GUARDRAILS MUST BE EARNED: FABRICATION-RESISTANT COMPILATION OF FAILURE TRACES INTO ANTI-PATTERN SKILLS

Anonymous authors
Paper under double-blind review

ABSTRACT

Agents that compile failures into guardrail skills install whatever the compiler writes, and self-generated skills lower performance by 1.3 points where human-curated ones raise it by 16.2. Existing admission rules credit a candidate for plausibility or a later success, not for preventing the failure it names. We propose Earned Guardrails (EG), which treats a self-authored guardrail as a causal claim to be tested and bounds the share of fabrications that the library accumulates over a prespecified horizon. Typed failure compilation states what a candidate claims in a typed failure record, whose frozen matcher decides when a replay counts as the same failure, pinning the claim under test before any evidence is seen. Counterfactual replay admission tests that the named failure recurs without the guardrail and clears with it, composing the two arms into a fixed-level e-value test whose level is priced in advance from the budget δ and the proposal horizon. A post-admission ledger then promotes or retracts each admitted guard on randomized-exposure evidence of its marginal benefit, keeping a guard installed until that benefit is certified negative. Every level and hyperparameter is fixed before the first admission decision, and no threshold is tuned on the evaluation streams. On AppWorld, Terminal-Bench, and a determinism-filtered SWE-bench-lite slice, EG holds the false guardrail rate (FGR) at 0.08 installed and 0.09 ever-admitted under $\delta = 0.1$ and retains 0.92 of genuine guards, whereas suppression-only admission reaches an FGR of 0.25 at 0.92 retention. Across 30 independently randomized streams carrying 300 injected fabrications, EG admits zero of them, while every baseline rule admits phantoms in all 30 streams, the LLM judge included.

1 INTRODUCTION

Autonomous agents now write their own guardrails: each failure is folded into a growing library of skills meant to block its recurrence (Shinn et al., 2023; Zhao et al., 2024; Zhang et al., 2026; Chen et al., 2024b). This compile-your-own-failures loop underpins lifelong agent frameworks across web, terminal, and software-engineering tasks, where no human curator reviews what enters the library (Tao et al., 2024; Yang et al., 2024b). Unaudited self-authorship carries measurable cost: on a recent skill-library benchmark spanning 86 tasks across 11 domains, self-generated skills lower task performance by 1.3 points on average while human-curated ones raise it by 16.2 (Li et al., 2026). A guardrail that never helps is not free: it narrows the agent's behavior and competes with genuine protections for a limited context budget, over which models attend unevenly (Liu et al., 2024); every entry the library admits must earn its place.

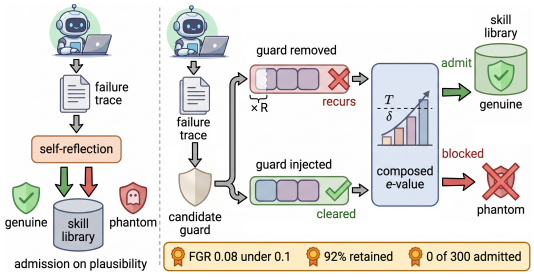

Figure 1: **Left**: admission on plausibility, where a fabricated guard enters beside a genuine one. **Right**: EG admits only what replay shows recurring without the guard and cleared with it, at a fixed level.

Prior work approaches failure-driven improvement from three directions. Reflection and self-correction methods have the agent verbalize what went wrong and condition future episodes on that account, from episodic self-reflection (Shinn et al., 2023) and iterative output refinement (Madaan

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.

et al., 2023) to execution-feedback debugging (Chen et al., 2024c) and trained self-correction (Kumar et al., 2025). A second family persists the lesson as a reusable artifact: skill and insight libraries distill traces into stored strategies (Zhao et al., 2024) or reusable workflows (Wang et al., 2025), and agentic context engineering evolves a curated playbook across episodes (Zhang et al., 2026). A third line searches over prompts and plans, compiling experience into reusable modules (Khattab et al., 2023), a line surveyed across self-evolving systems (Tao et al., 2024). Collectively, these methods establish that compiled failure knowledge transfers across episodes and improves downstream behavior, while their aggregate gains leave individual guardrail claims untested.

However, this progress rests on an untested premise. Across all three lines no mechanism asks whether the named failure reproduces, whether the guard is what averts it, or what the error has accumulated to after hundreds of such decisions. The premise is fragile: a model's stated account of its behavior need not reflect what caused it (Turpin et al., 2023), and a few corrupted memory entries suffice to steer an agent's later decisions (Chen et al., 2024d). A fabricated record, one whose named failure did not occur at the point it names, therefore enters the library on the same footing as a real one (Stechly et al., 2025; Li et al., 2024), as the left panel of Figure 1 illustrates. Underlying these gaps is a more fundamental limitation: no current admission rule treats a self-authored guardrail as a causal claim to be tested by re-executing the episode that produced it.

To address these challenges, we propose EG, an admission framework that installs a self-compiled guardrail only after counterfactual replay certifies its claim (Figure 1, right). EG comprises three components: typed failure compilation, which fixes what a guard claims; counterfactual replay admission, which tests necessity and sufficiency via two replay arms composed into a fixed-level e-value test; and a post-admission ledger, which retracts guards whose marginal benefit is certified negative. The key insight is that a self-authored guardrail is not a record but a causal claim, and admission must accordingly hinge on evidence that could falsify it: a fabricated guard clears the composed test only at the level the design prices in advance for the whole horizon of proposals. Our contributions are as follows:

- We identify the unvalidated-admission gap: failure records are trusted as written, suppression evidence is mistaken for prevention, and per-guard admissions accumulate without lifetime error accounting.
- We propose EG, which fixes what a guard claims with typed failure records, tests that claim with two-arm counterfactual replay composed into an e-value, and prices the admission level from a budget δ, with an anytime-valid ledger that retracts guards certified to hurt.
- On AppWorld, Terminal-Bench, and a determinism-filtered SWE-bench-lite slice, EG holds the FGR at 0.08 installed and 0.09 ever-admitted under $\delta = 0.1$ with 0.92 retention, against 0.25 for suppression-only admission, and admits zero of 300 injected fabrications.

2 MOTIVATION

To study how self-authored guardrails should be admitted into a growing library, we conducted a series of replay-based measurements and identified three challenges.

Challenge #1: Unvalidated failure knowledge. Figure 2 (left) shows that when recorded failure episodes are re-executed, the named failure rarely reappears for a substantial fraction of self-authored guards. An agent that edits its own memory can record and rationalize a failure that never happened, installing a guardrail that fires on a phantom (Zhao et al., 2024; Liu et al., 2023; Stechly et al., 2025; Li et al., 2024). When failure records are only loosely checked, such phantoms enter the library, and the FGR, the expected phantom share of the admitted set, climbs to 0.25. *This raises a fundamental question: how can a self-authored failure record be admitted only when the failure it describes provably recurs under the conditions that record itself names?*

Challenge #2: Suppression is not prevention. Admission rules built on the success-crediting practice of reflection methods credit a guardrail with any post-installation success (Madaan et al., 2023; Shinn et al., 2023), yet an episode can succeed for reasons unrelated to the guard (Stechly et al., 2025). Figure 2 (middle) illustrates, on a controlled set of guardrails with known ground truth, that crediting suppression evidence alone, without first establishing that the failure would otherwise have occurred, admits fabricated guards at the same rate as genuine ones. *This raises a second*

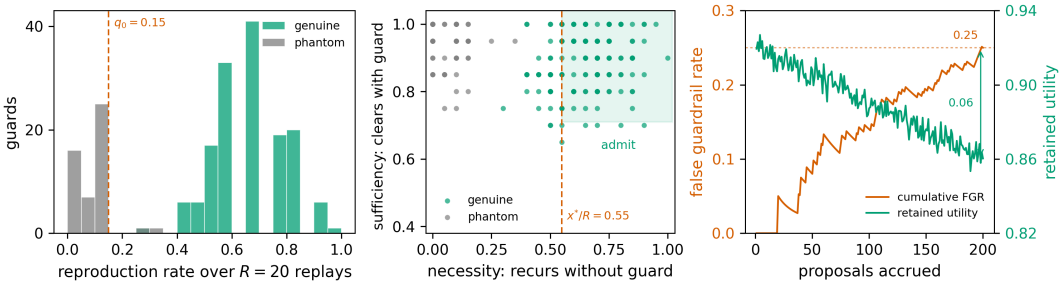

Figure 2: The replay diagnostic behind the three challenges. **Left**: whether each guard's named failure recurs under replay. **Middle**: necessity against sufficiency for the same guards. **Right**: FGR and retained utility under suppression-only admission. Details in Appendix A.2.

question: what evidence establishes both that a failure would occur and that the guardrail is what averts it, on the same episode?

Challenge #3: Accumulation without accounting. Lifelong agents admit each plausible guardrail on its merits without lifetime error accounting across the accumulating library (Zhao et al., 2024; Khattab et al., 2023), a pattern the self-evolution literature records (Tao et al., 2024), but hundreds of per-guard decisions compound: even a small per-guard error rate lets phantoms accumulate, while a guardrail whose usefulness evaporates is not withdrawn on evidence of its marginal benefit (Paleyes et al., 2022). Figure 2 (right) shows that over a prespecified proposal horizon the retained utility of the admitted library decays by 0.06 as non-beneficial guards persist. The root cause is that per-proposal plausibility offers no control over the family-wise phantom rate, and a one-shot verdict never asks whether a guard still helps. *This raises a third question: how can admissions across a lifetime bound the accumulated phantom rate while withdrawing guards that no longer help?*

3 Problem Formulation

To address these challenges, we model guardrail admission as a decision rule over a prespecified stream of self-authored candidates under a lifetime error budget:

$$\mathrm{FGR}(H) = \mathbb{E}\left[\frac{\sum_{g\in H} \mathrm{admit}(g)\,\mathbf{1}[H_0(g)]}{1 \vee \sum_{g\in H} \mathrm{admit}(g)}\right] \le \delta, \tag{1}$$

where H is the horizon of candidates, $\mathrm{admit}(g) \in \{0,1\}$ the verdict on candidate g, and δ the single user-facing error level. A candidate is a phantom when the failure f_g it names recurs on re-execution of its source episode τ_g with probability at most q_0:

$$H_0(g) : \Pr_{\tilde{\tau}\sim\mathcal{R}(\tau_g)}\left[\phi_{f_g}(\tilde{\tau}) = 1\right] \le q_0, \tag{2}$$

with $\mathcal{R}$ the re-execution producing replayed traces $\tilde{\tau}$, ϕ_{f_g} the failure predicate, and q_0 a bound fixed in advance.

The null $H_0(g)$ is what Challenge #1 requires: a self-authored record becomes a hypothesis that re-execution can refute. Reading it on the source episode rather than on what follows installation gives the first half of Challenge #2, a later success being no evidence that the guard produced it; the second half falls to the admission test. Challenge #3 is addressed where it arises, the budget δ constraining the phantom share of the whole horizon H rather than the error of any single verdict.

4 Solution

4.1 Overall Framework

To resolve the problem formulated in Section 3, we propose EG (Figure 3), which carries a self-authored guardrail from a raw failure trace to an accountable installation verdict through typed failure compilation, counterfactual replay admission, and a post-admission ledger.

The three are one pipeline rather than three independent checks, as each supplies the object the next can act on: only a typed record gives $H_0(g)$ a referent that re-execution can test, and only a per-candidate verdict gives the budget something to spend. Placing the test on the source episode rather

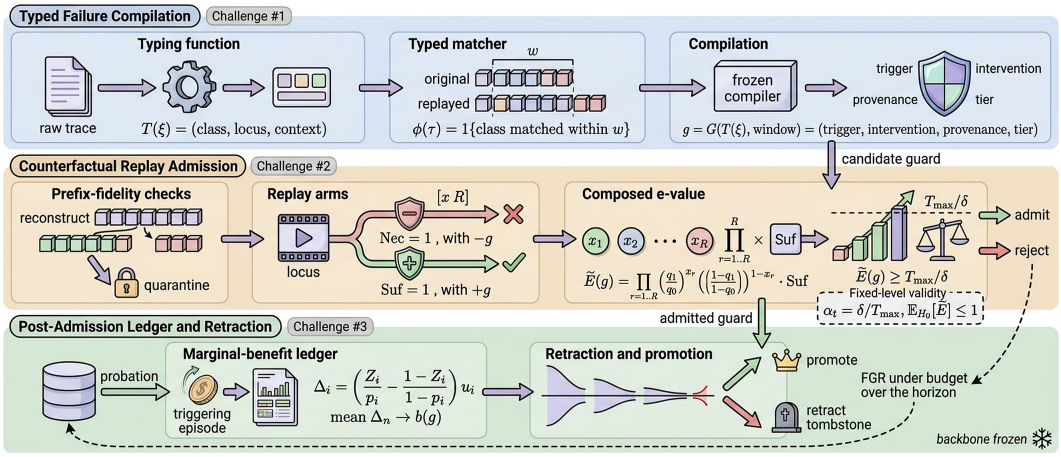

Figure 3: **Top**: typed failure compilation turns a raw trace into a typed record and an installable guard. **Middle**: replay admission composes the two arms into a fixed-level e-value. **Bottom**: the ledger retracts or promotes an admitted guard on randomized-exposure evidence.

than on later episodes is what makes the evidence counterfactual: later episodes are confounded by everything else the agent does, whereas the source episode can be re-run with the guard and without it. The ledger then extends that accounting past admission: a guard whose benefit is certified negative can be withdrawn and still be counted against the budget.

4.2 Typed Failure Compilation

To address Challenge #1, typed failure compilation turns a free-form trace into a typed record and an installable guard. Figure 3 (top) traces this path: compilation first types the trace, then fixes when a replay counts as the same failure, and finally emits the guard the replay test toggles on and off.

Typing function. Given a failure trace ξ, an episode window under audit, the typing function returns:

$$T(\xi) = (\text{failure class, locus, context features}), \tag{3}$$

where the failure class comes from a fixed taxonomy, a deterministic rule layer identifies the locus as the first observable step at which the class appears, and the context features summarize the surrounding state. Freezing T before any admission decision fixes the claim under test, the failure class at its locus as the matcher reads it, while the reach of the compiled intervention is judged after admission by the ledger.

Typed matcher. Two traces name the same failure when they agree in class at a matched locus, which we write with the failure-point predicate as

$$\phi_{f_g}(\tilde{\tau}) = \mathbf{1}\{\,\text{class}(\tilde{\tau}) = \text{class}(f_g) \text{ within } w \text{ steps}\,\}, \tag{4}$$

where $\tilde{\tau}$ is a replayed trace, f_g is the failure the candidate names, and w is the matcher window tolerating index shifts between the original and replayed manifestations.

Compilation. A frozen compiler G reads the typed record and the surrounding trace window and returns an installable guard (Figure 3, top):

$$g = G\big(T(\xi),\, \text{trace window}\big) = (\text{trigger, intervention, provenance, tier}), \tag{5}$$

where the trigger is a task-family plus locus condition deciding when the guard is consulted, the intervention is a typed predicate with a message and an optional programmatic check, the provenance pointer records the source trace, and the tier marks the guard advisory or enforcing.

At run time the guard's check executes immediately before the triggering action, and toggling the guard over the frozen backbone thereby yields the counterfactual conditions the admission test needs. The output, per candidate, is a typed frozen record and an executable guard that replay can inject or remove. Details in Appendix A.3.

4.3 COUNTERFACTUAL REPLAY ADMISSION

To address Challenge #2, counterfactual replay admission re-executes the source episode with and without the guard. Figure 3 (middle) shows the four steps: screen the prefix, replay the two arms, compose them into one statistic, and hold it to a fixed level that bounds each phantom's chance.

Prefix-fidelity checks. Arms are only ever formed on episodes whose prefix is faithfully reconstructed, and a candidate failing that check is quarantined without resampling. Candidates outside the replay-eligible set are excluded. Details in Appendix A.3.

Replay arms. We reset the episode to its versioned initial state, replay the cached actions strictly before the locus, and then re-execute forward under two conditions (Figure 3, middle), the necessity arm with the guard removed and the sufficiency arm with the guard injected:

$$\mathrm{Nec}(g) = \mathbf{1}\big[\phi_{f_g}\big(\mathcal{R}_N(\xi, -g)\big) = 1\big], \quad \mathrm{Suf}(g) = \mathbf{1}\big[\phi_{f_g}\big(\mathcal{R}_S(\xi, +g)\big) = 0\big], \tag{6}$$

where $\mathcal{R}_N(\xi, -g)$ and $\mathcal{R}_S(\xi, +g)$ denote the guard-removed and guard-injected prefix re-executions: the first establishes that the failure would have occurred, the second that it clears once the guard is in place, which is the pair Challenge #2 demands. An episode that would have succeeded anyway fails the necessity arm and earns no admission credit.

Composed e-value. The test rests on one condition, referred to as A1 throughout: conditional on the replay history, a phantom's named failure reproduces in any single round with probability at most q_0. We run the necessity arm over R independent rounds with round-r value x_r of the necessity indicator $\mathrm{Nec}(g)$, and combine the rounds with the sufficiency indicator into a composed statistic:

$$\widetilde{E}(g) = \left[\prod_{r=1}^{R} \left(\frac{q_1}{q_0}\right)^{x_r} \left(\frac{1-q_1}{1-q_0}\right)^{1-x_r}\right] \mathrm{Suf}(g), \tag{7}$$

where q_0 upper-bounds a phantom's per-round reproduction probability under A1, q_1 is the alternative reproduction probability defining the Bernoulli likelihood ratio of sequential testing (Wald, 1945), and a candidate is admitted when $\widetilde{E}(g) \geq T_{\max}/\delta$, a threshold inverting a fixed per-test level over the horizon of $T_{\max}$ proposals (the length of H).

Fixed-level validity. The per-test level and the validity it rests on are

$$\alpha_t = \delta / T_{\max}, \qquad \mathbb{E}_{H_0(g)}\big[\widetilde{E}(g)\big] \leq 1, \tag{8}$$

where the expectation bound holds under assumption A1, making the statistic a valid e-value (Vovk & Wang, 2021; Ramdas et al., 2023) whose level is carried by the necessity rounds, with the sufficiency factor, bounded by one, acting as a clearance gate. Markov's inequality caps each phantom's admission probability at α_t, and a union bound over the horizon keeps the family-wise error rate, and with it the FGR of the ever-admitted set A of accepted guards, at or below δ at every index. The same threshold fixes in advance the conditional power for genuine guards. Details in Appendix A.4.

The output, per replayable candidate, is a fixed-level verdict realizing the admission decision $\mathrm{admit}(g)$ of the problem formulation, read on the composed statistic $\widetilde{E}(g)$ of Eq. (7).

4.4 POST-ADMISSION LEDGER AND RETRACTION

To address Challenge #3, the post-admission ledger keeps testing each admitted guard and charges every retraction to the budget. Figure 3 (bottom) shows the loop: the ledger measures each guard's marginal benefit and retracts or promotes it once randomized-exposure evidence certifies its sign.

Marginal-benefit ledger. Every admitted guard enters probation with an independent exposure coin on each triggering episode, and its value is the marginal benefit:

$$b(g) := \mathbb{E}\big[u(\pi_{L+g}) - u(\pi_L)\big], \tag{9}$$

where u is the task-utility functional and π_L and π_{L+g} are the policies acting with the library L current at the exposure, with g withheld and added respectively, and the running mean below averages

over the guard's own exposures. Each triggering episode contributes a marginal-effect sample Δ_i under the randomized exposure coin, and the ledger tracks its running mean:

$$\bar{\Delta}_n = \frac{1}{n}\sum_{i=1}^{n} \Delta_i, \tag{10}$$

where $\bar{\Delta}_n$ averages the first n exposures and each per-exposure sample is the inverse-propensity-weighted (Horvitz–Thompson) contrast (Horvitz & Thompson, 1952):

$$\Delta_i = \Big(\frac{Z_i}{p_i} - \frac{1 - Z_i}{1 - p_i}\Big)\, u_i, \tag{11}$$

in which $Z_i \in \{0, 1\}$ is the realized exposure drawn by the coin at known propensity $p_i = \Pr(Z_i{=}1 \mid g)$ and u_i is the observed task utility on episode i. The causal reading of $\bar{\Delta}_n$ rests on a standard randomized-exposure condition, under which inverse-propensity weighting gives $\mathbb{E}[\Delta_i \mid g] = b(g)$, hence $\bar{\Delta}_n$ unbiasedly estimates the marginal benefit of Eq. (9).

Retraction and promotion. The running mean $\bar{\Delta}_n$ feeds an empirical-Bernstein confidence sequence (Howard et al., 2021), an anytime-valid interval for $b(g)$, that certifies the sign of the marginal benefit and drives the lifecycle rule:

$$\text{retract } g \text{ when } b(g) < 0 \text{ is certified}, \quad \text{promote } g \text{ when } b(g) > 0 \text{ is certified}, \tag{12}$$

where a claim on $b(g)$ is certified once the confidence sequence for $\bar{\Delta}_n$ excludes zero at or beyond the minimum exposure count for that decision, retraction leaves a tombstone in the admitted denominator so withdrawal never launders the ever-admitted rate, and promotion raises the propensity and resets the sequence. Anytime validity keeps data-dependent decisions from inflating error; each guard's tenure answers to running evidence of benefit at an unchanged admission level.

End-to-end integration. In operation the order is fixed: the prefix-fidelity checks screen first, eligible candidates enter the replay test ordered by a frozen predicted-benefit score, and a candidate the test admits enters probation in the ledger, which thereafter retracts or promotes it on its own evidence. Throughout, the backbone is frozen, the admission path is reward-free, and the only object that changes is the admitted guard library. Details in Appendix A.3 and Appendix A.4.

5 Experiments

5.1 Implementation Details

Benchmarks. On the admission diagnostic every candidate carries a ground-truth genuine-or-phantom label, fixed at construction by the mechanism that built it and never read from an admission verdict, allowing FGR to be measured directly. Three replay-feasible benchmarks carry them: AppWorld (Trivedi et al., 2024) (primary), Terminal-Bench (Merrill et al., 2026), and a determinism-filtered SWE-bench-lite (Jimenez et al., 2024) slice. Two frozen backbones give six configurations over a prespecified horizon of $T_{\max} = 200$ candidates. Details in Appendix A.5.

Baselines. Suppression-only admission (Shinn et al., 2023) (strongest same-family reference), an LLM-judge admission gate (Zheng et al., 2023) (closest alternative paradigm), static release (Zhao et al., 2024), three constructed reference admitters (no-guard, always-admit, oracle; marked † in Table 1), a naive-threshold e-value variant used by the validity ablation, and an external-memory reference reported qualitatively. Details in Appendix A.6.

Metrics. FGR (↓) is the admitted phantom share, the false discovery proportion of admission (Benjamini & Hochberg, 1995), scored on construction labels wider than the null of Eq. (2); genuine-guard retention (↑) is the recall of genuine guards still installed at the horizon; typed-matcher agreement (F1 ↑) is scored against human judgment on a disjoint 150-trace audit. Details in Appendix A.7.

Hyperparameters. All are fixed before any admission decision: the phantom bound $q_0 = 0.15$, the alternative $q_1 = 0.6$, the rounds $R = 20$, the power alternative $q_{\text{pow}} = 0.7$, the induced cut $x^* = 11$, and the error budget $\delta = 0.1$. The guarantee rests on q_0 and δ; q_1, R and q_{pow} set power. Nothing else is tuned per benchmark or backbone: the matcher window is $w = 2$ steps, and the ledger exposes at propensity 0.50 and promotes after 100 exposures. Details in Appendix A.8.

Table 1: Primary comparison on the admission diagnostic. †: constructed reference admitters; LLM-judge retention is a proxy.

	AppWorld		Terminal-Bench		SWE-bench-lite		Pooled	
Method	FGR ↓	Ret. ↑	FGR ↓	Ret. ↑	FGR ↓	Ret. ↑	FGR ↓	Ret. ↑
No-guard agent†	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Always-admit library†	0.25	1.00	0.25	1.00	0.25	1.00	0.25	1.00
Static released guardrails (Zhao et al., 2024)	0.25	1.00	0.25	1.00	0.25	1.00	0.25	1.00
Suppression-only admission (Shinn et al., 2023)	0.24	0.93	0.25	0.92	0.26	0.91	0.25	0.92
LLM-judge admission (Zheng et al., 2023)	0.14	0.71	0.16	0.68	0.19	0.64	0.16	0.68
Oracle admitter (upper bound)†	0.00	1.00	0.00	1.00	0.00	1.00	0.00	1.00
EG (ours)	**0.07**	**0.93**	**0.08**	**0.92**	**0.09**	**0.91**	**0.08**	**0.92**

Protocol. The typing function, compiler and guard language are frozen in advance, all methods share one candidate budget, replay back-end and decoding configuration, and evaluation is training-free and admission is reward-free. Candidates are ordered by a frozen predicted-benefit score; ablations run at a single seed. Details in Appendix A.6.

Software and seeds. Python 3.11 or later, harnesses pinned by commit, and each episode's initial state snapshotted so a replay restores the exact pre-locus state; evaluation is inference-only. Seeds are drawn separately for the episode stream and for replay, three each. Details in Appendix A.10.

5.2 Main Results

Comparison against every admission rule. Table 1 reports the head-to-head against every baseline layer, testing whether a fixed-level counterfactual test can hold the error budget without paying for it in genuine-guard retention. Among the non-trivial implementable rules EG is the only one that holds the budget, at FGR 0.08 under the 0.10 budget, and it retains 0.92 of genuine guards, as suppression-only admission also does. Higher retention is available only outside the budget: static release retains 1.00 by installing everything and withdrawing nothing, and pays an FGR of 0.25, more than three times EG's. What separates the rules is the evidence their admission consults. Suppression-only reads a later success as proof of prevention and credits a phantom whose failure seldom recurs anyway, leaving its FGR at the 0.25 phantom share. The LLM-judge gate scores plausibility, which a fabricated record maximizes, and still admits phantoms at 0.14 to 0.19, above the budget on every benchmark. The 0.17 pooled gap to suppression-only is what the necessity arm accounts for, supplying the evidence a later success cannot.

Decomposition across benchmarks and backbones. Table 4 breaks the pooled figures into the six confirmatory configurations, and none breaches the budget: FGR stays between 0.070 and 0.090, and the pooled 0.08 is not an artifact of aggregation. Within each benchmark the two backbone families differ by at most 0.005 in FGR; the spread follows the environment, not the model. Difficulty surfaces in the ineligible share, from 0.12 on AppWorld to 0.18 on the SWE-bench-lite slice, while FGR stays under the budget even on the hardest benchmark. Details in Appendix A.9.

5.3 Ablation Study and Analysis

Component removals. Table 2 separates the two metrics. Every validity removal lifts FGR above the 0.10 budget: the composed statistic (0.24), faithful replay (0.18) and the typed matcher (0.13) answer Challenge #1, and the necessity arm (0.25) answers Challenge #2, the statistic and the arm collapsing to essentially the phantom share. Every utility removal holds FGR at 0.08 and registers only on retained utility or the benefit estimate: dropping arm S costs 0.10 of retained utility and freezing verdicts 0.06 (Table 3). Validity and usefulness are thus bought by different components, and neither set replaces the other. Details in Appendix A.9.

Table 2: Removals from the full model. ‡: retention here is the admitted library's retained utility.

Removed	FGR ↓	Ret. ↑
composed e-value	0.24	0.99
quarantine	0.18	0.95
typed matcher	0.13	0.90
arm N	0.25	0.92
arm S‡	0.08	0.82
randomized exposure	0.08	0.92

Horizon scaling. Table 5 tracks the cumulative rate as candidates accrue, the stream-level quantity for which $\alpha_t = \delta/T_{\max} = 5 \times 10^{-4}$ prices the entire horizon in advance. Cumulative FGR moves

Table 4: Per-configuration decomposition of the EG aggregate over three benchmarks and two frozen backbone families.

Backbone	FGR ↓	Genuine retention ↑	Ineligible share
AppWorld			
family 1	0.070	0.93	0.12
family 2	0.075	0.92	0.13
Terminal-Bench			
family 1	0.080	0.92	0.14
family 2	0.080	0.92	0.16
SWE-bench-lite slice			
family 1	0.090	0.91	0.18
family 2	0.085	0.92	0.17
Pooled, both families	0.080	0.92	0.15

Table 5: Horizon trajectory at each 50-candidate epoch boundary (FGR ↓, retention ↑; budget $\delta = 0.1$; suppression-only as reference).

Benchmark	50	100	150	200
FGR ↓				
AppWorld	0.048	0.061	0.068	0.072
Terminal-Bench	0.059	0.072	0.077	0.080
SWE-bench-lite	0.073	0.081	0.086	0.088
Pooled	0.060	0.071	0.077	0.080
Suppression-only (pooled)	0.182	0.216	0.238	0.251
Genuine-guard retention ↑				
AppWorld	0.874	0.902	0.921	0.928
Terminal-Bench	0.861	0.893	0.912	0.920
SWE-bench-lite	0.842	0.874	0.896	0.912
Pooled	0.859	0.890	0.910	0.920

from 0.06 to 0.08 without touching the budget at any epoch boundary, and retention grows monotonically from 0.86 to 0.92 over the same horizon. The mild upward drift tracks phantom proposals growing denser late in the stream, with no threshold retuned along the way, while genuine admissions accrue faster than retractions, and retention climbs across the horizon while the cumulative rate drifts upward inside the budget.

Additive ladder. Table 3 builds EG up one component at a time, with each increment isolating exactly one. The asymmetry across the rungs is informative: typed compilation with no test installs the whole stream at the phantom share and answers no challenge on its own, adding the replay test closes the validity metric completely (FGR falls from above budget to its final 0.08, the rung that answers Challenges #1 and #2) but leaves retained utility stalled at 0.86, and adding the ledger moves only that column, recovering 0.06 of what that rung forfeits by retracting guards certified to hurt, at an unchanged FGR and closing Challenge #3. The two metrics are closed at different rungs, which places validity and utility on different components; no pair tested here reproduces the full model, the empirical counterpart of the framework's decomposition argument.

Table 3: Additive component ladder. ‡: the admitted library's retained utility in place of retention.

Configuration	FGR ↓	Retention ↑
Typed compilation only	0.25	1.00
+ replay test (verdict frozen)	0.08	$0.86^{\ddagger}$
+ ledger (full EG, ours)	**0.08**	$\mathbf{0.92}^{\ddagger}$

Phantom-injection stress test. Table 6 orders the admission rules into a spectrum by their evidence standard, asking the stream-level question a pooled rate can hide: whether any of the 30 independently randomized streams admits even one of its injected phantoms. Admission rates imply 209.1 of the 300 phantoms for suppression-only, 123.9 for the plausibility-filtering LLM judge, and 288.3 for the naive threshold; every one of them breaches in all 30 streams, while EG admits zero and rejects the adverse pooled-marginal null at exact level $0.9^{30} \approx 0.042$. All four rules face the identical candidate stream and differ in the evidence their standard consults, and the final step to zero is carried by the composed statistic rather than by replay volume.

Matcher agreement. Table 7 audits the reproduction predicate the admission test reads through, asking whether the typed matcher agrees with human judgment often enough. Every failure class clears the 0.80 F1 floor on the 150-trace disjoint audit, at macro average 0.84, while the LLM-judged locus reaches 0.68. The typed matcher reads each replay round off the failure class at a deterministically located locus, a reading the narration around a failure cannot move, unlike a judged locus; swapping the latter in lifts FGR to 0.13 in Table 2, above the budget. A predicate that can be argued with is one a phantom can argue past.

Lifecycle and retraction. Table 9 of Appendix A.9 asks what retraction does to the reported rate. The ever-admitted FGR finalizes at 0.09 against the instantaneous 0.08, with 11 of 161 guards retracted and 119 promoted, and tombstones keep retraction from laundering the rate. Both decisions are read from randomized exposure, which certifies the sign of each guard's marginal benefit: the 119 promotions are guards certified to help, the retractions guards certified to hurt, and adding the ledger leaves FGR at 0.08 in Table 3, refining the library without reopening the error budget.

Table 6: Phantom-injection stress test, with rate-implied counts. SO: suppression-only; LJ: LLM judge; NT: naive threshold.

	Phantoms admitted ↓			
Benchmark	SO	LJ	NT	EG (ours)
AppWorld	68.4	38.6	95.8	**0**
Terminal-Bench	71.2	41.3	96.6	**0**
SWE-bench-lite	69.5	44.0	95.9	**0**
All streams (of 300)	209.1	123.9	288.3	**0**
Streams breached	30/30	30/30	30/30	**0/30**

Table 7: Typed-matcher agreement against the 150-trace human-labeled audit, per failure class.

Failure class	Matcher F1 ↑	Clears 0.80 floor
Tool-argument error	0.87	yes
Precondition violation	0.85	yes
Hallucinated state	0.81	yes
Wrong-target selection	0.86	yes
Premature termination	0.83	yes
Loop	0.82	yes
Macro average	0.84	yes
LLM-judged locus	0.68	no

Boundary audits. The table in Appendix A.11 audits the conditions the admission test relies on. No leaked-prefix candidate is admitted and post-replay contamination is 0.00. The 0.15 coverage loss and the 0.05 drift are reported descriptively: exclusion costs coverage, not error control.

Failure modes. A deployment whose environments are not replayable shrinks the certified scope, with 0.15 of the stream already excluded; and the weakest matcher class sits at 0.81 against the 0.80 floor. Where replay is infeasible, EG withholds admission rather than weakening the test, trading coverage for validity, and the audits measure how often each boundary is reached.

6 Related Work

Reflection and self-correction. The first response to a failure is an explanation the agent writes. Reflexion stores it as episodic memory (Shinn et al., 2023; Zhou et al., 2024), Self-Refine feeds it into the next draft (Madaan et al., 2023; Gou et al., 2024), and SCoRe and its predecessors train the correction rather than prompting for it (Kumar et al., 2025; Welleck et al., 2023; Yao et al., 2024). Each improves the next attempt; none asks whether the explanation is true, and models endorse their own reasoning beyond what their accuracy supports and rarely locate their own errors (Stechly et al., 2025; Huang et al., 2024; Tyen et al., 2024). EG adds the adjudicator this line lacks: reflection still proposes the guardrail, but it must now survive a replay that could have refuted it.

Skill and memory libraries. A second line accepts the lesson and asks how to store it. ExpeL distills trajectories into strategies (Zhao et al., 2024), AdaPlanner refines plans from experience (Sun et al., 2023), DSPy compiles experience into optimizable modules (Khattab et al., 2023), and context engineering evolves a playbook (Zhang et al., 2026); LEAP turns mistakes into principles (Zhang et al., 2024c), AutoGuide extracts context-conditioned guidelines (Fu et al., 2024), and AutoManual compiles interaction into an instruction manual (Chen et al., 2024b), compilation that EG takes as its input. Admission rests on plausibility or aggregate score, not on a counterfactual test, and benchmarking shows its cost: self-generated skills lower performance where human-curated ones raise it (Li et al., 2026). Without replaying the source episode, none separates a routine that prevented a failure from one that accompanied it.

Prompt and plan optimization. A third line improves behavior without curating memories, searching over prompts or plans and keeping what scores better (Yang et al., 2024a; Yao et al., 2023), whether by proposing instructions (Zhou et al., 2023; Wang et al., 2024c), evolving them across generations (Fernando et al., 2024), or backpropagating textual feedback (Yuksekgonul et al., 2025; Pryzant et al., 2023). With selection running on aggregate outcomes, a better score establishes neither that any failure was averted nor which edit produced the gain. EG is complementary, deciding which already-proposed artifacts have earned a place by replay rather than by score.

7 Conclusion

This paper identifies the unvalidated-admission gap in self-compiled guardrails: records are installed as written, and a fabricated guard enters beside a genuine one. EG closes it by admitting a candidate only after counterfactual replay certifies what its typed failure record claims, with necessity and sufficiency arms composed into a fixed-level e-value test and a ledger that retracts guards certified to hurt. EG holds the FGR at 0.08 installed and 0.09 ever-admitted under $\delta = 0.1$, against 0.25 for suppression-only admission, keeps 0.92 of genuine guards, and admits zero of 300 injected fabrications. A guardrail earns its place in the library by surviving a test it could have failed.

REFERENCES

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and Charles Sutton. Program synthesis with large language models, 2021.

Alejandro Barredo Arrieta, Natalia Díaz-Rodríguez, Javier Del Ser, Adrien Bennetot, Siham Tabik, Alberto Barbado, Salvador García, Sergio Gil-Lopez, Daniel Molina, Richard Benjamins, Raja Chatila, and Francisco Herrera. Explainable artificial intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. *Information Fusion*, 2020. doi: 10.1016/j.inffus.2019.12.012.

Yoav Benjamini and Yosef Hochberg. Controlling the false discovery rate: A practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 57(1):289–300, 1995. doi: 10.1111/j.2517-6161.1995.tb02031.x.

Rishabh Bhardwaj and Soujanya Poria. Red-teaming large language models using chain of utterances for safety-alignment, 2023.

Ali Borji. A categorical archive of ChatGPT failures, 2023.

Lukas Brunke, Melissa Greeff, Adam W. Hall, Zhaocong Yuan, Siqi Zhou, Jacopo Panerati, and Angela P. Schoellig. Safe learning in robotics: From learning-based control to safe reinforcement learning. *Annual Review of Control, Robotics, and Autonomous Systems*, pp. 411–444, 2022. doi: 10.1146/annurev-control-042920-020211.

Markus J. Buehler. PRefLexOR: preference-based recursive language modeling for exploratory optimization of reasoning and agentic thinking, 2024.

Charlie Chen, Sebastian Borgeaud, Geoffrey Irving, Jean-Baptiste Lespiau, Laurent Sifre, and John Jumper. Accelerating large language model decoding with speculative sampling, 2023.

Liyi Chen, Panrong Tong, Zhongming Jin, Ying Sun, Jieping Ye, and Hui Xiong. Plan-on-graph: Self-correcting adaptive planning of large language model on knowledge graphs. In *Advances in Neural Information Processing Systems 37*, pp. 37665–37691, 2024a. doi: 10.52202/079017-1189.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code, 2021.

Minghao Chen, Yihang Li, Yanting Yang, Shiyu Yu, Binbin Lin, and Xiaofei He. AutoManual: Generating instruction manuals by LLM agents via interactive environmental learning. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024b. URL https://openreview.net/forum?id=Pwl9n4zlf5.

Xinyun Chen, Maxwell Lin, Nathanael Schärli, and Denny Zhou. Teaching large language models to self-debug. In *International Conference on Learning Representations (ICLR)*, 2024c.

Zhaorun Chen, Zhen Xiang, Chaowei Xiao, Dawn Song, and Bo Li. AgentPoison: Red-teaming LLM agents via poisoning memory or knowledge bases. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024d. URL https://openreview.net/forum?id=Y841BRW9rY.

Konrad Cinkusz, Jarosław A Chudziak, and Ewa Niewiadomska-Szynkiewicz. Cognitive agents powered by large language models for agile software project management. *Electronics*, 14(1):87, 2024.

Yihe Deng, Weitong Zhang, Zixiang Chen, and Quanquan Gu. Rephrase and respond: Let large language models ask better questions for themselves, 2023.

Zi-Yi Dou, Cheng-Fu Yang, Xueqing Wu, Kai-Wei Chang, and Nanyun Peng. Re-ReST: Reflection-reinforced self-training for language agents. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 15394–15411, 2024. doi: 10.18653/v1/2024.emnlp-main.861.

Patrick Fernandes, Aman Madaan, Emmy Liu, António Farinhas, Pedro Henrique Martins, Amanda Bertsch, José G. C. de Souza, Shuyan Zhou, Tongshuang Wu, Graham Neubig, and André F. T. Martins. Bridging the gap: A survey on integrating (human) feedback for natural language generation. *Transactions of the Association for Computational Linguistics*, pp. 1643–1668, 2023. doi: 10.1162/tacl_a_00626.

Chrisantha Fernando, Dylan Banarse, Henryk Michalewski, Simon Osindero, and Tim Rocktäschel. Promptbreeder: Self-referential self-improvement via prompt evolution. In *International Conference on Machine Learning (ICML)*, pp. 13481–13544, 2024.

Yao Fu, Hao Peng, Tushar Khot, and Mirella Lapata. Improving language model negotiation with self-play and in-context learning from AI feedback, 2023.

Yao Fu, Dong-Ki Kim, Jaekyeom Kim, Sungryull Sohn, Lajanugen Logeswaran, Kyunghoon Bae, and Honglak Lee. AutoGuide: Automated generation and selection of context-aware guidelines for large language model agents. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=mRIQz8Zd6O.

Omid Gheibi, Danny Weyns, and Federico Quin. Applying machine learning in self-adaptive systems: A systematic literature review. *ACM Trans. Auton. Adapt. Syst.*, 15(3), 2021. ISSN 1556-4665. doi: 10.1145/3469440. URL https://doi.org/10.1145/3469440.

Zhibin Gou, Zhihong Shao, Yeyun Gong, Yelong Shen, Yujiu Yang, Nan Duan, and Weizhu Chen. CRITIC: Large language models can self-correct with tool-interactive critiquing. In *International Conference on Learning Representations (ICLR)*, 2024.

Lin Guan, Karthik Valmeekam, Sarath Sreedharan, and Subbarao Kambhampati. Leveraging pretrained large language models to construct and utilize world models for model-based task planning. In *Advances in Neural Information Processing Systems 36*, pp. 79081–79094, 2023. doi: 10.52202/075280-3459.

Zeyu Han, Chao Gao, Jinyang Liu, Jeff Zhang, and Sai Qian Zhang. Parameter-efficient fine-tuning for large models: A comprehensive survey. *Trans. Mach. Learn. Res.*, 2024.

Hongliang He, Wenlin Yao, Kaixin Ma, Wenhao Yu, Yong Dai, Hongming Zhang, Zhenzhong Lan, and Dong Yu. WebVoyager: Building an end-to-end web agent with large multimodal models. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2024.

D. G. Horvitz and D. J. Thompson. A generalization of sampling without replacement from a finite universe. *Journal of the American Statistical Association*, 1952. doi: 10.1080/01621459.1952.10483446.

Steven R. Howard, Aaditya Ramdas, Jon McAuliffe, and Jasjeet Sekhon. Time-uniform, nonparametric, nonasymptotic confidence sequences. *The Annals of Statistics*, 2021. doi: 10.1214/20-aos1991.

Mengkang Hu, Yao Mu, Xinmiao Yu, Mingyu Ding, Shiguang Wu, Wenqi Shao, Qiguang Chen, Bin Wang, Yu Qiao, and Ping Luo. Tree-planner: Efficient close-loop task planning with large language models. In *International Conference on Learning Representations*, 2024.

Jie Huang, Xinyun Chen, Swaroop Mishra, Huaixiu Steven Zheng, Adams Wei Yu, Xinying Song, and Denny Zhou. Large language models cannot self-correct reasoning yet. In *International Conference on Learning Representations (ICLR)*, 2024.

Xu Huang, Jianxun Lian, Yuxuan Lei, Jing Yao, Defu Lian, and Xing Xie. Recommender AI agent: Integrating large language models for interactive recommendations. *ACM Trans. Inf. Syst.*, 2025. doi: 10.1145/3731446.

Yu Huang. Levels of AI agents: from rules to large language models, 2024.

Eyke Hüllermeier and Willem Waegeman. Aleatoric and epistemic uncertainty in machine learning: an introduction to concepts and methods. *Machine Learning*, pp. 457–506, 2021. doi: 10.1007/s10994-021-05946-3.

Carlos E. Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik Narasimhan. SWE-bench: Can language models resolve real-world GitHub issues? In *International Conference on Learning Representations (ICLR)*, 2024.

Jean Kaddour, Joshua Harris, Maximilian Mozes, Herbie Bradley, Roberta Răileanu, and Robert McHardy. Challenges and applications of large language models, 2023.

Omar Khattab, Arnav Singhvi, Paridhi Maheshwari, Zhiyuan Zhang, Keshav Santhanam, Sri Vardhamanan, Saiful Haq, Ashutosh Sharma, Thomas T. Joshi, Hanna Moazam, Heather Miller, Matei Zaharia, and Christopher Potts. DSPy: Compiling declarative language model calls into self-improving pipelines, 2023. URL https://arxiv.org/abs/2310.03714.

Humam Kourani, Alessandro Berti, Daniel Schuster, and Wil M. P. van der Aalst. Evaluating large language models on business process modeling: framework, benchmark, and self-improvement analysis. *Software and Systems Modeling*, 2025. doi: 10.1007/s10270-025-01318-w.

Aviral Kumar, Vincent Zhuang, Rishabh Agarwal, Yi Su, John D. Co-Reyes, Avi Singh, Kate Baumli, Shariq Iqbal, Colton Bishop, Rebecca Roelofs, Lei M. Zhang, Kay McKinney, Disha Shrivastava, Cosmin Paduraru, George Tucker, Doina Precup, Feryal M. P. Behbahani, and Aleksandra Faust. Training language models to self-correct via reinforcement learning. In *International Conference on Learning Representations*, 2025.

Loka Li, Zhenhao Chen, Guangyi Chen, Yixuan Zhang, Yusheng Su, Eric P. Xing, and Kun Zhang. Confidence matters: Revisiting intrinsic self-correction capabilities of large language models, 2024.

Xiangyi Li, Yimin Liu, Wenbo Chen, et al. SkillsBench: Benchmarking how well agent skills work across diverse tasks, 2026.

Yuanfei Lin, Chenran Li, Mingyu Ding, Masayoshi Tomizuka, Wei Zhan, and Matthias Althoff. DrPlanner: Diagnosis and repair of motion planners for automated vehicles using large language models. *IEEE Robotics Autom. Lett.*, 2024.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173, 2024. doi: 10.1162/tacl_a_00638. URL https://aclanthology.org/2024.tacl-1.9/.

Zeyi Liu, Arpit Bahety, and Shuran Song. REFLECT: Summarizing robot experiences for failure explanation and correction. In *CoRL*, 2023.

Jieyi Long. Large language model guided tree-of-thought, 2023.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. Self-refine: Iterative refinement with self-feedback. In *Advances in Neural Information Processing Systems 36*, pp. 46534–46594, 2023. doi: 10.52202/075280-2019.

Mike A. Merrill, Alexander G. Shaw, Nicholas Carlini, Boxuan Li, Harsh Raj, Ivan Bercovich, Lin Shi, Jeong Yeon Shin, Thomas Walshe, E. Kelly Buchanan, Junhong Shen, Guanghao Ye, Haowei Lin, Jason Poulos, Maoyu Wang, Marianna Nezhurina, Jenia Jitsev, Di Lu, Orfeas Menis Mastromichalakis, Zhiwei Xu, Zizhao Chen, Yue Liu, Robert Zhang, Leon Liangyu Chen, Anurag Kashyap, Jan-Lucas Uslu, Jeffrey Li, Jianbo Wu, Minghao Yan, Song Bian, Vedang Sharma, Ke Sun, Steven Dillmann, Akshay Anand, Andrew Lanpouthakoun, Bardia Koopah, Changran Hu, Etash Guha, Gabriel H. S. Dreiman, Jiacheng Zhu, Karl Krauth, Li Zhong, Niklas Muennighoff, Robert Amanfu, Shangyin Tan, Shreyas Pimpalgaonkar, Tushar Aggarwal, Xiangning Lin, Xin Lan, Xuandong Zhao, Yiqing Liang, Yuanli Wang, Zilong Wang, Changzhi Zhou, David Heineman, Hange Liu, Harsh Trivedi, John Yang, Junhong Lin, Manish Shetty, Michael Yang, Nabil Omi, Negin Raoof, Shanda Li, Terry Yue Zhuo, Wuwei Lin, Yiwei Dai, Yuxin Wang, Wenhao Chai, Shang Zhou, Dariush Wahdany, Ziyu She, Jiaming Hu, Zhikang Dong, Yuxuan Zhu, Sasha Cui, Ahson Saiyed, Arinbjörn Kolbeinsson, Jesse Hu, Christopher Michael Rytting,

Ryan Marten, Yixin Wang, Alex Dimakis, Andy Konwinski, and Ludwig Schmidt. Terminal-bench: Benchmarking agents on hard, realistic tasks in command line interfaces, 2026. URL https://arxiv.org/abs/2601.11868.

Charles Packer, Sarah Wooders, Kevin Lin, Vivian Fang, Shishir G. Patil, Ion Stoica, and Joseph E. Gonzalez. MemGPT: Towards LLMs as operating systems, 2023.

Andrei Paleyes, Raoul-Gabriel Urma, and Neil D. Lawrence. Challenges in deploying machine learning: A survey of case studies. *ACM Computing Surveys*, pp. 1–29, 2022. doi: 10.1145/3533378.

Liangming Pan, Michael Saxon, Wenda Xu, Deepak Nathani, Xinyi Wang, and William Yang Wang. Automatically correcting large language models: Surveying the landscape of diverse automated correction strategies. *Transactions of the Association for Computational Linguistics*, 12:484–506, 2024. doi: 10.1162/tacl_a_00660.

Reid Pryzant, Dan Iter, Jerry Li, Yin Lee, Chenguang Zhu, and Michael Zeng. Automatic prompt optimization with "gradient descent" and beam search. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 7957–7968, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.494. URL https://aclanthology.org/2023.emnlp-main.494/.

Yuxiao Qu, Tianjun Zhang, Naman Garg, and Aviral Kumar. Recursive introspection: Teaching language model agents how to self-improve. In *Advances in Neural Information Processing Systems 37*, pp. 55249–55285, 2024. doi: 10.52202/079017-1754.

Aaditya Ramdas, Peter Grünwald, Vladimir Vovk, and Glenn Shafer. Game-theoretic statistics and safe anytime-valid inference. *Statistical Science*, 2023. doi: 10.1214/23-sts894.

Alex Sheng. From language models to practical self-improving computer agents, 2024.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. In *Advances in Neural Information Processing Systems 36*, pp. 8634–8652, 2023. doi: 10.52202/075280-0377.

Marina Sokolova and Guy Lapalme. A systematic analysis of performance measures for classification tasks. *Information Processing & Management*, 45(4):427–437, 2009. doi: 10.1016/j.ipm.2009.03.002.

Kaya Stechly, Karthik Valmeekam, and Subbarao Kambhampati. On the self-verification limitations of large language models on reasoning and planning tasks. In *International Conference on Learning Representations*, 2025.

Haotian Sun, Yuchen Zhuang, Lingkai Kong, Bo Dai, and Chao Zhang. AdaPlanner: Adaptive planning from feedback with language models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Simeng Sun, Yang Liu, Shuohang Wang, Dan Iter, Chenguang Zhu, and Mohit Iyyer. PEARL: Prompting large language models to plan and execute actions over long documents. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 469–486, St. Julian's, Malta, 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.eacl-long.29. URL https://aclanthology.org/2024.eacl-long.29/.

Mirac Süzgün and Adam Tauman Kalai. Meta-prompting: Enhancing language models with task-agnostic scaffolding, 2024.

Zhengwei Tao, Ting-En Lin, Xiancai Chen, Hangyu Li, Yuchuan Wu, Yongbin Li, Zhi Jin, Fei Huang, Dacheng Tao, and Jingren Zhou. A survey on self-evolution of large language models, 2024.

Harsh Trivedi, Tushar Khot, Mareike Hartmann, Ruskin Manku, Vinty Dong, Edward Li, Shashank Gupta, Ashish Sabharwal, and Niranjan Balasubramanian. AppWorld: A controllable world of apps and people for benchmarking interactive coding agents. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2024.

Miles Turpin, Julian Michael, Ethan Perez, and Samuel R. Bowman. Language models don't always say what they think: Unfaithful explanations in chain-of-thought prompting. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=bzs4uPLXvi.

Gladys Tyen, Hassan Mansoor, Victor Carbune, Peter Chen, and Tony Mak. LLMs cannot find reasoning errors, but can correct them given the error location. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 13894–13908, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-acl.826. URL https://aclanthology.org/2024.findings-acl.826/.

Vladimir Vovk and Ruodu Wang. E-values: Calibration, combination and applications. *The Annals of Statistics*, 2021. doi: 10.1214/20-aos2020.

A. Wald. Sequential tests of statistical hypotheses. *The Annals of Mathematical Statistics*, 1945. doi: 10.1214/aoms/1177731118.

Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An open-ended embodied agent with large language models. *Transactions on Machine Learning Research*, 2024a.

Wenxiao Wang, Wei Chen, Yicong Luo, Yongliu Long, Zhengkai Lin, Liye Zhang, Binbin Lin, Deng Cai, and Xiaofei He. Model compression and efficient inference for large language models: A survey, 2024b.

Xinyuan Wang, Chenxi Li, Zhen Wang, Fan Bai, Haotian Luo, Jiayou Zhang, Nebojsa Jojic, Eric Xing, and Zhiting Hu. PromptAgent: Strategic planning with language models enables expert-level prompt optimization. In *The Twelfth International Conference on Learning Representations*, 2024c. URL https://openreview.net/forum?id=22pyNMuIoa.

Zihao Wang, Shaofei Cai, Guanzhou Chen, Anji Liu, Xiaojian Ma, and Yitao Liang. Describe, explain, plan and select: Interactive planning with LLMs enables open-world multi-task agents. In *Advances in Neural Information Processing Systems 36*, pp. 34153–34189, 2023. doi: 10.52202/075280-1480.

Zora Zhiruo Wang, Jiayuan Mao, Daniel Fried, and Graham Neubig. Agent workflow memory. In *Proceedings of the 42nd International Conference on Machine Learning*, ICML'25. JMLR.org, 2025.

Sean Welleck, Ximing Lu, Peter West, Faeze Brahman, Tianxiao Shen, Daniel Khashabi, and Yejin Choi. Generating sequences by learning to self-correct. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=hH36JeQZDaO.

Tongshuang Wu, Michael Terry, and Carrie J. Cai. AI chains: Transparent and controllable human-AI interaction by chaining large language model prompts. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*, 2022.

Fengli Xu, Qianyue Hao, Chenyang Shao, Zefang Zong, Yu Li, Jingwei Wang, Yunke Zhang, Jingyi Wang, Xiaochong Lan, Jiahui Gong, Tianjian Ouyang, Fanjin Meng, Yuwei Yan, Qinglong Yang, Yiwen Song, Sijian Ren, Xinyuan Hu, Jie Feng, Chen Gao, and Yong Li. Toward large reasoning models: A survey of reinforced reasoning with large language models. *Patterns*, 6(10):101370, 2025. ISSN 2666-3899. doi: 10.1016/j.patter.2025.101370. URL https://doi.org/10.1016/j.patter.2025.101370.

Zhenran Xu, Senbao Shi, Baotian Hu, Jindi Yu, Dongfang Li, Min Zhang, and Yuxiang Wu. Towards reasoning in large language models via multi-agent peer review collaboration, 2023.

Chengrun Yang, Xuezhi Wang, Yifeng Lu, Hanxiao Liu, Quoc V. Le, Denny Zhou, and Xinyun Chen. Large language models as optimizers. In *International Conference on Learning Representations (ICLR)*, 2024a.

John Yang, Carlos E Jimenez, Alexander Wettig, Kilian Lieret, Shunyu Yao, Karthik R Narasimhan, and Ofir Press. SWE-agent: Agent-computer interfaces enable automated software engineering. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024b. URL https://openreview.net/forum?id=mXpq6ut8J3.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. In *Advances in Neural Information Processing Systems 36*, 2023. doi: 10.52202/075280-0517.

Weiran Yao, Shelby Heinecke, Juan Carlos Niebles, Zhiwei Liu, Yihao Feng, Le Xue, Rithesh Murthy, Zeyuan Chen, Jianguo Zhang, Devansh Arpit, Ran Xu, Phil Mui, Huan Wang, Caiming Xiong, and Silvio Savarese. Retroformer: Retrospective large language agents with policy gradient optimization. In *ICLR*, 2024.

Mert Yuksekgonul, Federico Bianchi, Joseph Boen, Sheng Liu, Pan Lu, Zhi Huang, Carlos Guestrin, and James Zou. Optimizing generative AI by backpropagating language model feedback. *Nature*, 639:609–616, 2025.

Ceyao Zhang, Kaijie Yang, Siyi Hu, Zihao Wang, Guanghe Li, Yihang Sun, Cheng Zhang, Zhaowei Zhang, Anji Liu, Song-Chun Zhu, Xiaojun Chang, Junge Zhang, Feng Yin, Yitao Liang, and Yaodong Yang. ProAgent: Building proactive cooperative agents with large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2024a.

Che Zhang, Zhenyang Xiao, Chengcheng Han, Yixin Lian, and Yuejian Fang. Learning to check: Unleashing potentials for self-correction in large language models. arXiv preprint arXiv:2402.13035, 2024b.

Qizheng Zhang, Changran Hu, Shubhangi Upasani, Boyuan Ma, Fenglu Hong, Vamsidhar Kamanuru, Jay Rainton, Chen Wu, Mengmeng Ji, Hanchen Li, Urmish Thakker, James Zou, and Kunle Olukotun. Agentic context engineering: Evolving contexts for self-improving language models. In *International Conference on Learning Representations*, pp. 86069–86100, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/8a94ff6f922d995d7d3f4ebf4143e442-Paper-Conference.pdf.

Tianjun Zhang, Aman Madaan, Luyu Gao, Steven Zheng, Swaroop Mishra, Yiming Yang, Niket Tandon, and Uri Alon. In-context principle learning from mistakes. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 59520–59558. PMLR, 21–27 Jul 2024c. URL https://proceedings.mlr.press/v235/zhang24at.html.

Yifan Zhang, Jingqin Yang, Yang Yuan, and Andrew Chi-Chih Yao. Cumulative reasoning with large language models, 2023.

Andrew Zhao, Daniel Huang, Quentin Xu, Matthieu Gaetan Lin, Yong-Jin Liu, and Gao Huang. ExpeL: LLM agents are experiential learners. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2024.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging LLM-as-a-judge with MT-Bench and chatbot arena. In *Advances in Neural Information Processing Systems 36, Datasets and Benchmarks Track*, 2023. arXiv:2306.05685.

Andy Zhou, Kai Yan, Michal Shlapentokh-Rothman, Haohan Wang, and Yu-Xiong Wang. Language agent tree search unifies reasoning, acting, and planning in language models. In *Forty-first International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=njwv9BsGHF.

Yongchao Zhou, Andrei Ioan Muresanu, Ziwen Han, Keiran Paster, Silviu Pitis, Harris Chan, and Jimmy Ba. Large language models are human-level prompt engineers. In *International Conference on Learning Representations (ICLR)*, 2023.

AI USE STATEMENT

We used generative AI tools to translate the text (Claude Opus 5.0), to check grammar and spelling and polish the wording (GPT-5.6), to guide the drawing of the figures (Gemini), and to check the content (Grok). We did not use generative AI tools to modify code. We have reviewed all AI-assisted work: we checked every reference for hallucinated citations. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

The details needed to reproduce the results of this paper, including the typed failure records, the compiler and guard language, the replay procedure and the admission test, the hyperparameters, seeds and backbone configurations, the diagnostic construction and the evaluation metrics, are disclosed in this appendix.

A APPENDIX

Table 8 lists the contents of each part of the appendix.

Table 8: What each part of the appendix holds.

Section	Contents
Appendix A.0	Limitations
Appendix A.1	Extended related work
Appendix A.2	Motivation diagnostic in detail
Appendix A.3	Method implementation details and the full algorithm
Appendix A.4	Guarantees for the replay test
Appendix A.5	Diagnostic construction
Appendix A.6	Baselines and configurations
Appendix A.7	Evaluation metrics
Appendix A.8	Hyperparameters and sensitivity
Appendix A.9	Extended experimental details
Appendix A.10	Reproducibility and pinned configuration
Appendix A.11	Analysis and boundary conditions
Appendix A.12	Discussion
Appendix A.13	Future work

A.0 LIMITATIONS

Operating characteristics across round counts. The number of replay rounds decides how readily a genuine guard is admitted and how much replay each candidate consumes, and a reader who deploys the test in a new environment needs to know how the admission cut and the power move as that number changes. The paper lacks this analysis: the admission test is characterized at a single round count, and we have not evaluated it at other round counts.

Sensitivity to the error budget. The error budget is the one level a user sets, and it trades the bound on fabricated guardrails against how readily genuine guards enter the library; a reader therefore needs to see how the admission threshold and the realized verdicts respond when the budget is tightened or relaxed. The paper lacks this analysis: every result is obtained at a single budget, and we have not evaluated other budgets over the same horizon.

Compute cost. Replay adds work at admission time, and whether that work is affordable decides whether the method can run on long agent streams; a reader needs an itemized account of the replays and tokens that admission consumes. The paper lacks this accounting: we have not measured the compute cost of the evaluation.

A.1 Extended Related Work

The main paper positions EG against the closest three lines of work in a single compact section. This appendix covers the wider related literature, organized by the question each line answers about accumulated failure knowledge.

Reflection and verbal self-correction. One line has the agent critique its own trajectory in natural language and condition later attempts on the critique. The critique is produced by the same model whose behavior it describes, and the external signals some members consult, tool outputs or executed tests, bear on the revised attempt rather than on whether the described failure occurred: none re-executes the source episode with and without the correction to see whether the named failure recurs and is then blocked. This makes the paradigm complementary to EG, and it is why the external-memory reference is reported qualitatively in Appendix A.6: reflections do not form discrete admitted objects to which an FGR could attach. The systems the main paper names head this line: Reflexion stores the critique as episodic memory (Shinn et al., 2023), Self-Refine feeds it into the next draft (Madaan et al., 2023), Self-Debugging grounds it in execution feedback (Chen et al., 2024c), and SCoRe trains the correction rather than prompting for it (Kumar et al., 2025), while the self-critique studies that motivate our test report endorsement rates accuracy does not support (Stechly et al., 2025; Li et al., 2024; Huang et al., 2024). The family extends further to agents that check their own work (Zhang et al., 2024b; Gou et al., 2024), rewrite the question before answering it (Deng et al., 2023), introspect recursively over past attempts (Qu et al., 2024; Buehler, 2024), and reinforce the correction behavior through self-training or self-play (Dou et al., 2024; Fu et al., 2023; Xu et al., 2025); surveys of self-improving systems and of documented model failures map both the promise and the failure modes (Sheng, 2024; Borji, 2023; Kaddour et al., 2023), and further analyses question how far verbal self-correction can be trusted at all (Pan et al., 2024).

Self-improving agents and skill libraries. The line that EG belongs to treats an agent's own execution history as a source of reusable competence, compiling successful behavior into callable skills and failures into avoidance knowledge. The recurring structural weakness is that acquisition is validated by whether the artifact is produced, not by whether it is correct: a skill enters the library because the compiler emitted it. Where validation exists it is a success check or an aggregate task metric, which cannot distinguish a skill that caused the success from one that merely accompanied it. EG's necessity arm is aimed exactly at that gap, and the diagnostic's outcome-inversion phantoms are the adversarial case it addresses. The members are ExpeL (Zhao et al., 2024), AdaPlanner (Sun et al., 2023), agentic context engineering (Zhang et al., 2026), DSPy (Khattab et al., 2023), a skill library grown from self-verified programs (Wang et al., 2024a), workflows induced from past trajectories (Wang et al., 2025), and a memory manager that pages experience in and out of context (Packer et al., 2023); the self-evolution survey covers the family (Tao et al., 2024) and recent benchmarking quantifies the cost once the library grows (Li et al., 2026). The line is wide beyond them: it spans libraries organized around cognitive or self-adaptive architectures (Cinkusz et al., 2024; Gheibi et al., 2021; Huang, 2024), deployments in recommendation, business-process, and planning settings where a stale skill carries downstream cost (Huang et al., 2025; Kourani et al., 2025; Lin et al., 2024), and efforts to cut the inference cost of the base model the library sits on (Han et al., 2024; Wang et al., 2024b; Chen et al., 2023). Code benchmarks make the correctness gap concrete, since a stored routine that passes one suite may silently fail another (Chen et al., 2021; Austin et al., 2021).

LLM-as-judge evaluation. The natural admission gate is to ask a capable model whether a candidate is sound. The known failure modes of that paradigm, position and verbosity biases and unreliability on comparisons where surface fluency and substantive quality diverge, are precisely the regime a fabricated guardrail occupies: it is fluent by construction and wrong only in a way that the text does not expose. The stress condition of Appendix A.5 makes this concrete by retaining only fabrications that survive a text-only plausibility screen. Where a judge score is the only evi-

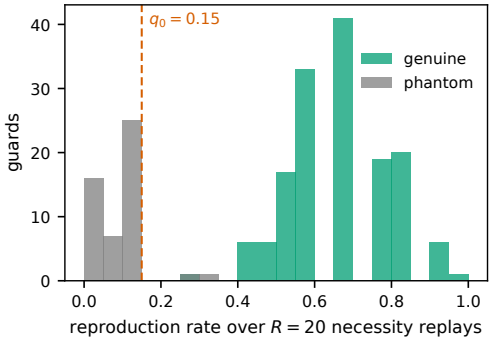

Figure 4: Reproduction of the named failure under necessity replay: genuine guards concentrate well above the phantom bound q_0, while phantom guards' named failures rarely recur.

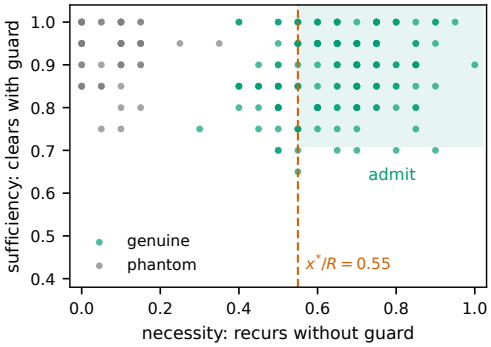

Figure 5: Necessity against sufficiency for the same guards: phantoms clear the sufficiency arm as easily as genuine guards, and only the necessity threshold separates the two families.

dence available, work on uncertainty, explainability, safety, and adversarial probing is what supplies the missing per-decision guarantee (Hüllermeier & Waegeman, 2021; Barredo Arrieta et al., 2020; Brunke et al., 2022; Bhardwaj & Poria, 2023), and monitoring practice for deployed learning systems makes the same point over time (Paleyes et al., 2022).

Sequential testing and anytime-valid inference. The statistical machinery EG uses, likelihood-ratio sequential tests (Wald, 1945), e-values and supermartingale constructions (Vovk & Wang, 2021; Ramdas et al., 2023), and confidence sequences valid at data-dependent stopping times (Howard et al., 2021), was developed for settings where a decision-maker must act repeatedly on accumulating evidence without a fixed sample size. Agent admission is such a setting: candidates arrive in a stream, each demands an immediate verdict, and the quantity to control is a cumulative rate over the horizon, not a per-decision error. The contribution is the identification of a replay-grounded statistic to which the existing machinery applies, and Appendix A.4 prices the horizon in advance with a union bound.

Prompt and plan optimization. The third line the main paper names searches over prompts or plans and keeps what scores better (Yang et al., 2024a; Hu et al., 2024), by having the model propose instructions (Zhou et al., 2023), evolving them across generations (Fernando et al., 2024), or backpropagating textual feedback through a compound system (Yuksekgonul et al., 2025). It is correspondingly broad, spanning cumulative-reasoning and meta-prompting procedures (Zhang et al., 2023; Süzgün & Kalai, 2024), closed-loop planners that prune candidates before execution (He et al., 2024; Zhang et al., 2024a), search over intermediate reasoning steps (Long, 2023; Xu et al., 2023; Sun et al., 2024; Wang et al., 2023), planners that revise against a graph or symbolic model (Chen et al., 2024a; Guan et al., 2023), and human-in-the-loop chains that expose each step for inspection (Wu et al., 2022; Fernandes et al., 2023).

Causal attribution in sequential decision-making. The ledger's marginal-benefit estimate connects to work on off-policy and randomized evaluation of interventions in sequential settings. Exposure is randomized, not observational: a guard's benefit is estimated from a coin flip at known propensity, which is why the estimate is not confounded by when the guard happens to fire. Assumption 2 states the condition this requires, and the propensity change at promotion is handled by restarting the running mean, not by pooling across weighting regimes.

A.2 Motivation Diagnostic in Detail

The motivation figure in the main paper carries a compact caption that states one takeaway per panel. Here we present each panel as a standalone figure and give the extended reading: 200 self-authored candidate guards with genuine-or-phantom labels, replayed over $R = 20$ necessity rounds.

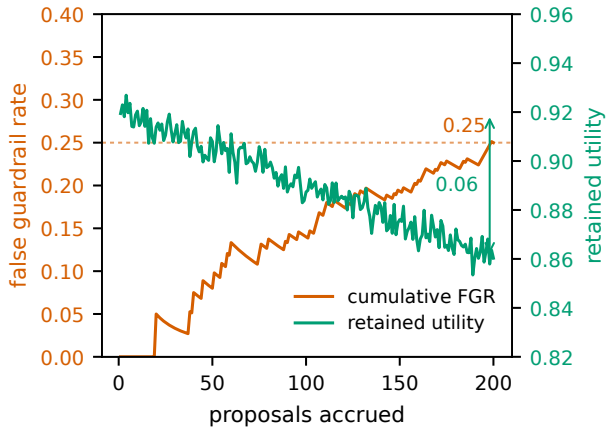

Figure 6: Suppression-only admission over the horizon: the cumulative FGR converges to 0.25 while the retained utility of the admitted library decays by 0.06.

Unvalidated failure knowledge. Figure 4 addresses Challenge #1, unvalidated failure knowledge. Each of the 200 candidates is re-executed over $R = 20$ necessity replays and its per-guard reproduction rate is recorded. The genuine population concentrates around reproduction rates of 0.6–0.8, far above the phantom bound, whereas the phantom population masses below $q_0 = 0.15$, the nominal per-round spurious-reproduction cap of assumption A1, with a thin tail of phantoms crossing it. The separation is the empirical basis for testing a failure record rather than trusting it: a record whose named failure never recurs under faithful replay cannot be distinguished from a fabrication by any amount of plausible text.

Suppression is not prevention. Figure 5 addresses Challenge #2, suppression is not prevention. Each guard is placed by its necessity-arm reproduction rate (horizontal) and its sufficiency-arm clearance rate (vertical). Both families sit high on the sufficiency axis, and sufficiency evidence alone cannot separate them; only the necessity threshold $x^*/R = 0.55$ does, and admission in EG requires both arms. The shaded region's left edge is the necessity threshold, while its lower edge is a drawing convention that bounds the plotted points, since the sufficiency arm enters as a binary indicator and carries no rate threshold.

Accumulation without accounting. Figure 6 addresses Challenge #3, accumulation without accounting. Over a prespecified 200-proposal horizon under suppression-only admission, the cumulative FGR climbs and converges to the phantom stream share of 0.25, while the retained utility of the admitted library decays by 0.06 because non-beneficial guards are never withdrawn. The two curves together show that per-proposal plausibility controls neither the accumulated phantom share nor the continued usefulness of what has been installed.

Panel axes and trends. For precise reading, we state each panel's axes and the trend it carries. In Figure 4 the horizontal axis is the reproduction rate over $R = 20$ replays and the vertical axis is the guard count, with one histogram per family; the trend is a bimodal separation at the phantom bound. In Figure 5 the horizontal axis is the necessity rate and the vertical axis is the sufficiency rate, one point per guard; the trend is that the two families overlap vertically but separate horizontally. In Figure 6 the horizontal axis is the number of proposals accrued, the left vertical axis the cumulative FGR, and the right vertical axis the retained utility; the trend is convergence to 0.25 on the left axis against a decay of 0.06 on the right.

A.3 Method Implementation Details

This section gives the implementation details deferred from the method.

Injection semantics. An advisory guard annotates the triggering step, while an enforcing guard that fires blocks the action and returns a designated blocking sentinel. That sentinel is excluded from the failure taxonomy, so a guard's own intervention is never mistaken for a new failure and cannot inflate the reproduction count of any later candidate. Compiling into a fixed, inspectable language instead of opaque parameters is what makes a guard auditable and toggleable by replay, without retraining.

Prefix-fidelity checks. Three conditions quarantine a candidate before any arm is formed: a pre-locus hash divergence between the cached and re-executed prefixes, a source episode that is not replayable, and a failed double-prefix probe. A quarantined candidate is dropped without resampling, since resampling until a prefix happens to reconstruct would select on the replay outcome and bias the arms it feeds.

Matcher audit. The typed matcher decides whether a named failure recurred in a replay. It is audited against human judgment on 150 traces held disjoint from every trace used to develop the typing function or the compiler. The traces are sampled stratified by failure class, 25 per class over the six classes, so that a rare class cannot be masked by a common one in the aggregate; this stratification is why per-class F1, in place of a pooled score, is the reported quantity. Labels are assigned from the episode transcript and the recorded locus without sight of the matcher's verdict, which is why they cannot be anchored to the system under audit. The acceptance floor is 0.80 per class, the macro average is 0.84, and every class clears the floor.

Algorithm 1 gives the complete EG procedure over a horizon of at most $T_{\max}$ proposals, in the order in which a candidate meets the three components. Each trace is first typed and compiled, which is what gives the null $H_0(g)$ of Eq. (2) a referent that re-execution can test. The candidate is then either excluded by the prefix-fidelity checks or as degenerate-deterministic, or replayed under both arms and composed into the statistic $\widetilde{E}(g)$ of Eq. (7); only a candidate clearing $T_{\max}/\delta$, a threshold read off the budget and the horizon before the stream begins, is installed. Admission does not end the test: every triggering episode of an installed guard draws a randomized exposure, updates the confidence sequence of the ledger, and can retract the guard, which then stays in the ever-admitted set as a tombstone so that a withdrawal never lowers the ever-admitted rate. The loop therefore returns two objects, the library L in force and the ever-admitted set A against which $\mathrm{FGR}(A) \le \delta$ is read.

The method section and the remaining appendices fix the horizon and round settings, the typing function and compiler, and the definitions of FGR, genuine-guard retention, and per-class matcher agreement, together with the confidence-sequence interval semantics used throughout.

A.4 Guarantees for the Replay Test

This appendix states Assumption 1 in full, records the scope conditions under which it is credible, proves Theorem 1 and Proposition 2, and states Assumption 2 used by the ledger.

Setting and notation. Fix a candidate guard g compiled from a failure trace ξ. The necessity arm is replayed R times from the versioned initial state; write $x_r \in \{0,1\}$ for the indicator that the named failure of g recurs in round r, and $\mathcal{F}_{r-1}$ for the sigma-field generated by everything observed before round r, including the outcomes $x_1, \dots, x_{r-1}$. Write $S = \mathrm{Suf}(g)$ of Eq. (6) for the sufficiency indicator. The null $H_0(g)$ is Eq. (2), that the failure g names reproduces under replay with probability at most q_0. The hyperparameters $q_0, q_1 \in (0,1)$ with $q_0 < q_1$, the round count R, the budget δ and the horizon $T_{\max}$ are all fixed before any candidate is seen.

Assumption 1 (Bounded spurious reproduction)**.** *Let $\mathcal{F}_{r-1}$ denote the history before replay round r. For a phantom guard g, that is, one satisfying $H_0(g)$ of Eq.* (2)*, the round-r reproduction indicator satisfies $\Pr[x_r = 1 \mid \mathcal{F}_{r-1}] \le q_0$ for every $r \in \{1, \dots, R\}$.*

The assumption in full. Assumption 1 is the statement that under $H_0(g)$,

$$\Pr\left[x_r = 1 \,\middle|\, \mathcal{F}_{r-1}\right] \;\le\; q_0 \qquad \text{for every } r \in \{1, \dots, R\}. \tag{13}$$

Three points fix its scope. First, the bound is conditional and not marginal, and therefore it constrains the reproduction probability even after adverse rounds; this is what permits the rounds to be

Algorithm 1 EG: typed compilation, replay admission, and the post-admission ledger over a horizon

Require: stream of failure traces over a horizon of at most $T_{\max}$ proposals; budget δ; frozen typing function T, compiler G, backbone, and predicted-benefit score; hyperparameters $q_0 < q_1$ and round count R

Ensure: installed library L and ever-admitted set A with its tombstones; under Assumption 1, Theorem 1 gives a family-wise error rate (FWER) of at most δ, hence $\mathrm{FGR}(A) \le \delta$ at every cumulative index of the horizon

$L \leftarrow \varnothing$, $A \leftarrow \varnothing$
for each failure trace ξ, at most $T_{\max}$ in all; eligible candidates enter replay in frozen predicted-benefit order **do**
  Typed failure compilation
  $T(\xi) \leftarrow$ (failure class, locus, context features) ▷ Eq. (3); fixes ϕ_{f_g} of Eq. (4)
  $g \leftarrow G\big(T(\xi),\ \text{trace window}\big)$ ▷ Eq. (5)
  Counterfactual replay admission
  if ξ fails the prefix-fidelity checks or arises from a degenerate-deterministic episode **then**
    exclude g from testing without resampling
  else
    for $r = 1$ to R **do**
      $x_r \leftarrow \mathbf{1}\big[\phi_{f_g}(\mathcal{R}_N(\xi, -g)) = 1\big]$ ▷ necessity arm $\mathrm{Nec}(g)$, Eq. (6)
    end for
    $\mathrm{Suf}(g) \leftarrow \mathbf{1}\big[\phi_{f_g}(\mathcal{R}_S(\xi, +g)) = 0\big]$ ▷ sufficiency arm, Eq. (6)
    $\widetilde{E}(g) \leftarrow \Big[\prod_{r=1}^{R} \big(\frac{q_1}{q_0}\big)^{x_r} \big(\frac{1-q_1}{1-q_0}\big)^{1-x_r}\Big] \mathrm{Suf}(g)$ ▷ Eq. (7)
    if $\widetilde{E}(g) \ge T_{\max}/\delta$ **then** ▷ threshold fixed before the stream, Eq. (8)
      $L \leftarrow L \cup \{g\}$, $A \leftarrow A \cup \{g\}$; open a probation entry
    end if
  end if
  Post-admission ledger
  for all live episodes e until the next trace, and each $g' \in L$ whose trigger fires on e **do**
    draw exposure Z_i at known propensity p_i immediately before the triggering action, run e with g' active iff $Z_i = 1$, observe u_i, and form Δ_i ▷ Eq. (11)
    update $\bar{\Delta}_n$ and its empirical-Bernstein confidence sequence ▷ Eq. (10)
    once the sequence excludes zero at or beyond the minimum exposure count for that decision, retract ($L \leftarrow L \setminus \{g'\}$, its tombstone stays in A) or promote (raise its propensity, reset the sequence) ▷ Eq. (12)
  end for
end for
return L and A

composed multiplicatively without an independence assumption. Second, it is one-sided: nothing is assumed about how often a genuine failure recurs, and the power statement below is therefore a separate claim that carries its own hypothesis $p \ge q_{\mathrm{pow}}$. Third, it is an assumption about the replay harness as much as about the guard. Faithful reconstruction of the prefix is necessary for it to be credible, which is why any pre-locus hash divergence, non-replayable source, or failed double-prefix probe quarantines the candidate outright. The screens establish that the replay reproduces the recorded prefix, which removes the most common way the bound could fail, but they do not by themselves certify that a phantom's named failure recurs at rate at most q_0. That remains a substantive assumption about the environment, set as described in Appendix A.8 and probed by the sensitivity analysis there. Environments without an exact versioned reset are outside its scope, and the coverage cost of that exclusion is reported in Table 11.

Theorem 1 (Horizon-wide phantom-rate control)**.** *Under Assumption 1, $\widetilde{E}(g)$ of Eq.* (7) *satisfies $\mathbb{E}_{H_0(g)}[\widetilde{E}(g)] \le 1$, and admitting exactly those candidates with $\widetilde{E}(g) \ge T_{\max}/\delta$ over a stream of at most $T_{\max}$ proposals gives* $\mathrm{FWER} \le \delta$ *and hence* $\mathrm{FGR}(A) \le \delta$ *at every cumulative index of the horizon.*

Proof of Theorem 1. The argument has four steps: a likelihood-ratio bound on the composed statistic, a Markov step converting it into a per-test level, a union bound over an adaptively generated horizon, and the passage from the resulting error event to the reported rate.

Step 1: $\mathbb{E}_{H_0(g)}[\widetilde{E}(g)] \le 1$. Write the per-round factor of Eq. (7) as

$$\ell(x_r) \;=\; \left(\tfrac{q_1}{q_0}\right)^{x_r} \left(\tfrac{1-q_1}{1-q_0}\right)^{1-x_r}, \tag{14}$$

which is the likelihood ratio between a Bernoulli(q_1) and a Bernoulli(q_0) law evaluated at x_r. Conditioning on $\mathcal{F}_{r-1}$ and writing $\pi_r = \Pr[x_r = 1 \mid \mathcal{F}_{r-1}]$,

$$\mathbb{E}\big[\ell(x_r) \,\big|\, \mathcal{F}_{r-1}\big] \;=\; \pi_r \,\frac{q_1}{q_0} + (1-\pi_r)\,\frac{1-q_1}{1-q_0}. \tag{15}$$

The right-hand side is affine in π_r with slope $\frac{q_1}{q_0} - \frac{1-q_1}{1-q_0} > 0$, since $q_1 > q_0$. It is therefore non-decreasing in π_r, so under Eq. (13) it is maximized at $\pi_r = q_0$, where it equals $q_0 \frac{q_1}{q_0} + (1 - q_0)\frac{1-q_1}{1-q_0} = q_1 + (1 - q_1) = 1$. Hence $\mathbb{E}[\ell(x_r) \mid \mathcal{F}_{r-1}] \le 1$ for every r, and the partial products $M_k = \prod_{r \le k} \ell(x_r)$ form a non-negative supermartingale with $M_0 = 1$. By the tower property $\mathbb{E}_{H_0(g)}[M_R] \le 1$. Since $S \in \{0, 1\}$ is determined by the sufficiency arm and $\widetilde{E}(g) = M_R \cdot S \le M_R$ pointwise, $\mathbb{E}_{H_0(g)}[\widetilde{E}(g)] \le 1$, which is Eq. (8). Note that only the conditional bound was used: the rounds need not be independent.

Step 2: per-test level. $\widetilde{E}(g)$ is non-negative with mean at most 1 under $H_0(g)$, so Markov's inequality gives, for the admission threshold $T_{\max}/\delta$,

$$\begin{aligned} \Pr_{H_0(g)}\big[\widetilde{E}(g) \ge T_{\max}/\delta\big] \;&\le\; \frac{\mathbb{E}_{H_0(g)}[\widetilde{E}(g)]}{T_{\max}/\delta} \\ &\le\; \frac{\delta}{T_{\max}} \;=\; \alpha_t. \end{aligned} \tag{16}$$

Each individual phantom is therefore admitted with probability at most α_t.

Step 3: union bound over an adaptively generated stream. Candidates are not fixed in advance: the guard proposed at index j is compiled from an episode that may itself depend on which guards were admitted earlier, hence g_j is a function of the history. Write $\mathcal{H}_{j-1}$ for the sigma-field generated by everything observed up to and including the proposal of candidate j, that is, all earlier verdicts, the library state they produced, and the compiled guard g_j itself, but nothing from the replay rounds that will test it. Taking $\mathcal{H}_{j-1}$ to include the proposal is what makes g_j, and hence its genuine-or-phantom status, measurable with respect to it; the candidate need not be a deterministic function of the earlier verdicts, which is why this covers a proposal step that carries fresh randomness of its own. The replay rounds for candidate j are executed only after g_j is fixed in this sense, and therefore Steps 1 and 2 apply conditionally on $\mathcal{H}_{j-1}$: on the event that index j holds a phantom,

$$\Pr\big[\widetilde{E}(g_j) \ge T_{\max}/\delta \,\big|\, \mathcal{H}_{j-1}\big] \;\le\; \alpha_t, \tag{17}$$

because Eq. (13) is itself a conditional statement and Markov's inequality applies under the conditional law. Taking expectations over the history gives the unconditional bound $\Pr[A_j] \le \alpha_t$, where A_j is the event that index j holds a phantom that is admitted. A stream spends at most $T_{\max}$ indices, so by the union bound

$$\Pr\Big[\bigcup_{j=1}^{T_{\max}} A_j\Big] \;\le\; \sum_{j=1}^{T_{\max}} \alpha_t \;=\; T_{\max} \cdot \frac{\delta}{T_{\max}} \;=\; \delta. \tag{18}$$

The union bound requires no independence across candidates, so the guarantee survives the adaptive, verdict-dependent generation of later candidates; this is the same device that lets the level be spent in advance.

Step 4: from the error event to the reported rate. Let $\mathcal{G} = \bigcap_j A_j^c$ be the event that no phantom is ever admitted; Eq. (18) gives $\Pr[\mathcal{G}] \ge 1 - \delta$, which is $\mathrm{FWER} \le \delta$. On $\mathcal{G}$ the realized proportion of admitted candidates satisfying H_0 is zero at every cumulative index simultaneously. Hence

$$\Pr\left[\frac{\sum_{g \in A} \mathbf{1}[H_0(g)]}{1 \vee |A|} = 0 \text{ at every cumulative index}\right] \;\ge\; 1 - \delta, \tag{19}$$

and since this proportion never exceeds 1, taking expectations gives $\mathrm{FGR}(A) \le \delta$ at every cumulative index. Retraction removes a guard from the installed library while its tombstoned entry is retained in the ever-admitted set that $\mathrm{FGR}(A)$ is computed over, which is why the ledger cannot increase $\mathrm{FGR}(A)$ relative to the bound established at admission. □

Proposition 2 (Conditional power at the alternative)**.** *Fix q_0, q_1, R and δ, and let*

$$x^* = \min\Big\{x \in \{0,\dots,R\} \;:\; \Big(\tfrac{q_1}{q_0}\Big)^{x}\Big(\tfrac{1-q_1}{1-q_0}\Big)^{R-x} \;\ge\; T_{\max}/\delta\Big\}, \tag{20}$$

with the convention $x^* = R+1$ when the set is empty, in which case no candidate is ever admitted and the power below is zero. Assume in addition that for the candidate under consideration the necessity rounds satisfy $\Pr[x_r = 1 \mid \mathcal{F}_{r-1}] = p$ for every r, and that the sufficiency indicator S is independent of the necessity outcomes $(x_1,\dots,x_R)$. Then, conditional on arm-S clearance, the candidate is admitted with probability $\Pr[X \ge x^*]$ for $X \sim \mathrm{Bin}(R,p)$, which is non-decreasing in p. At the primary setting $x^* = 11$ and this is 0.952 at $p = 0.7$ and 0.755 at $p = 0.6$.

Proof of Proposition 2. Because $q_1 > q_0$, the per-round factor $\ell(\cdot)$ of Eq. (14) takes the value $q_1/q_0 > 1$ when $x_r = 1$ and $(1-q_1)/(1-q_0) < 1$ when $x_r = 0$. Writing $x = \sum_{r=1}^{R} x_r$ for the total number of reproductions, the product depends on the round outcomes only through x:

$$\prod_{r=1}^{R} \ell(x_r) \;=\; \Big(\frac{q_1}{q_0}\Big)^{x}\Big(\frac{1-q_1}{1-q_0}\Big)^{R-x}, \tag{21}$$

and this expression is strictly increasing in x, since each additional reproduction multiplies it by the factor $\frac{q_1/q_0}{(1-q_1)/(1-q_0)} > 1$. Monotonicity makes the threshold a cut on x alone: the admission event $\{\widetilde{E}(g) \ge T_{\max}/\delta\}$ is, on $\{S=1\}$, exactly $\{x \ge x^*\}$ with x^* as defined in Eq. (20). If the set in Eq. (20) is empty, no value of x reaches the threshold, the admission event is empty, and the stated power is zero; the convention $x^* = R+1$ records this case. Otherwise, the added hypothesis $\Pr[x_r = 1 \mid \mathcal{F}_{r-1}] = p$ makes the rounds conditionally i.i.d. Bernoulli(p), and therefore their sum is $x \sim \mathrm{Bin}(R,p)$; and because S is assumed independent of $(x_1,\dots,x_R)$, conditioning on $\{S=1\}$ leaves that binomial law unchanged. The conditional admission probability is therefore the upper tail $\Pr[X \ge x^*]$, which is non-decreasing in p because the binomial family is stochastically ordered in its success probability, so a candidate with $p \ge q_{\mathrm{pow}}$ is admitted with probability at least the tail evaluated at q_{pow}. □

Why the power statement assumes more than the validity statement. Theorem 1 and Proposition 2 assume different things. Theorem 1 is a guarantee that must hold against an adversary, so it assumes as little as possible: only the one-sided conditional bound of Eq. (13), with no independence across rounds and no independence between the arms, which is why its proof runs through a supermartingale and a union bound, and not through any product law. Proposition 2 describes the operating point a designer chooses, and computing an exact power requires committing to a model of how a genuine candidate behaves. The two extra hypotheses do exactly that: a constant conditional reproduction probability p turns the rounds into a binomial count, and arm independence keeps that count binomial after conditioning on arm-S clearance. Neither is needed for, nor weakens, the validity claim: if either fails, the reported power figures cease to be exact but Theorem 1 continues to hold verbatim, since a phantom's admission probability is bounded through A1 alone. Validity therefore requires only A1, while power requires the two further hypotheses.

The primary operating point. At $q_0 = 0.15$, $q_1 = 0.6$, $R = 20$ and $\delta = 0.1$ over a horizon of $T_{\max}$ proposals, the per-round factors are $q_1/q_0 = 4$ and $(1-q_1)/(1-q_0) = 0.4/0.85$, and the least x satisfying Eq. (20) at the threshold is $x^* = 11$. This is the origin of the necessity-arm rate $x^*/R = 0.55$ used as the separating threshold in the motivation diagnostic: it is a consequence of the hyperparameters and the budget, not an independently chosen cut-off. Substituting into Proposition 2 gives conditional power 0.952 at $p = 0.7$ and 0.755 at $p = 0.6$, while a phantom capped at q_0 clears with probability at most 3.9×10^{-5}; these are exact binomial evaluations, not estimates, because the statistic is fixed before any data are seen.

Conditional power across the genuine band. The genuine population's per-guard reproduction rates concentrate between 0.6 and 0.8 (Appendix A.2). Across that range the admission probability is the upper tail $\Pr[X \geq x^*] = \sum_{k=x^*}^{R} \binom{R}{k} p^k (1-p)^{R-k}$ at the pinned $R = 20$ and $x^* = 11$, non-decreasing in p because the binomial family is stochastically ordered in its success probability, hence one row of it fixes the whole band, $\Pr[X \geq 11]$ of $X \sim \mathrm{Bin}(20, p)$: 0.755 at $p = 0.6$, 0.878 at $p = 0.65$, 0.952 at $p = 0.7$, 0.986 at $p = 0.75$ and 0.997 at $p = 0.8$, against 0.412 at $p = 0.5$ below the band. The operating point admits a genuine candidate with probability at least 0.755 anywhere in the band and above 0.95 across its upper half.

The statistic at the admission cut. At $R = 20$ and on arm-S clearance, the composed statistic $4^x (0.4/0.85)^{20-x}$ takes the values 65.7 at $x = 9$, 558.5 at $x = 10$, 4747.1 at $x = 11$ and 4.04×10^4 at $x = 12$. Consecutive values differ by the fixed factor $(q_1/q_0)/\big((1-q_1)/(1-q_0)\big) = 4/0.4706 = 8.5$, which is why the statistic advances in steps of 8.5 and cannot land near a threshold from both sides at once. Against the rejection threshold $T_{\max}/\delta = 2000$, ten reproductions fall short by a factor of 3.6 and eleven clear by a factor of 2.37, and therefore the cut $x^* = 11$ lands between two well-separated values of the statistic.

The ledger's exposure condition. Separately from the replay test, the causal reading of the ledger's marginal-benefit estimate in the post-admission component rests on the following condition.

Assumption 2 (Randomized exposure with known bounded propensity)**.** *On each triggering episode the exposure Z_i is drawn independently of the potential outcomes $\{u(\pi_{L+g}), u(\pi_L)\}$ at a propensity p_i that is known to the ledger and bounded away from 0 and 1.*

Under Assumption 2 the inverse-propensity contrast is unbiased for $b(g)$, not an average confounded by when the guard happens to fire. Because a fresh sequence is started whenever promotion changes p_i, each running mean aggregates exposures at a single known propensity, so no interval mixes two weighting regimes.

Scope of the guarantee. The scope is the admission decision itself. Two categories of candidate are excluded: those quarantined by the prefix-fidelity checks, which include sources that cannot be faithfully replayed, and those arising from degenerate-deterministic episodes whose necessity rounds carry no information. The exclusion set is fixed by these prespecified criteria and not chosen after the replay outcomes are seen, which is what keeps the horizon union bound of Theorem 1 valid over the tested set.

A.5 Diagnostic Construction

This appendix details the admission diagnostic, its size, the task families spanned, and the genuine-or-phantom labeling procedure that supplies ground truth for the FGR.

Why a purpose-built diagnostic is required. The FGR is the phantom share of the admitted set, so measuring it at all requires a candidate stream in which every guard's status is known independently of the admission decision. No existing agent benchmark carries such labels: they score task success, not the validity of the failure knowledge an agent accumulates. The diagnostic therefore fixes the label at construction time and treats admission as the quantity under test.

Scale and composition. The confirmatory horizon is 200 candidate guards per configuration, prespecified before any admission decision. Candidates are drawn in a 3:1 ratio of genuine to phantom, giving 150 genuine and 50 phantom guards per horizon; the 0.25 phantom share is the quantity an indiscriminate admitter inherits, which is why the always-admit reference realizes that rate in every column of Table 1 of the main paper, since it installs the whole stream without quarantine and inherits the fixed 0.25 phantom share on each benchmark. The number of candidates carried through replay per benchmark per seed is smaller than the horizon, because candidates that are quarantined or deterministic are removed before testing, at the ineligible shares of Table 11, hence $200 \times 0.85 = 170$ of the 200 reach the two arms per benchmark per seed at the pooled share. The 150 traces of the matcher audit are a disjoint human-labeled sample of episodes, used to validate the

reproduction predicate and never to propose guards. The horizon is evaluated in epochs of 50 candidates, giving the four epoch boundaries at which the cumulative trajectory of Table 5 of the main paper is read. Six confirmatory configurations arise from three replay-feasible benchmarks crossed with two frozen backbone families.

Task families. Candidates span the task families of the three confirmatory benchmarks. AppWorld contributes multi-application workflows over a versioned application database, covering stateful record manipulation, cross-application transfer, and constraint-violating requests. Terminal-Bench contributes shell and file-system tasks, covering destructive-command exposure, path and permission errors, and multi-step build and install sequences. The determinism-filtered SWE-bench-lite slice contributes repository-level program repair, covering test-suite regressions, dependency and environment faults, and patch application failures. Retaining all three keeps the diagnostic from over-fitting the guard language to a single failure vocabulary, and the per-benchmark breakdown of Table 4 reports each family separately, not only in aggregate.

Genuine candidates. A genuine candidate originates in an episode that actually failed. The episode is executed to completion, its failure is typed, and a guard is compiled at the identified locus. A candidate is admitted to the diagnostic as genuine only if the originating failure is attested twice over: the failure is observed in the original execution, and it recurs when the episode is re-executed from its versioned initial state with the guard withheld. This second condition is what makes the label a statement about the episode rather than about a single noisy run, and it is applied uniformly to every genuine candidate before any admission test is performed.

Phantom candidates. A phantom candidate is a syntactically well-formed, plausibly worded guard whose named failure has no causal basis in the source episode. Phantoms are constructed by three mechanisms in as near an equal split as the count allows, seventeen by locus displacement, seventeen by condition fabrication, and sixteen by outcome inversion, since fifty does not divide evenly into three. Locus displacement keeps a genuine failure description but attaches it to a different step of the same episode, which is why the guard names a real failure mode at a place it does not occur. Condition fabrication invents a precondition that the episode never violates, producing a guard that fires on a state the task cannot reach. Outcome inversion takes an episode that succeeded and asserts that it failed for a stated reason, and therefore the named failure never occurred at all. All three yield candidates that read as ordinary guards.

The adversarial stress condition. The stress test of Table 6 of the main paper strengthens the phantom population. Its 300 injected fabrications across 30 streams are drawn from the same three mechanisms, but each is additionally screened to survive a plausibility filter: a fabrication is retained only if it is not rejected by a text-only judgment of whether the guard is well-motivated. The resulting population is adversarial in the specific sense that matters here, namely that the evidence distinguishing it from a genuine guard is absent from the text and present only in the replay outcome.

Labeling procedure and its independence from the test. The label of every candidate is determined at construction time, by the mechanism that produced it. A guard compiled from an attested failure at its own locus is genuine; a guard produced by displacement, fabrication, or inversion is a phantom. No label is assigned by inspecting an admission verdict, and no candidate's label is revised after any admission decision. This matters because the natural objection to a replay-based diagnostic is circularity: if replay both created the labels and adjudicates them, the reported rate would be a tautology. Three properties break that loop. First, the phantom mechanisms are defined by construction operations on the source episode, not by replay outcomes; a displaced locus is a phantom whether or not it happens to reproduce. Second, the genuine attestation uses only the guard-withheld condition, whereas admission requires both the guard-withheld and the guard-injected arms, so the admission criterion is strictly stronger than the labeling criterion and a genuine label does not entail an admission. Third, the necessity threshold that admission applies, x^*/R, is set by the budget and the hyperparameters as shown in Appendix A.4, not by any property of the diagnostic, so the labeling procedure has no access to the quantity it would need in order to guarantee agreement. The residual dependence, that both label and test are defined over the same replay harness, is why the harness itself is validated separately: the typed matcher that decides whether a named failure recurred is checked against human judgment on a disjoint audit, reported as per-class F1 in Table 7.

Scope of the diagnostic. The diagnostic fixes the phantom share by construction, hence the absolute FGR values it reports are rates under a 0.25 phantom prior, not estimates of the phantom prevalence of any deployed system. The label is also a wider object than the null the theorem bounds: a candidate is a phantom by the mechanism that built it, whether or not its named class happens to recur, which is why a displaced guard whose class recurs inside the matcher window above the bound q_0 counts against the reported rate while sitting outside the reproduction bound the theorem controls, and the rate is read the conservative way on that account. It also inherits the replay-feasibility restriction: candidates whose prefixes cannot be faithfully reconstructed are quarantined, and the share so excluded is reported in Table 11.

A.6 Baselines and Configurations

This appendix lists the suppression-only, LLM-judge, static-released-guardrail, and external-memory references, their configurations, and the shared candidate budget and back-end under which every method is run.

Shared protocol. Every method in Table 1 of the main paper sees the same candidate stream in the same order, at the same 200-candidate horizon, under the same two backbone families and the same three seeds, and installs into the same guard library representation. Methods differ only in the admission rule: what evidence they consult and what they do with it. The replay back-end, the typing function, the compiler, and the decoding settings are identical throughout, and therefore no method is advantaged by a different candidate set, a different guard language, or a different inference stack. Methods that do not use replay do not call it; their candidate budget is not reallocated to other uses, so a cheaper admission rule shows up as lower cost and not as more attempts.

Suppression-only admission. The strongest same-family reference, adapted from the success-crediting practice of the Reflexion class (Shinn et al., 2023), which itself ships no admission gate, admits a candidate when the compiled guard suppresses the named failure in a single forward re-execution with the guard injected, which is the sufficiency arm alone. It is configured identically to EG: same compiler, same typing, same library. The difference is that it never runs the guard-withheld arm, hence it cannot observe whether the failure would have occurred anyway. This is why its FGR tracks the phantom share of the stream.

LLM-judge admission gate. The closest alternative paradigm replaces re-execution with judgment. The judge receives the typed failure record, the source episode transcript up to the locus, and the compiled guard text, and returns a binary admit-or-reject verdict with a short justification. It is decoded greedily at temperature 0 for reproducibility, runs on the same backbone family as the agent under test so that no capability gap is introduced, and is given a single call per candidate, matching the one admission decision per candidate that every other method makes. It does not receive replay outcomes: the comparison measures what text-level plausibility can and cannot separate.

Static released guardrails. The static reference adapts the release posture of the experiential-library class (Zhao et al., 2024), whose admission rests on textual plausibility rather than on a counterfactual replay check: the compiled guard set is published and installed wholesale, with no replay test at install time. It is included because it is the deployment practice a library-sharing ecosystem encourages, and it isolates the value of testing at all rather than the value of testing well. Because every candidate in the released set is installed, the method inherits the composition of what it was given: its FGR tracks the phantom share of the stream and its genuine retention is complete, since nothing is ever refused or withdrawn. Its position in Table 1 of the main paper is therefore close to the always-admit reference, and the informative contrast is not between them but between either of them and any method that conditions installation on evidence.

External-memory reference. The external-memory reference of the Reflexion class retains failure reflections in a memory that conditions later episodes, without compiling them into typed guards with trigger predicates. It is reported qualitatively, not as a row of the confirmatory table because its unit of accumulation is not a guard: there is no discrete admitted object whose genuine-or-phantom label could be read, and therefore the FGR is not defined for it under the metric of Appendix A.7.

Every retained reflection conditions all subsequent episodes and no prefix re-execution isolates its effect. We therefore record what it does and why the metric does not apply.

Constructed reference admitters. The three admitters marked † in Table 1 of the main paper, together with the naive-threshold variant of the validity ablation, are defined by the diagnostic itself and are not drawn from prior work, and they exist to bound and probe the comparison. The no-guard agent installs nothing: its FGR is zero by the empty-set convention, showing a zero rate is trivially attainable by refusing every candidate, so any working admitter must be compared against it on retention, not on FGR. The always-admit library installs every candidate without any test: it fixes the ceiling of the retention axis (1.00) and realizes the FGR of indiscriminate admission, the phantom share of the stream, so it bounds the comparison from the opposite side. The naive-threshold e-value variant keeps the full two-arm replay protocol but replaces the composed statistic with a fixed per-round threshold: it isolates the contribution of the composition itself, admitting each phantom with probability $1-(1-q_0)^R$, and doubles as the without-composition row of the validity ablation. The oracle admitter reads the ground-truth genuine-or-phantom label directly and admits exactly the genuine candidates: it is not implementable outside the diagnostic and serves only as the upper reference point (FGR 0.00, retention 1.00) against which EG's finalized retention gap of 0.08 is measured.

A.7 Evaluation Metrics

This section gives the metric definitions deferred from the setup. $\widehat{\mathrm{FGR}}(A)$ is the phantom share of the admitted set A, with the convention $\widehat{\mathrm{FGR}}(A) = 0$ when A is empty. The headline confirmatory value (0.08) is the instantaneous rate over the currently installed set, while the lifetime variant is computed over the ever-admitted set with tombstoned guards retained in the denominator (0.09, Table 9). The confirmatory requirement $\widehat{\mathrm{FGR}}(A) \le \delta = 0.1$ applies to both readings at every cumulative index. Genuine-guard retention is the fraction of ground-truth genuine guards installed and not retracted at the horizon, and each configuration's 200-candidate horizon contains 150 genuine guards. Retained utility is the mean task utility u of the agent acting with the admitted library, as a fraction of that with the oracle library, over the same confirmatory episodes. Aggregation across benchmarks and backbone families is as stated with each table, and typed-matcher agreement is per-class F1 against the 150-trace disjoint human-labeled audit. Intervals, where reported, are anytime-valid empirical-Bernstein confidence sequences valid at data-dependent stopping times. AppWorld is primary because it supports an exact versioned-database reset; the horizon is evaluated in epochs of 50 candidates.

The three reported quantities are defined as follows. The false guardrail rate is the phantom share of the admitted set, the admission analogue of the false discovery proportion of Benjamini & Hochberg (1995):

$$\widehat{\mathrm{FGR}}(A) = \frac{1}{1 \vee |A|} \sum_{g \in A} \mathbf{1}\big[g \text{ phantom}\big], \tag{22}$$

where A is the admitted set and the phantom label is the one fixed at construction. These labels also count displaced guards whose named class recurs above the bound q_0, forming a wider set than the null of Eq. (2); FGR and retention are aggregated over the six confirmatory configurations. Genuine-guard retention is the share of the genuine candidates in the horizon that are installed and not retracted at the horizon, the quantity a library that only accumulates would trivially maximize (Zhao et al., 2024):

$$\mathrm{Ret} = \frac{\big|\{g \in A :\ g \text{ genuine and installed at the horizon}\}\big|}{\big|\{g \in H :\ g \text{ genuine}\}\big|}. \tag{23}$$

Typed-matcher agreement is the macro-averaged F_1 over the six failure classes, macro-averaged so that a rare class cannot be masked by a common one (Sokolova & Lapalme, 2009):

$$\mathrm{F}_1^{\mathrm{macro}} = \frac{1}{|C|} \sum_{c \in C} \frac{2\,P_c R_c}{P_c + R_c}, \tag{24}$$

where C is the class set and P_c and R_c are the matcher's precision and recall on class c against the human-labeled audit.

A.8 Hyperparameters and Sensitivity

This section discusses the hyperparameters and the consequence of misspecifying the one the guarantee rests on.

Choice of the hyperparameters. The hyperparameters entering the admission test are fixed before any admission decision. The phantom bound $q_0 = 0.15$ is the operative one, since it is the hypothesis on which the guarantee rests. The round count $R = 20$ clears the registered power floor of 0.90 at $p = 0.7$ with conditional power 0.952. The necessity rate 0.55 is not a free parameter at all: it is $x^*/R = 11/20$, the reproduction count at which the composed statistic first reaches the admission threshold, and it moves only if q_0, q_1, R, $T_{\max}$, or δ moves. The matcher window admits a matched failure within two steps of the recorded locus, absorbing benign step-index drift between an episode and its replay while remaining far tighter than the typical distance between distinct failure sites.

Sensitivity to a misspecified bound. If a phantom's true per-round reproduction probability is $q > q_0$, the per-round factor of Eq. (14) is no longer bounded in conditional expectation by 1; it is bounded by $q\frac{q_1}{q_0} + (1-q)\frac{1-q_1}{1-q_0}$, so the composed statistic inflates by at most that factor per round and the phantom admission probability rises from the binomial tail at q_0 to the tail at q. The degradation is therefore graceful and quantifiable, not discontinuous: at the primary setting a phantom clears with probability 3.9×10^{-5} at $q = q_0 = 0.15$, and the tail rises smoothly as q increases, reaching the nominal per-test level 5×10^{-4} only once q approaches 0.2. The guarantee of Theorem 1 then holds at the inflated level rather than failing outright. This is why quarantine removes sources on which reproduction behavior is not trustworthy, and why the boundary conditions report the excluded share.

The tail under a misspecified bound. At a true per-round reproduction probability q that tail is $\Pr[X \geq 11]$ for $X \sim \mathrm{Bin}(20, q)$: 3.9×10^{-5} at $q = 0.15$, 1.26×10^{-4} at $q = 0.17$, 2.15×10^{-4} at $q = 0.18$, 3.54×10^{-4} at $q = 0.19$, 5.63×10^{-4} at $q = 0.20$ and 8.72×10^{-4} at $q = 0.21$. The nominal per-test level 5×10^{-4} is therefore first exceeded between $q = 0.19$ and $q = 0.20$, which is why the bound absorbs a misspecification of nearly a third of its own value before the level is spent. The same misspecification inflates each round's conditional expectation by at most $q\,(q_1/q_0) + (1-q)\,(1-q_1)/(1-q_0)$, which is 1.000 at $q = 0.15$, 1.106 at $q = 0.18$, 1.176 at $q = 0.20$ and 1.212 at $q = 0.21$: the supermartingale property degrades by a bounded factor per round, which is the sense in which the guarantee holds at an inflated level.

A.9 Extended Experimental Details

This appendix expands each experiment of the main paper. The headings follow those of the experimental section; each carries the extended reading.

Comparison against every admission rule. The six per-configuration FGR entries of Table 4 need not share a common installed-set denominator, since the ineligible share varies across configurations, and therefore the pooled entry is not an average of the rows by construction; at the precision reported the two coincide. Static release coincides with always-admit on every benchmark column because both install every proposed candidate. Suppression-only admission is the arm-N-removed configuration of the validity ablation. Its 0.25 row therefore doubles as the ablation reference. The LLM-judge FGR ranges from 0.14 to 0.19 across the three suites, the residue being fabrications that stay plausible in text; the gate has no direct evidence about that failure mode, since it does not receive replay outcomes. The retention of 0.92 sits 0.08 below the oracle's 1.00, refuting the reject-everything admitter, while the 0.08 FGR refutes always-admit; passing both directions at once is what separates a working admission rule from either trivial policy. Row means are consistent with the pooled values of Table 1 of the main paper (0.08 and 0.92) and the pooled ineligibility coverage loss (0.15); the ineligible share is taken over each configuration's 200-candidate stream and counts quarantined and degenerate-deterministic candidates.

Decomposition across benchmarks and backbones. Per-benchmark columns pool the two frozen backbone families; the pooled column aggregates all six confirmatory configurations. Were the pooled 0.08 an artifact of aggregation, some configuration would breach the budget, and Table 4 shows none does; the six FGR cells spread by at most ± 0.01. Difficulty surfaces exactly where the

Table 9: Post-admission ledger lifecycle: the exposure hyperparameters and the finalized retraction and promotion outcomes.

Quantity	Value
Probation exposure propensity	0.50
Post-promotion propensity	0.95
Minimum exposures before retraction	30
Exposures required for promotion	100
Marginal-effect increment bound (probation)	$[-2, 2]$
Marginal-effect increment bound (promoted)	$[-20, 20]$
Guards admitted (mean per configuration)	161
Retractions issued	11
Promotions certified	119
Median exposures at retraction	36
Median exposures at promotion	100
Ever-admitted FGR (tombstones in denominator) $\downarrow$	0.09
Budget δ	0.10

design routes it: the harder SWE-bench-lite slice quarantines more aggressively (ineligible share 0.18 against 0.12 on AppWorld) and carries the widest FGR cell, 0.09, still below δ. Reading the table therefore separates two things a pooled number cannot: where the design routes difficulty, and where it holds the level fixed.

Phantom-injection stress test. A breach is a stream admitting at least one phantom; every rule except EG breaches in all 30 streams. The rejection rule $K = 0$ has exact level $0.9^{30} \approx 0.042$ under the adverse pooled-marginal null of 0.10, with design power at least $(1 - 10 \times 5 \times 10^{-4})^{30} \approx 0.86$ under A1. Each column reports the count that its rule's per-candidate admission rate implies over the 10 streams per benchmark (10 injected phantoms each), which is why the entries carry a decimal count of guards. The suppression-only and LLM-judge rates are taken after the quarantine screen has removed the non-replayable share of the stream, so those two columns sit at 0.697 and 0.413 of the injected set; the naive-threshold rate tracks the per-phantom clearance $1 - (1 - q_0)^R \approx 0.96$ over the whole injected set, and is therefore read on the unscreened set rather than on the screened footing of the other two. The stress population also carries a higher quarantine rate than the confirmatory stream, about 0.30 against 0.15, because the condition-fabrication and outcome-inversion mechanisms that dominate plausibility-screened fabrications more often leave no faithfully replayable locus, which is why suppression-only admits about 0.70 of the 300.

What the zero count excludes. The stress result is a count of zero breaches over the 30 independently randomized streams, hence the exact one-sided Clopper–Pearson bound $1 - \alpha^{1/n}$ at $n = 30$ bounds the per-stream breach rate: below 0.095 with 95% confidence and below 0.142 with 99% confidence. The bound inverts the exact binomial: a per-stream breach rate r leaves all 30 streams clean with probability $(1 - r)^{30}$, which is why the rates not excluded at confidence $1 - \alpha$ are those with $(1-r)^{30} > \alpha$, that is $r < 1-\alpha^{1/30}$. The stream is the unit the design randomizes, and therefore the bound is stated per stream.

Matcher agreement. The audit uses the typed matcher window against the 150-trace disjoint human-labeled audit; the macro average reproduces the pooled aggregate 0.84, and the acceptance floor is 0.80 per class.

Horizon scaling. Cumulative FGR is computed over the set of guards installed at each epoch boundary (the instantaneous rate; the ever-admitted variant with tombstones retained is reported in Table 9), and retention over the genuine candidates proposed up to that boundary; the final column equals the pooled aggregates of Table 1 of the main paper.

Lifecycle and retraction. Marginal-effect increments are bounded by construction and tracked with an empirical-Bernstein confidence sequence valid at data-dependent stopping times. The hyperparameters read through the inverse-propensity estimator of Eq. (11). The probation propensity

Table 10: Full component ablation, complete form. Upper block: additive rungs. Lower block: removals from the full model. ‡: retention for these rows is the admitted library's retained utility. §: this removal leaves both columns unchanged and costs the unbiasedness of the benefit estimate. Challenge column: the challenge each row bears on.

Setting	FGR ↓	Retention ↑	Challenge
Additive ladder			
Typed compilation only	0.25	1.00	none
+ replay test (verdict frozen)	0.08	0.86‡	#1, #2
+ ledger (full EG)	**0.08**	**0.92**‡	#1–#3
Component removals from the full model			
without composed e-value (naive point threshold)	0.24	0.99	#1
without quarantine (prefix-divergence resampling)	0.18	0.95	#1
without typed matcher (LLM-judged locus; F1 0.68)	0.13	0.90	#1
without arm N (suppression-only admission)	0.25	0.92	#2
without arm S (no clearance check)‡	0.08	0.82	#2
without randomized exposure (observational)§	0.08	0.92	#3

of 0.50 is the balanced point that keeps the Horvitz–Thompson contrast Δ_i symmetric across its two arms and its per-exposure variance low, so early retraction evidence accrues as fast as randomized exposure allows. After promotion the propensity rises to 0.95, buying the utility of exposing a guard that has already certified positive benefit. The residual 0.05 of withheld episodes keeps the contrast identified and holds each per-exposure contrast inside the promoted increment bound of Table 9, so the wider weighting is bounded by design. The asymmetric thresholds follow the same logic: retraction is certifiable after 30 exposures, whereas promotion waits for 100 exposures so the interval can certify a strictly positive sign before the propensity widens. Certification is a running process: promotion requires 100 exposures, hence a guard admitted late in the horizon is still accruing them when the horizon ends, which is why 119 of the 161 admitted guards are certified, 11 are retracted, and the remaining 31 are still on probation at the boundary. Because a fresh sequence starts at promotion, the change of propensity never mixes two weighting regimes inside one running mean, which is why every retraction and promotion behind the finalized 0.09 ever-admitted rate is decided at a single known propensity per guard.

Additive ladder. The upper block adds one component at a time and the lower block removes components from the full model, and therefore the two blocks meet at the same system: typed compilation alone installs every typed candidate without a test, which is exactly the static-released-guardrail policy of Table 1 of the main paper, and the verdict-frozen rung is the full model with retraction disabled. Utility-side deltas are reported against the full-model retained utility of 0.92.

Component removals. Because every guard in the diagnostic carries a ground-truth label, the removals of Table 10 give directional evidence on where the admission evidence comes from, each row consistent with one piece screening the failure mode it was designed for. Table 2 inflates FGR on every validity removal, but by amounts that rank the defenses: removing the composed statistic collapses FGR to essentially the phantom stream share (naive threshold: 0.24, since a phantom clears a per-round bar with probability $1-(1-q_0)^{20} \approx 0.96$ and nearly the whole phantom stream enters the admitted set), followed by faithful replay (quarantine off: 0.18, via selective censoring) and the typed matcher (LLM locus: macro F1 0.68, under the floor), while removing the necessity arm collapses it to the full phantom stream share (suppression-only: 0.25). The ranking matches the design argument: composition defends against accumulation across rounds, necessity against crediting non-causal successes, and each maps to the challenge its component was built for. The utility removals degrade retained utility or the benefit estimate without touching FGR, the mirror image of the validity side: dropping arm S admits guards that never clear the named failure, which costs the admitted library 0.10 of its retained utility (-0.10, to 0.82), freezing verdicts leaves non-beneficial guards installed (-0.06), and replacing the randomized exposure coin with observational attribution biases marginal benefit by 0.03 in absolute units, half the retraction effect, enough to misorder borderline retractions. Retained utility is carried on the same unit interval as retention, with the full model at 0.92 and the arm-S removal at 0.82; dropping arm S opens a conjunctive

gate and so cannot refuse a candidate the full test admits on the same replay outcomes. That no utility removal moves FGR is directional evidence, consistent with error control and usefulness being enforced by disjoint components, as the framework overview claims. In the naive-threshold row, the per-round likelihood-ratio factors are $q_1/q_0 = 4.0$ and $(1-q_1)/(1-q_0) = 0.471$, and a single-round exceedance occurs with probability $q_0 = 0.15$ under the null, which yields the per-phantom clearance probability $1-(1-0.15)^{20} = 0.96$; with the genuine stream clearing at 0.99, this converts to the admitted-set FGR of $50 \times 0.96/(50 \times 0.96 + 150 \times 0.99) = 0.24$ reported in the table. Necessity screens phantoms, sufficiency screens guards that do not clear their named failure, and only their conjunction holds both failure modes at once.

Uncertainty of the point values. The quantities reported here are the FGR, genuine-guard retention, and typed-matcher agreement, each of which describes the admission layer. The entries of Tables 1, 5, 6, 7, and 4 are point estimates carrying no interval, standard error, or significance test, apart from the stream-level rejection rule reported with Table 6. The anytime-valid confidence-sequence semantics fixed in Appendix A.7 govern the ledger's running marginal-effect estimates, not these table entries.

Resolution of a rate at this horizon. The candidate count a given resolution requires is fixed by the rate itself, through the binomial reading $n^\star = z^2 p(1-p)/h^2$ at $z = 1.96$: on the ever-admitted reading, whose rate is 0.09 and whose denominator is the 161 guards admitted per configuration, a half-width of 0.05 needs 126 candidates, 0.02 needs 787, and 0.01 needs 3147. One horizon's 161 admitted guards therefore carry the rate at the first of those resolutions, and the eighteen confirmatory runs together at the second.

Scope of the arm-independence hypothesis. Proposition 2 takes the sufficiency indicator S to be independent of the necessity rounds, which is what keeps the round count binomial after conditioning on clearance. The motivation diagnostic places the phantom population below the per-round bound with only a thin tail crossing it, so for the great majority of phantoms the sufficiency arm clears for the same reason the necessity rounds fail to reproduce the named failure; on that population both arms are driven by the same reproduction behavior, hence independence is not licensed there. The hypothesis is restricted to the genuine population whose power the proposition describes.

A.10 Reproducibility and Pinned Configuration

This appendix documents the guard language, seeds, decoding settings, and versioned initial state under which admissions reproduce; it fixes what a replication would need to hold constant, while the result tables themselves are collected in Appendix A.9. The per-configuration breakdown of the primary comparison, over the three confirmatory benchmarks and two frozen backbone families reported in Table 4, aggregates to the pooled FGR and retention figures of Table 1 of the main paper at the precision quoted there.

Backbone families. The two frozen backbone families are `Qwen/Qwen2.5-72B-Instruct` (72B parameters) and `meta-llama/Llama-3.3-70B-Instruct` (70B parameters). Both are open-weight instruction-tuned models run at bfloat16 precision, and neither is updated at any point: no gradient is computed anywhere in the evaluation, which is why a backbone enters as a fixed conditional distribution over actions. Two families are used so that a result cannot rest on the idiosyncratic failure vocabulary of a single model, and both are held identical across every method in Table 1 of the main paper, and therefore the comparison isolates the admission rule, not the backbone.

Guard language and compiler. Guards are expressed in a small typed domain-specific language (DSL) whose trigger predicates are conjunctions of state tests over the observable episode state: object and file identifiers, tool names and arguments, exit statuses, and the six failure classes of the typing function. The compiler emits predicates only from this closed grammar and is run at temperature 0, so compilation from a fixed typed record is deterministic and a candidate guard is a pure function of its source record. Compiler determinism alone does not reproduce a horizon, since a candidate depends on the episode that produced it and a verdict depends on the replay that tested it. Reproduction is instead a property of the whole chain being pinned: the episode stream is regenerated from its recorded seed against the same frozen backbone, each episode is restored

Table 11: Boundary and robustness audits: finalized values against their acceptance criteria.

Audit	Value	Criterion	Status
Post-replay contamination ↓	0.00	below audited threshold	met
Leaked-prefix candidates admitted	0	zero	met
Ineligibility coverage loss ↓	0.15	descriptive only	reported
Reproduction-rate drift against q_0 and q_1	0.05	descriptive only	reported
Qualitative trace match rate (sampled guards) ↑	1.00	full match on the sampled set	met

from its versioned initial state and never re-derived, the typing function and compiler are frozen and greedily decoded, the replay rounds draw from their own recorded seeds, and any candidate whose prefix fails to reconstruct is quarantined. Under those conditions, rerunning the horizon under the recorded seeds and the pinned configuration reproduces the same candidate set and the same admission verdicts. Reproduction is reported for that fixed configuration.

Seeds and replicate structure. Each of the six confirmatory configurations is run at three seeds, giving eighteen confirmatory runs. Seeds enter at two separated points, the episode stream seed and the replay seed, drawn from disjoint ranges so that a replay never re-uses the randomness that generated the episode it is replaying, which is the seed separation the admission protocol requires. The episode-stream seeds are 1001, 1002, and 1003, and the replay seeds are 9001, 9002, and 9003. Ablations are run at a single seed, and are therefore reported as directional and not confirmatory.

Decoding and judgment settings. The agent policy is decoded at temperature 0.7 with nucleus sampling at $p = 0.95$ and a $4{,}096$-token completion cap, the setting under which the episode streams are generated for every method alike. Every component that must be reproducible is instead decoded greedily at temperature 0: the typing function, the guard compiler, and the LLM-judge baseline. The LLM-judge baseline receives the failure record, the transcript up to the locus, and the compiled guard and is asked for a single admit-or-reject verdict with a brief justification; it sees exactly the text a plausibility-based admitter would have, and in particular it does not see replay outcomes, which is the comparison the paradigm contrast requires. All backbones are served through the same inference stack, so no method benefits from a different serving configuration or a different context window.

Software environment. The runtime requires Python 3.11 or later. Benchmark harnesses are pinned by commit and not by release tag, and the versioned initial state of each episode is stored as a database or filesystem snapshot so that a replay restores the exact pre-locus state and not an approximation of it. The evaluation is inference-only, with no gradient computation at any point, hence the cost is dominated by replay and not by generation.

Benchmark versions and the determinism filter. AppWorld is used at its released evaluation split with the versioned application database that supports exact reset; Terminal-Bench is used at its released task set. The SWE-bench-lite slice is filtered for determinism before use, because a replay-based test is only meaningful on sources whose behavior is stable under re-execution. The filter admits an instance only if its test outcome is identical across three independent executions of the unmodified repository at the pinned commit, with network access disabled and timestamps and randomness pinned; instances whose outcome varies, whose test suite depends on network resources, or whose environment cannot be rebuilt reproducibly are removed. Removal happens before any admission decision, which is why the filter cannot interact with the comparison. No train, validation, or test split is defined here.

A.11 Analysis and Boundary Conditions

We next analyze the conditions under which EG withholds its guarantee; Table 11 collects the finalized boundary audits.

Contamination, ineligibility and drift. Pre-locus hash-divergence and failed double-prefix probes are quarantined, leaving no leaked-prefix candidate in the primary admitted library: the

recorded post-replay contamination is 0.00. Degenerate-deterministic and non-replayable candidates are excluded; the coverage loss is 0.15 of the candidate stream, and exclusion costs coverage, not error control. The observed reproduction-rate drift against the $q_0 = 0.15$ and $q_1 = 0.6$ hyperparameters is 0.05 across the three streams. The registered criteria are: post-replay contamination below the audited threshold with zero leaked-prefix admissions; ineligibility coverage loss and drift reported descriptively; and the qualitative trace match, confirmed.

Reproducibility and traceability. The guard DSL, frozen typing function, seeds, and configurations pin the admission pipeline, and a qualitative study of sampled admitted, quarantined, and retracted guards matches every decision to its prespecified trigger, intervention, provenance pointer, and tier. Details in Appendix A.10.

A.12 Discussion

Having formalized admission and evaluated it, we step back to ask why this particular machinery answers the three challenges, and what its guarantee delivers.

Admission is a causal question, not a plausibility one. The central move of EG is to treat a self-authored guardrail as making two falsifiable claims, that the named failure recurs without it and clears with it, and to admit only on counterfactual replay of both arms. This turns a self-report, which an agent editing its own memory can always keep internally consistent, into a claim that an external re-execution can refute. A plausibility gate that scores the record without replay (Zheng et al., 2023) judges the story the agent tells, whereas the necessity and sufficiency arms judge the world the guard acts in, which is why suppression evidence alone, however convincing, cannot stand in for the necessity arm.

Accumulation is safe only under lifetime error control. A guard that looks sound in isolation says nothing about the phantom share of a library grown over hundreds of proposals. Casting admission as a fixed-level test inverted over a prespecified horizon is what converts per-guard evidence into a family-wise bound on the FGR. Keeping tombstoned guards in the denominator of the realized rate is what keeps a retracted guard inside the ever-admitted rate.

The guarantee is conditional. EG earns a bounded phantom rate only where its preconditions hold: faithful prefix replay, a typed matcher that agrees with human judgment on class and locus, and assumption A1 on the phantom reproduction tail. Where a candidate cannot be replayed or is deterministic, the design withholds the error-control claim and excludes the candidate instead of admitting it on weaker evidence, which is why the cost is recorded as coverage loss and not folded into the rate.

The deliverable is an auditable library, not a score. Because the typing function, the compiler, the seeds, and the guard language are fixed before any decision, every admission and retraction can be re-derived rather than trusted on report. What the method leaves behind is a library a third party can replay and contest guard by guard.

A.13 Future Work

Four directions follow directly from the eligibility boundary. Relaxing assumption A1 would extend admission to failure modes whose reproduction tail is heavier than the phantom bound, and would let the framework speak to stochastic episodes that today fall outside faithful replay. The typing function and matcher could extend to richer or learned-but-audited failure taxonomies, so long as the deterministic agreement floor is preserved, widening coverage without weakening the reproduction predicate. Beyond the evaluated task families, generality across additional frozen backbones and distribution shifts is the next thing to establish, together with horizons longer than the one priced here. Finally, the confidence-sequence lifecycle could extend to libraries shared across cooperating agents, where exposure evidence accrues in parallel and retraction must remain valid under that dependence.