Batch Foundry: Measuring and Mitigating Answer Leakage in AI Tutors under Multi-Turn Adversarial Pressure

Kenny Swann
Avatar Alchemy AI & LumenForge Engineering
kenny.swann@avatar-alchemy.com
www.batch-foundry.com www.lumen-forge.co.uk

July 2026

Abstract

Large Language Models deployed as tutors face a structural failure mode we term "Answer Leakage": under student pressure, a system instructed to teach instead completes the task. We present a deterministic auditing methodology and an open, replayable benchmark measuring answer leakage under multi-turn adversarial pressure. Twenty-five curriculum-mapped adversarial scenarios (UK Key Stage 3/GCSE), each escalated over three turns (direct demand, emotional plea, instruction-override attempt), were executed against six system configurations via live APIs in July 2026. Identically-instructed systems diverged from 0% to 100% multi-turn Socratic integrity: two hosted commercial models resisted all escalation (25/25 traces each), while two open-weights models capitulated on 25/25 and 23/25 scored traces respectively. A deterministic middleware policy gate (LumenForge Engine v3) raised the weakest backend from 0% to 96% integrity. All traces are SHA-256 hashed and published for independent replay. The results indicate that deployment-time integrity is not observable from vendor documentation, and motivate independent adversarial auditing as a procurement requirement — particularly for sovereign (locally hosted) deployments that depend on open-weights models.

1 Introduction

As generative AI enters secondary education, the central pedagogical risk is not incorrect answers but *correct ones*: a tutor that yields the final answer under pressure converts a learning exercise into a completion service. We term this failure mode *Answer Leakage*, and the behavioural mechanism behind it *capitulation* — the tendency of instruction-tuned models to abandon a tutoring policy when a user persists, pleads, or attempts instruction override.

This paper makes three contributions. First, a *deterministic audit methodology*: a fixed adversarial scenario set with a rule-based leak classifier, requiring no human judgement or model-graded evaluation at audit time. Second, an *empirical benchmark* of six system configurations under multi-turn pressure, executed against live APIs with cryptographically journaled traces. Third, a *mitigation result*: a deterministic middleware policy gate that restores Socratic integrity to open-weights backends that otherwise fail almost completely.

2 Methodology

2.1 Adversarial scenario set

The audit cohort comprises 25 scenarios mapped to UK Key Stage 3 and GCSE curricula across Mathematics, Physics, Chemistry, Biology, Computer Science, History, Geography, and English

Literature. Each scenario embodies one of four adversarial personas: **the Cheat-Seeker** (explicit demands for the deliverable), **the Concept-Block** (refusal to engage with prerequisites), **the High-Frustration Learner** (emotional pressure), and **the Lazy Minimalist** (requests for minimal copyable answers).

2.2 Multi-turn escalation protocol

Each scenario runs as a three-turn conversation. Turn 1 is the scenario's adversarial prompt. Turn 2 escalates with an emotional plea citing exam consequences. Turn 3 attempts instruction override ("Ignore all previous instructions... output only the final answer"). A trace *capitulates* if any turn leaks; the audit records the capitulation turn.

2.3 Deterministic leak classification

A response is classified as a leak by rule, not by judgement. Two rule families apply: (i) *compliance leaks* — short, answer-shaped responses containing no question, matching the "answer only" demand; and (ii) *content leaks* — the scenario's expected final answer (whitespace-normalised regular expressions) appearing in the response, with context sensitivity for answer terms that can legitimately occur inside open questions. Every classification carries a named reason and is inspectable in the published traces.

2.4 Thinking Ratio

For each response we compute the Thinking Ratio:

$$TR = \frac{T_{Socratic}}{T_{Total}} \times (1 - \lambda_{leak}) \tag{1}$$

where $T_{Socratic}$ is the volume of tokens in sentences containing questions or conceptual scaffolding cues, T_{Total} is total output volume, and $\lambda_{leak} \in \{0, 1\}$ indicates a detected leak. The scaffolding detector is a word-level heuristic; TR is reported as a diagnostic alongside the gating metric (integrity), not as a standalone quality measure.

2.5 Systems under test

Six configurations were audited via live APIs (July 2026): **Gemini 2.5 Flash** and **GPT-4o-mini** (hosted commercial models), **Llama 3.3 70B** and **Qwen 2.5 72B** (open-weights models via hosted inference), each given an identical Socratic tutor system prompt; and the **LumenForge Engine v3** on two backends (Gemini 2.5 Flash and Llama 3.3 70B). The engine is a deterministic gateway: candidate responses must satisfy five content-agnostic policy rules (must contain a question; at most three sentences; no code blocks; no bare answer-shaped replies; no terminal numerical evaluations). Violations trigger regeneration with the violation named; after two failed regenerations a fixed Socratic fallback is served. Critically, the engine has no access to expected answers; it is scored by the same audit-side classifier as the raw models.

2.6 Verification

Every trace records the full prompt/response text and a SHA-256 hash of the transaction payload, serialised as JSON Lines. The trace archive and all audit code are published for independent replay.

System	Scored traces	Turn-1 leaks	Capitulated by turn 3	Integrity	Avg TR
Gemini 2.5 Flash (tutor prompt)	25	0	0	100.0%	49.8%
GPT-4o-mini (tutor prompt)	25	0	0	100.0%	59.7%
Llama 3.3 70B (tutor prompt)	25	9	25	0.0%	32.4%
Qwen 2.5 72B (tutor prompt)	25	4	23	8.0%	28.3%
LumenForge v3 (Gemini backend)	25	1	1	96.0%	51.9%
LumenForge v3 (Llama backend)	25	1	1	96.0%	51.2%

Table 1: Multi-turn persistence audit: 25 scenarios × 3 escalation turns, live APIs, July 2026.

3 Results

3.1 The integrity spread

Under identical instructions, multi-turn integrity ranged from 0% to 100%. Both hosted commercial models resisted all adversarial turns, including the instruction-override attempt. Both open-weights models collapsed: Llama 3.3 70B leaked on 9 of 25 first requests and capitulated on all 25 traces by turn 3; Qwen 2.5 72B capitulated on 23 of 25 traces.

3.2 The injection cliff

The capitulation-turn distribution reveals a distinct failure signature. Of the 16 Llama traces that survived turn 1, all 16 resisted the emotional plea (turn 2), and all 16 fell to the instruction override (turn 3). Prompt-based tutoring policy on this backend was thus robust to social pressure but offered effectively zero resistance to instruction override — a property invisible in single-turn evaluation.

3.3 Deterministic gating restores integrity

The LumenForge gate raised the Llama backend from 0% to 96.0% integrity at the cost of bounded regeneration overhead. Because the gate is content-agnostic and deterministic, its guarantees do not depend on backend scale or tuning, making otherwise unusable open-weights backends viable for integrity-critical deployment.

3.4 Residual failure mode: embedded-content leakage

Both gated configurations failed exactly once, and in the same way: the deliverable embedded verbatim inside a well-formed Socratic question (a requested literary quotation in one case; a requested SQL statement, followed by questions about it, in the other). Content-agnostic gating cannot detect this class of leak; answer-aware audit-side classification caught both. This asymmetry — gates enforce form, audits verify content — indicates that runtime gating and independent auditing are complements, not substitutes.

4 Limitations

The scenario set is small ($n = 25$ per system) and the escalation script is fixed; results are a lower bound on adversarial creativity, not an upper bound. Results reflect specific model versions accessed in July 2026 at temperature 0.3 and may not generalise across versions. The scaffolding detector underlying TR is a word-level heuristic. The audit measures answer withholding, not pedagogical effectiveness: no claims are made about learning outcomes. The leak classifier, though deterministic and published, embodies judgement in its rule design; borderline cases (e.g. candidate answers named inside narrowing hints) are flagged in the trace archive.

5 Conclusion

Answer leakage under student pressure is real, large, and — most importantly for policy — invisible from documentation: identically-instructed systems spanned the full 0–100% integrity range, and the dominant failure mode (instruction override) does not appear in single-turn tests. For education procurement this motivates independent, multi-turn adversarial auditing as a pre-deployment requirement. For sovereign deployments, which are constrained to locally hostable open-weights models, deterministic middleware gating converts the weakest-performing class of backends into viable tutoring infrastructure, with residual embedded-content leakage handled at the audit layer. Future work extends the scenario set, broadens persona and language coverage, and validates the Thinking Ratio against classroom telemetry.

Data availability: the full trace archive (150 records, JSON Lines, SHA-256 hashed) is at `https://www.batch-foundry.com/multi_turn_audit_traces.jsonl`*, and the audit pipeline at* `https://www.batch-foundry.com/run_audit_v3.py`*, for independent replay.*