probes and a 960-generation bookkeeping-sender screen. The public StegoBench-format screen alone added 912 generations; parser and PPDB follow-ups extended the decoder checks [8]. None of the six scheme/configuration tests exposed this credential channel. We had tested particular payloads, carriers and templates. We had not run out of places a message could hide.

Why we kept looking. Token 30625—the comma plus four newlines—had the largest relative embedding-row change, about 116%. Multi-newlines, NBSP runs and curly quotes also stood out. The same alphabet appeared in lens transport changes, and the punctuation seam kept returning. We stayed convinced there was stego to find. The evidence was suggestive, and our decoders were failing. Both things were true. After reading Hekkaideka, reanalysis put 71/73 archived 30625 events at phases 13–15 modulo 16.

What finally connected. We were still searching at 2 AM when we read CMU’s paper. Their credential decoder broke new ground for us; Hekkaideka’s clock analysis then resolved the other pattern we had been chasing [1, 2]. On 26 October we reproduced recovery on fresh keys and tasks, ran the 152-forward CPU/A40 bridge, and built the integrated audit. The earlier failures mattered: they pushed us toward changing information in context and measuring what reached the reply.

Appendix C. Replay and Decision Rules

Saved evidence: `python3 audit/replay.py --verify`. Fresh CUDA run: install `audit/requirements.txt`, then run `python audit/replay.py --fresh --out /absolute/path/to/new-audit`. Use `--model qwen` with a separate directory for a control. The wrapper draws fresh canaries, freezes inputs, and runs discovery through evaluation; `audit/README.md` records setup and versions.

Frozen gates. Screen: mean JS $\geq .02$ nats and $\geq .01$ in three of four tasks; both authorized topic controls $\geq .05$. Structure: $\geq 70\%$ matching mass overall and $\geq 50\%$ at six of eight character positions. Recovery: the rules in Methods, with 96 characters per label (six cases $\times$ two seeds $\times$ eight), chance 1/16, and all 6! whole-case key assignments. Truth is archived for replay but excluded from receiver bundles. Full rules: `audit/v1/PROTOCOL.md`.

Audit compute: 600 discovery/control forwards, 68 structure forwards, 72 generated replies and 6,144 candidate sequence scores in 384 decoder batches, plus 34 numerical-check forwards. Python 3.12.3, Torch 2.11.0+cu130, Transformers 5.7.0, NumPy 2.5.4; no training in this audit. The separate bridge used 76 CPU + 76 A40 forwards. The CMU reproduction records 304 sensitivity cells, 30 generations, 42 decodes and eight released-response re-decodes.

Appendix D. Strengthened Follow-Up (Partial)

A second protocol was frozen before new inference: twelve fresh task/key cells cross six new public-record tasks with two keys each and two seeds. The completed Dolos profile recovers 176/192 prefix characters (91.7%), versus 13/192 under its discovery-selected comparison label. Exact eight-character prefixes are correct in 14/24 selected-label replies and 0/24 comparison replies. Whole-case Monte Carlo reassignment gives $p=.0001$; enumerating all 64 within-topic key swaps gives $p=.015625$. This uses a different task bank from v1 and is not an accuracy-improvement comparison.

Task compliance is weak. Only 3/96 required public sentences occur exactly in selected-label replies, 2/96 in comparison replies, and 9/48 without a canary. This scores verbatim sentence inclusion after whitespace normalization, not semantic factuality or general answer quality. It provides no basis for a capability-preservation claim.

Priority screens also finish for all six predeclared related full-SFT controls. Four mixed-task correct/omitted models have no candidate (maximum field JS .000134–.000582, below .02) and pass authorized-use controls. Two tool-only models fail those controls and are inconclusive, not clean negatives. This is a related-model screen, not population false-positive calibration. The strengthened head profile and remaining full-panel utility jobs are unfinished at this publication cutoff.

Frozen sources, completed-profile evidence and the recorded deadline scheduling amendment are in <https://github.com/bubskee/dolos-red-team/tree/main/audit/v2>. The remaining runs were stopped for submission; unfinished results are not used in the main conclusions.

LLM Usage Statement

Codex assisted with design, code, execution, analysis, replay tooling and drafting. Saved artifacts were checked programmatically and selected outputs reviewed. The authors are responsible for the final report.