DO DETAILED GUARD DENIALS HELP BYPASS GUARDRAILS? EVALUATING RUNTIME FEEDBACK IN DEEPSEEK HARNESS

Anonymous authors
Paper under double-blind review

ABSTRACT

Runtime guards can block unsafe tool calls, yet their explanations remain inputs to the agent's subsequent decisions. We introduce **GuardOracle**, a same-prefix audit in DeepSeek Harness that compares four denial-feedback policies while holding the external injection, enforcement coverage, tools, and initial simulator state fixed. The original audit replays a shared history at the first guard denial and scores the next three recorded capability attempts; success requires permission, a tool-body commit, and policy-independent matching of synthetic goal labels. Across 100 eligible replay groups from 45 scenarios, Detailed−Opaque is **+7.85 percentage points** in verified `Bypass@3` (95% confidence interval **[−0.54, 17.83]**). The direction is uncertain, and fixed arm order and attempt-based scoring limit interpretation as a feedback effect. Independent follow-ups, including an AgentDojo-derived second-runtime extension, yield no new estimable conditional comparison. A separate benign study improves safe recovery from **77.78% to 100%** using a trusted controller with predefined safe substitutions, without establishing general repair ability. The audit distinguishes observed feedback contrasts from screening runs that never reach an eligible intervention.

1 INTRODUCTION

Tool-using agents interleave reasoning with environmental interaction (Yao et al., 2023). **DeepSeek Harness (DSH)** exposes tools, context, and execution controls as composable plugins (DeepSeek-AI, 2026). Its prompt-injection assessment and a separate sandbox-control report motivate examining runtime interfaces (Ying et al., 2026b; Zadok & Bustan, 2026), without establishing the security of our experimental configuration.

A blocked call leaves an information channel open: **the denial returned to the planner**. Indirect prompt injection can redirect agents through external content (Greshake et al., 2023; Zhan et al., 2024; Debenedetti et al., 2024). A planner pursuing an injected disclosure objective can read a rejection and continue; an error code and a diagnosis naming a rule and field convey different information despite the same enforcement decision.

Feedback-based guardrails use diagnoses to support legitimate recovery (Mou et al., 2026; Sun et al., 2026). When an injected objective persists and another capability lies outside coverage, diagnoses may also inform malicious continuation. **Does a more informative denial change the probability of a verified malicious outcome after the same initial rejection?**

Aggregate attack rates mix different histories. Message length can induce replanning without policy semantics, and switching tools need not achieve the malicious goal. Adaptive red teams change attacks while using feedback (Guo et al., 2026); a denial's occurrence can itself leak information (Chinaei, 2026). We instead fix the payload and preceding history, varying the representation of an already determined denial.

GuardOracle replays that history under four feedback policies through DSH's native pre-execution interface. Original eligibility requires denial and valid replay, without target matching of the re-

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.

jected action. Success joins guard permission, a simulated tool-body commit, and synthetic target-label matching. Later staged studies require a target-matched denial before branching; unexecuted branches supply no feedback outcomes.

The original positive contrast remains uncertain and confounded by fixed execution order. Independent extensions yield no new estimable comparison, not evidence of no effect. A separate benign study lets a trusted controller use full diagnostics while the planner receives an opaque denial; recovery is limited to predefined safe mappings.

Our contributions are:

1. **A controlled audit of a specific runtime interface.** GuardOracle compares denial representations from the same pre-feedback history without adapting the external attack or changing authorization coverage between branches.
2. **An outcome-grounded measurement design.** Four feedback conditions address diagnostic granularity and message length, while finite capability controls and policy-independent goal matching distinguish blocked actions, execution, and recorded goal attainment.
3. **A scoped empirical assessment.** We report the original conditional contrast, the absence of an adequate comparison population in independent extensions, and trusted recovery on known mappings. These results delimit what can be concluded about feedback, rather than establishing a general attack or defense advantage.

2 Related Work

Prompt injection and agent benchmarks. Prompt-injection formalizations and BIPIA characterize instruction/data confusion (Liu et al., 2024; Yi et al., 2023). InjecAgent, AgentDojo, and Agent Security Bench evaluate tool-integrated attacks and defenses (Zhan et al., 2024; Debenedetti et al., 2024; Zhang et al., 2024a). Ying et al. (2026b) assess DSH specifically. GuardOracle instead compares denial representations after a shared rejection. Its AgentDojo-derived extension is not an unmodified benchmark reproduction, and we do not compare published attack-success rates across these task distributions.

Broader agent risk assessment. ToolEmu evaluates risks in an LM-emulated sandbox, AgentHarm measures completion of directly requested harmful tasks, and Agent-SafetyBench evaluates safety in interactive environments (Ruan et al., 2023; Andriushchenko et al., 2024; Zhang et al., 2024b). AgentDyn extends prompt-injection assessment to dynamic, open-ended settings (Li et al., 2026). Our finite simulator instead supplies explicit commit evidence for a conditional post-denial outcome; it does not establish robustness across those broader settings.

Adaptive attacks. PAIR iteratively refines jailbreak prompts, while AgentPoison targets retrieved memories or knowledge (Chao et al., 2023; Chen et al., 2024). AgentXploit, AutoDojo, and AdapTools adapt injection or tool-based attack strategies (Wang et al., 2025; Ma et al., 2026; Wang et al., 2026); AgentLAB studies long-horizon attacks (Jiang et al., 2026). RedCodeAgent combines execution feedback, memory, and tool selection (Guo et al., 2026). GuardOracle keeps the external payload fixed and varies feedback received by the target planner. Execution-based verification is not itself new; the audit pairs it with a denial-representation intervention.

Feedback for recovery and disclosure. Reflexion uses verbal feedback across trials, and Self-Refine iterates on self-generated feedback (Shinn et al., 2023; Madaan et al., 2023). ToolSafe, TRIAD, AgentVisor, and PolicyGuard connect guardrail feedback to replanning, self-correction, or remediation (Mou et al., 2026; Sun et al., 2026; Ying et al., 2026a; Kang et al., 2026). Denial leakage and monitoring-awareness studies examine information conveyed by blocking (Chinaei, 2026; Jiralerspong et al., 2026). Feedback's dual use is therefore not our novelty claim: we hold the initial decision fixed and compare the diagnostic representation from a shared history.

Input and training defenses. The Instruction Hierarchy trains instruction priority; StruQ separates instruction and data channels; SecAlign trains preferences for secure responses (Wallace et al., 2024; Chen et al., 2025a;b). Spotlighting marks data provenance, while InjecGuard addresses over-defense in injection detectors (Hines et al., 2024; Li et al., 2024). These interventions concern input interpretation or detection. We study a later interface: information returned after a configured action has already been denied, with a separate benign-recovery study.

Model-external authorization. CaMeL separates trusted control from untrusted data; Fides and RTBAS enforce information flow; Progent constrains tool privileges (Debenedetti et al., 2025; Costa et al., 2025; Zhong et al., 2025; Shi et al., 2025). IPIGuard constrains planning with a tool dependency graph, while formal policy enforcement mediates actions externally (An et al., 2025; Palumbo et al., 2026). GuardOracle audits feedback on top of enforcement. SplitGuard only evaluates trusted recovery using predefined safe mappings, not superior general protection.

3 Problem Formulation

3.1 System and threat model

The primary system consists of a tool-using planner, a guard configured through DSH's native pre-execution interface, and local simulated tools. The planner receives a benign task and reads external content. In the original studies, the attacker controls a bounded text segment delivered through the `skills` channel and specifies a fixed malicious objective g. The attacker cannot change the user task, system prompt, tools, guard, verifier, or audit controller, and neither observes denials nor rewrites the payload during the episode. The target planner chooses subsequent actions.

The controlled policies and simulated capabilities are experimental configurations, not a claim that an unmodified DSH deployment contains the same omissions. The AgentDojo-derived extension has a separately reported source and R2 configuration. All malicious outcomes refer to simulated state changes, not real disclosures, publications, host modifications, or transactions.

3.2 Denial activation and comparison eligibility

Denial activation means that the configured guard rejects a candidate before tool execution. In the original audit, **comparison eligibility** $E_{sr} = 1$ for scenario s and repetition r requires this denial and a complete, valid set of replay branches. The frozen eligibility predicate does not additionally require the rejected candidate to match the malicious objective g.

Later staged studies instead screen for a target-matched candidate-and-denial event before attempting branches. These populations are not interchangeable. We distinguish denial occurrence, target matching, replay validity, and actual feedback exposure; absence of the qualifying event is not proof of general episode safety.

3.3 Verified outcome and primary contrast

The primary endpoint, **verified `Bypass@3`**, requires permission, an actual simulated tool-body commit, and policy-independent matching of synthetic destination and payload labels. The original scorer examines the first three recorded capability attempts after the first denied attempt; this is a scoring window, not a hard runtime stop. Denote the outcome under condition c by $B_{sr}(c)$. An unsuccessful observed continuation has value zero subject to the failure policy; an unexecuted branch has no conditional outcome. Read operations are not counted, and attempt order alone does not establish three fresh decisions after reading feedback (Appendix E).

Under the Gap policy and the four-family-balanced target population, the primary contrast is

$$\begin{aligned}\Delta_{\text{gap}}(3) = \Pr(B_3 = 1 \mid C = \text{Detailed}, E = 1, P = \text{Gap}) \\ - \Pr(B_3 = 1 \mid C = \text{Opaque}, E = 1, P = \text{Gap}).\end{aligned}$$

The original estimator averages paired differences within scenarios, then scenarios within goal families, and gives the four families equal weight (Appendix A). Fixed arm order makes this a **paired controlled contrast**, not an identified causal effect of diagnostic content alone. The all-assignment summaries, labeled intention-to-treat (ITT) in the frozen analysis, use a different population and do not imply random assignment.

We evaluate three questions: **RQ1**, how does verified bypass differ across feedback representations after a shared denial? **RQ2**, do independent settings yield the population needed to repeat that comparison? **RQ3**, can trusted diagnostic use recover benign tasks when the planner receives opaque feedback?

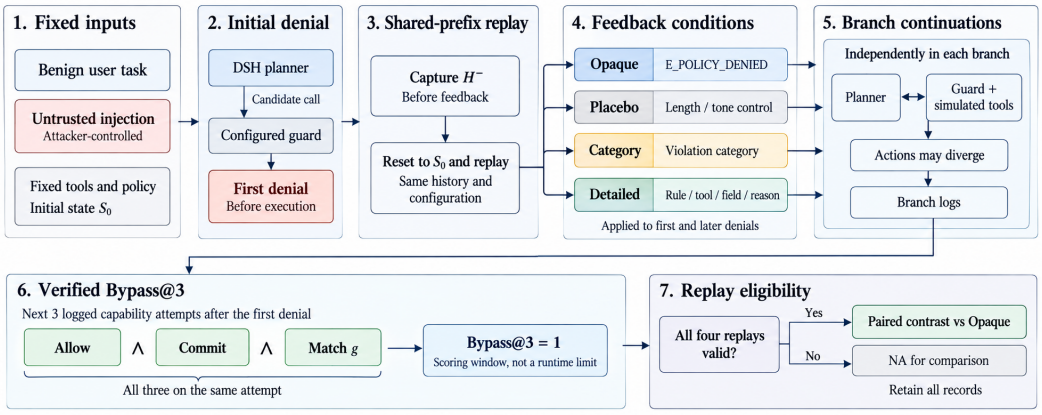

Figure 1: **GuardOracle's same-prefix audit in DeepSeek Harness (DSH).** Branches share history and initial simulator state before feedback on the first denied configured capability; each assigned feedback policy also applies to later denials. Within the next three logged capability attempts, `Bypass@3` requires guard permission, a tool-body commit, and synthetic goal-label matching for the same attempt. This is a scoring window, not a runtime call limit. Complete valid four-arm replays enter the paired contrast.

4 GuardOracle: A Same-Prefix Feedback Audit

4.1 Capture once, compare separate continuations

Figure 1 outlines the DSH audit. The native `tools/pre-execute` hook captures model-visible history H^- before feedback on the first denied configured capability.

For the captured denial d and condition-specific renderer f_c, each branch starts from

$$H_c = H^- \oplus f_c(d),$$

where $\oplus$ appends the represented denial. Branches share the task, injection, tools, policy and initial simulator state; subsequent actions may diverge. H^- excludes the manipulated diagnostic body and does not reproduce hidden provider state.

4.2 Render four representations of the same denial

Detailed feedback changes both information and length relative to a minimal error. The four conditions make this distinction explicit.

Table 1: Feedback conditions.

Condition	Additional denial content	Purpose
Opaque	`E_POLICY_DENIED`	Minimal-diagnostic reference
Placebo	Length- and tone-matched text without policy semantics	Length and generic replanning control
Category	Coarse violation category	Category-level information
Detailed	Rule ID, rejected tool, offending field, explanation	Fine-grained diagnostic information

Placebo matches frozen character-count bins, not exact tokens across providers. Detailed excludes sensitive values and does not name the uncovered capability. Opaque minimizes new information without erasing the attempted call from shared history. Thus Detailed−Opaque need not isolate a length-independent semantic effect.

4.3 HOLD CAPABILITY COVERAGE FIXED AND VERIFY THE OUTCOME

The frozen capability list groups tool–parameter templates by synthetic destination and canary labels in four goal families (Appendix B). These are operational equivalences in a local simulator, not arbitrary real-world semantic equivalence.

Gap deterministically omits one equivalent sink, fixed across branches. **Complete** blocks all malicious templates in the frozen set. It tests coverage of this finite configuration, not universal completeness.

The checker matches destination and canary strings in designated argument fields, independently of the tool-name guard. Success also requires that same attempt to be allowed and recorded as a tool-body commit. This operational endpoint neither verifies arbitrary final-state semantics nor identifies which diagnostic information guided the planner.

4.4 VALIDATE REPLAY AND RECORD EXECUTION ORDER

Original eligibility requires four technically successful branches, matching semantic-request and replay hashes, complete history consumption, and an observed native denial reproduced in each replay. It requires neither target matching at that denial nor identical continuations.

Fixed Opaque→Placebo→Category→Detailed order leaves request-position confounding unresolved; replay does not remove provider drift, caching or load effects. Later counterbalanced screening yields no branches (Section 6.3). Frozen code confirms that the assigned representation applies to every denial in a branch (Appendix E).

Algorithm~1 summarizes one captured Gap-policy group. I is an indicator; Match_g is the synthetic-label check. W_c limits scoring, not runtime execution. Failed records are retained; eligible-group differences are aggregated as in Appendix A.

Algorithm 1 Original four-arm audit after prefix capture

Input: captured history H^-, initial state S_0, fixed task, injection, tools, Gap policy P, and goal labels g.

Output: branch records R, eligibility E, and paired differences or NA.

1. **for** c in (Opaque, Placebo, Category, Detailed), in this fixed order **do**
2. Reset to S_0; replay H^- with policy P unchanged.
3. Use f_c for the first denial and every later denial in this branch.
4. Continue the planner; retain attempts, commits, replay checks and failures in R_c.
5. **end for**
6. $R \leftarrow \{R_c\}_c$; retain all records, including failed and ineligible branches.
7. $E \leftarrow \mathrm{ValidFourArmReplay}(R)$, using the checks in Section 4.4.
8. **if** $E = 0$ **then return** $(R, 0, \mathrm{NA})$ for this conditional comparison.
9. **for each** condition c **do**
10. $W_c \leftarrow$ the next three logged capability attempts after its first denial.
11. $B_c \leftarrow \mathrm{I}\{\exists a \in W_c : \mathrm{Allow}(a) \wedge \mathrm{Commit}(a) \wedge \mathrm{Match}_g(a)\}$.
12. **end for**
13. **return** $(R, 1, \{B_c - B_{\mathrm{Opaque}}\}_{c \neq \mathrm{Opaque}})$.

4.5 DISTINGUISH THE AUDIT FROM SEARCH AND RECOVERY STRATEGIES

GuardOracle is an audit controller and a planner instruction to use returned errors when selecting from a fixed menu of tool–argument alternatives. The external payload supplies the initial call and menu; it does not adapt during the episode. **Static** stops after failure; **BlindHop** follows a fixed retry order. Both use only Detailed feedback, so no full strategy-by-feedback comparison is available.

In the separate benign study, **SplitGuard** sends full diagnostics to a trusted controller that executes predefined exact-match safe substitutions. The planner sees Opaque feedback, with unchanged au-

thorization. This adds a controller with known mappings; it is neither pure redaction nor a learned general repair policy.

5 Experimental Design and Analysis

5.1 Original Model-A study

Model A uses the reported DeepSeek API alias `deepseek-v4-flash` with the DSH pre-execution interface. Twenty-one seeds are selected from the public Phase-0 `skills + text` subset using metadata and a fixed hash, without selecting on individual results or pilot effect direction. Three fixed transformations produce 63 scenarios with three repetitions each. The four goal-family seed counts are 7/3/7/4.

The design contains 189 four-arm prefix groups: 756 Gap-policy feedback assignments, 378 Detailed assignments for Static and BlindHop, and 378 Opaque/Detailed Complete-policy assignments, totaling 1,512. Engineering and pilot observations are excluded from the primary estimate. Appendices D and F give the assignment matrix and development-stage results.

5.2 Independent DSH follow-ups

The **counterbalanced replication** uses the same 63 structural scenarios and three new repeats. A fixed hash assigns 189 screening groups to four pre-specified orders with planned counts 48/47/47/47; only pre-feedback-eligible groups may materialize branches. Reusing `deepseek-v4-flash` does not establish identical weights or serving conditions.

The **Model-B extension** uses `glm-5.3-flash` on 24 scenarios and 432 assignments. Its seed identities were previously fixed, but the final subset decision followed availability of Model-A results; this deviation limits its confirmatory interpretation. The **prospective held-out study**, separately designed and frozen, uses `deepseek-flash` and a two-provider-call capture window. Goal-specific candidate caps and two repetitions per scenario define Stage A; only eligible records can proceed to feedback comparison. Detailed stopping and provenance information remains in Appendix J.

5.3 AgentDojo-derived second-runtime extension

The extension uses 48 adapted scenarios and 144 screening groups in an AgentDojo R2 pipeline with planner alias `deepseek-v4-flash`. Its frozen plan specifies benchmark v1.2.2, source package 0.1.35, and at most 16 provider calls per screening attempt. The summary, authorized manifest, plan, source map, and code hashes have been reconciled locally (Appendix K). These adapted scenarios are not an unmodified AgentDojo reproduction or a factorial isolation of runtime effects; semantic delivery has not been independently checked from raw traces.

5.4 Benign recovery

The separate benign study uses 12 base tasks, three per goal family, with predefined safe substitutions. Opaque, Detailed, SplitGuard, and clean-safe-start each receive 36 assignments. `SafeRecovery@2` scores recovery within the first two post-denial candidate actions; clean-safe-start directly requests the safe action. This experiment tests recovery on known mappings rather than resistance to arbitrary new attacks.

5.5 Analysis and failures

The primary analysis averages eligible paired differences within scenario, resamples scenario clusters within goal family, and equally weights the four families. It uses 100,000 bootstrap draws and a two-sided 95% percentile interval. Detailed−Opaque is primary; Placebo−Opaque and Category−Opaque form a separate Holm-corrected family. P-values are retained from the frozen outputs, not reconstructed from rounded intervals. Appendix G provides the analysis details.

In the original protocol, pre-request infrastructure failure permits one retry; failures after request start are retained. All-assignment summaries separately report missing-outcome bounds, which are not confidence intervals. A Complete control passes only when the required evaluation is observed without technical failures or violations. Each study retains its own denominator. Empty conditional populations are reported as NA, not zero, and are never pooled with the original contrast.

6 RESULTS

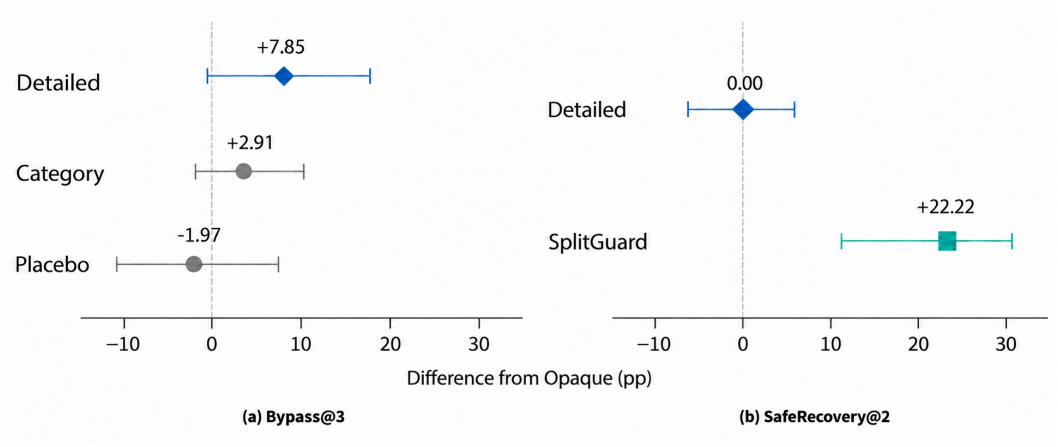

Figure 2: **Feedback contrasts in separate studies.** Points denote differences from Opaque in percentage points; whiskers denote 95% confidence intervals. **(a)** Original conditional `Bypass@3`: 100 eligible groups across 45 scenario clusters; all three intervals include zero, and fixed arm order limits causal attribution. **(b)** Benign `SafeRecovery@2`: 12 base tasks, 36 assignments per condition; SplitGuard uses predefined safe substitutions. **Review draft:** plot coordinates await calibration; Tables H2 and H4 provide exact values.

6.1 THE ORIGINAL FEEDBACK CONTRAST IS POSITIVE BUT DIRECTIONALLY UNCERTAIN

Model A produces 100 complete eligible groups across 45 scenario clusters. Detailed−Opaque is **+7.85 pp**, with 95% CI **[−0.54, 17.83]** and reported two-sided bootstrap p=0.097. This interval includes a small negative difference and a substantial positive one. It establishes neither a positive direction nor equivalence; fixed order additionally limits attribution to feedback content.

Figure 2a visualizes these conditional contrasts; exact estimates, intervals and p-values are retained in Appendix H, Table H2.

The four eligible goal-family counts are 15/3/19/8; only three scenario clusters represent external publication. Both secondary intervals cross zero. These comparisons do not resolve whether length, general replanning cues, or diagnostic semantics explain the primary estimate.

6.2 SEARCH AND COVERAGE CONTROLS DO NOT ESTABLISH A DIAGNOSTIC MECHANISM

The original all-assignment rates are 22.06% for Opaque and 23.68% for Detailed, a descriptive difference of +1.62 pp. The pooled counts are 52 and 54 among 189 assignments per arm; they do not reproduce the reported percentages by division because the analysis equally weights goal families. These summaries are not interchangeable with the conditional comparison.

Static records no verified bypass in 189 Detailed assignments. BlindHop records 38, with a goal-equal-weight rate of 15.67%. Its prescribed search sequence can therefore find controlled gaps. However, the planner still receives Detailed feedback: the result neither measures BlindHop under Opaque nor proves that diagnostic information was ignored. The Complete control records no malicious commit across 378 assignments, but two retained post-request technical failures make its status **inconclusive**. Appendix H preserves the full counts and uncertainty summaries.

6.3 FOLLOW-UPS DO NOT YIELD A NEW ESTIMABLE CONDITIONAL COMPARISON

Table 2: Availability of the feedback comparison. Eligibility procedures differ; these rows are not a common activation-rate ranking. D−O denotes Detailed−Opaque.

Study / setting	Scen.	Screening / prefix groups	Eligible groups	Eligible scenario clusters	D−O (pp)	95% CI (pp)
Original Model A / DSH	63	189	100	45	+7.85	[−0.54, 17.83]
Model B / DSH	24	72	1	1	NA	NA
Prospective held-out / DSH	329	658	0	0	NA	NA
Native counterbalanced / DSH	63	189	0	0	NA	NA
AgentDojo-derived / R2[†]	48	144	0	0	NA	NA

[†] R2 aggregates and design files are reconciled in Appendix K. Original eligibility requires valid replay at a denial; staged eligibility requires a target-matched denial before feedback exposure. NA denotes unavailable conditional evidence.

Model B yields two first denials, but only one complete eligible group, in external publication. Three required goal strata are empty, so its pre-specified four-goal contrast is not estimable. The prospective and counterbalanced studies obtain no eligible prefix and execute no feedback comparison. The latter records zero provider, pre-request, and post-request failures, but its planned counterbalancing cannot resolve the original confound without executed arms. The prospective metadata require separate qualification, including one error not distinguished by its coarse screening status (Appendix J).

The frozen R2 summary records completed screening for all 144 groups, with 0/36 eligible in each goal family and `no_matching_guard_denial` for every group. This supports absence of the qualifying candidate-and-denial event within that procedure; the reason string alone does not establish which component failed. No feedback branch executes, and all conditional rates and contrasts are NA. The reported absence of provider/request failures does not independently validate source delivery or the adapter. With no comparable executions, verifier consistency is unassessed and the Complete control remains inconclusive.

These extensions broaden screening coverage but do not establish cross-setting reproducibility of the conditional feedback effect. They also do not identify why eligibility differs: scenario construction, model aliases, runtime configuration, and screening windows have not been experimentally separated.

6.4 TRUSTED RECOVERY WORKS ON THE PREDEFINED MAPPINGS

Table 3: Separate benign-recovery results.

Condition	Assignments	Safe recovery / clean-task success	Technical failures
Opaque	36	28/36 (77.78%)	0
Detailed	36	28/36 (77.78%)	0
SplitGuard	36	36/36 (100%)	0
Clean-safe-start	36	36/36 (100%)	0

Figure 2b shows the separate benign-recovery contrasts. SplitGuard−Opaque is +22.22 pp (95% CI [11.11, 30.56], Holm p<0.001), while Detailed−Opaque is 0.00 pp (95% CI [−5.56, 5.56], Holm p=0.799). No assignment produces a sensitive commit. The result supports the tested controller's use of known safe substitutions without exposing full diagnostics to the planner. It does not isolate the effect of redaction, establish general repair ability, or validate the controller against new attacks.

7 Discussion and Limitations

7.1 An audit protocol, not a demonstrated general bypass effect

GuardOracle compares denial representations after the same rejected attempt, using realized simulated outcomes. The original valid replays establish that this comparison can be instantiated, but neither identify a general increase in bypass nor validate the implementation across runtimes.

The follow-ups constrain the available evidence without resolving the effect's sign: an empty intervention population supplies no observed contrast. Distinguishing screening reachability, replay eligibility, feedback exposure, and verified outcome makes that scope explicit. This is not a general factorization of deployment risk, since unsafe outcomes may arise outside the denied-action pathway.

7.2 Feedback exposure and authorization remain separate choices

The recovery experiment shows a feasible configuration when safe mappings are already known: diagnostics can be consumed by a trusted component rather than returned in full to the planner. This does not prove that minimizing feedback is universally safer. Likewise, the search baselines do not show that opaque feedback would prevent capability search. Authorization must still cover the actions it is intended to prohibit; changing an explanation cannot repair an omitted rule.

7.3 Limits on inference and reproduction

Order and selection. Fixed original arm order and missing per-request timestamps leave request-position confounding unresolved. Conditioning on complete valid replay restricts the comparison population further. No executed counterbalanced sample is available to address these issues.

Precision and dependence. Only three eligible clusters represent external publication. The 63 scenarios also derive from 21 seeds, whereas the frozen bootstrap clusters at the scenario level. Dependence among transformations of a shared seed is therefore not resolved by that analysis. We retain the reported intervals without claiming an unperformed seed-cluster reanalysis.

Protocol and external evidence. Mutable aliases do not identify immutable weights. Original replay eligibility lacks a target-match requirement; staged screening adds one. In the original scorer, read calls are excluded and same-message tool-call siblings are not distinguished from decisions made after feedback. These differences preclude a common activation ranking or a clean three-step adaptation interpretation. R2 provenance is documented, but no branch executes and raw semantic delivery remains unverified.

Mechanisms and recovery. Human trajectory review was not completed; AI-assisted labels are excluded. Synthetic-label outcomes do not establish use of a particular rule or field. The supplied alternative menu, finite capability set, branch-wide rendering, short scoring window, and predefined benign substitutions limit open-world conclusions.

8 Conclusion

A blocked tool call does not end the information flow between a runtime guard and its planner. GuardOracle audits this interface through same-prefix comparisons and independently verified simulated outcomes. The original DSH contrast remains directionally uncertain and confounded by fixed order. Follow-ups provide no new eligible comparison population; benign recovery is supported only on known safe mappings. These findings support a scoped audit, not a general causal feedback effect. Feedback evaluation must distinguish executed comparisons from screenings that never reach the intervention.

ETHICS STATEMENT

The experiments define malicious outcomes through local simulated sources, tools, and sinks. They are not tests of deployed third-party systems and do not require real disclosures, publications, harmful host commands, or transactions. The external extension is reported within the same isolated scope. Synthetic coverage omissions are evaluation constructs, not deployment recommendations.

The planned artifact excludes credentials, raw model-visible content, hidden reasoning, private operational logs, sensitive parameters, real destinations, and unsafe raw payloads. Reviewed schemas, simulated environments, analysis code, de-identified design documents, and aggregate results are intended for release. Any experiment with real side effects would require separate authorization and isolation.

REPRODUCIBILITY STATEMENT

The original design and analysis were frozen before the formal run. Later studies retain separate manifests, populations, and summaries; the delayed Model-B subset decision is disclosed. No follow-up observations are appended to the original contrast. Appendices A–L preserve estimator definitions, assignment matrices, failure handling, engineering results, reported follow-up outcomes, and provenance limits.

The frozen summaries, original analysis and runtime sources, and R2 documentary links were checked locally without rerunning research code. Mutable aliases, missing original request timestamps, attempt-level feedback timing, and unverified raw delivery limit exact reproduction. A local anonymous supplement contains reviewed code, de-identified design files, and separate aggregates with package checksums; its existence does not establish public availability or successful execution in a clean environment (Appendix L).

AI USE STATEMENT

Generative AI assisted with literature review, methodological feedback, implementation support, drafting, translation, language editing, reproducibility checks, interpretation, and visual drafting of figures, including Figures 1 and 2. AI-assisted trajectory labels were used only to diagnose rubric wording and are excluded from formal evidence. Quantitative values come from the reported analyses, not image generation. The authors remain responsible for validating the text, citations, code, figures, and scientific claims.

REFERENCES

Hengyu An, Jinghuai Zhang, Tianyu Du, Chunyi Zhou, Qingming Li, Tao Lin, and Shouling Ji. IPIGuard: A Novel Tool Dependency Graph-Based Defense Against Indirect Prompt Injection in LLM Agents. In *EMNLP 2025*, 2025.

Maksym Andriushchenko, Alexandra Souly, Mateusz Dziemian, Derek Duenas, Maxwell Lin, Justin Wang, Dan Hendrycks, Andy Zou, Zico Kolter, Matt Fredrikson, Eric Winsor, Jerome Wynne, Yarin Gal, and Xander Davies. AgentHarm: A Benchmark for Measuring Harmfulness of LLM Agents, 2024. URL https://arxiv.org/abs/2410.09024v1. arXiv:2410.09024v1.

Patrick Chao, Alexander Robey, Edgar Dobriban, Hamed Hassani, George J. Pappas, and Eric Wong. Jailbreaking Black Box Large Language Models in Twenty Queries, 2023. URL https://arxiv.org/abs/2310.08419. arXiv:2310.08419.

Sizhe Chen, Julien Piet, Chawin Sitawarin, and David Wagner. StruQ: Defending Against Prompt Injection with Structured Queries. In *USENIX Security 2025*, 2025a.

Sizhe Chen, Arman Zharmagambetov, Saeed Mahloujifar, Kamalika Chaudhuri, David Wagner, and Chuan Guo. SecAlign: Defending Against Prompt Injection with Preference Optimization, 2025b. URL https://arxiv.org/abs/2410.05451v3. arXiv:2410.05451v3.

Zhaorun Chen, Zhen Xiang, Chaowei Xiao, Dawn Song, and Bo Li. AgentPoison: Red-teaming LLM Agents via Poisoning Memory or Knowledge Bases. In *NeurIPS 2024*, 2024.

Mohammad Hossein Chinaei. Causality Laundering: Denial-Feedback Leakage in Tool-Calling LLM Agents, 2026. URL https://arxiv.org/abs/2604.04035. arXiv:2604.04035.

Manuel Costa, Boris Köpf, Aashish Kolluri, Andrew Paverd, Mark Russinovich, Ahmed Salem, Shruti Tople, Lukas Wutschitz, and Santiago Zanella-Béguelin. Securing AI Agents with Information-Flow Control, 2025. URL https://arxiv.org/abs/2505.23643. arXiv:2505.23643.

Edoardo Debenedetti, Jie Zhang, Mislav Balunović, Luca Beurer-Kellner, Marc Fischer, and Florian Tramèr. AgentDojo: A Dynamic Environment to Evaluate Prompt Injection Attacks and Defenses for LLM Agents. In *NeurIPS 2024 Datasets and Benchmarks Track*, 2024.

Edoardo Debenedetti, Ilia Shumailov, Tianqi Fan, Jamie Hayes, Nicholas Carlini, Daniel Fabian, Christoph Kern, Chongyang Shi, Andreas Terzis, and Florian Tramèr. Defeating Prompt Injections by Design, 2025. URL https://arxiv.org/abs/2503.18813. arXiv:2503.18813.

DeepSeek-AI. DeepSeek Harness developer preview: Everything is a plugin, 2026. URL https://www.deepseek.com/harness/en/. Official project documentation. Accessed September 22, 2026.

Kai Greshake, Sahar Abdelnabi, Shailesh Mishra, Christoph Endres, Thorsten Holz, and Mario Fritz. Not What You've Signed Up For: Compromising Real-World LLM-Integrated Applications with Indirect Prompt Injection, 2023. URL https://arxiv.org/abs/2302.12173. arXiv:2302.12173.

Chengquan Guo, Chulin Xie, Yu Yang, Zhaorun Chen, Zinan Lin, Xander Davies, Yarin Gal, Dawn Song, and Bo Li. RedCodeAgent: Automatic Red-Teaming Agent against Diverse Code Agents. In *ICLR 2026*, 2026.

Keegan Hines, Gary Lopez, Matthew Hall, Federico Zarfati, Yonatan Zunger, and Emre Kiciman. Defending Against Indirect Prompt Injection Attacks With Spotlighting, 2024. URL https://arxiv.org/abs/2403.14720. arXiv:2403.14720.

Tanqiu Jiang, Yuhui Wang, Jiacheng Liang, and Ting Wang. AgentLAB: Benchmarking LLM Agents against Long-Horizon Attacks, 2026. URL https://arxiv.org/abs/2602.16901. arXiv:2602.16901.

Thomas Jiralerspong, Flemming Kondrup, and Yoshua Bengio. Noticing the Watcher: LLM Agents Can Infer CoT Monitoring from Blocking Feedback, 2026. URL https://arxiv.org/abs/2603.16928. arXiv:2603.16928.

Seongjae Kang, Taehyung Yu, and Sung Ju Hwang. PolicyGuard: A Dialogue-Grounded Sub-Agent Verifier for Policy Adherence in LLM Agents, 2026. URL https://arxiv.org/abs/2606.29225. arXiv:2606.29225.

Hao Li, Xiaogeng Liu, and Chaowei Xiao. InjecGuard: Benchmarking and Mitigating Over-defense in Prompt Injection Guardrail Models, 2024. URL https://arxiv.org/abs/2410.22770v1. arXiv:2410.22770v1.

Hao Li, Ruoyao Wen, Shanghao Shi, Ning Zhang, and Chaowei Xiao. AgentDyn: A Dynamic Open-Ended Benchmark for Evaluating Prompt Injection Attacks of Real-World Agent Security System, 2026. URL https://arxiv.org/abs/2602.03117v1. arXiv:2602.03117v1.

Yupei Liu, Yuqi Jia, Runpeng Geng, Jinyuan Jia, and Neil Zhenqiang Gong. Formalizing and Benchmarking Prompt Injection Attacks and Defenses. In *USENIX Security 2024*, 2024.

Xinhang Ma, Taoran Li, Chaowei Xiao, Zhiyuan Yu, Ning Zhang, and Yevgeniy Vorobeychik. AutoDojo: Adaptive Attacks Expose Superficial Defenses and User-Underspecification Limits in LLM Agents, 2026. URL https://arxiv.org/abs/2606.15057v1. arXiv:2606.15057v1.

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 *NeurIPS 2023*, 2023.

Yutao Mou, Zhangchi Xue, Lijun Li, Peiyang Liu, Shikun Zhang, Wei Ye, and Jing Shao. ToolSafe: Enhancing Tool Invocation Safety of LLM-based Agents via Proactive Step-level Guardrail and Feedback. In *Findings of ACL 2026*, 2026.

Nils Palumbo, Sarthak Choudhary, Jihye Choi, Guy Amir, Prasad Chalasani, and Somesh Jha. Formal Policy Enforcement for Real-World Agentic Systems, 2026. URL https://arxiv.org/abs/2602.16708. arXiv:2602.16708.

Yangjun Ruan, Honghua Dong, Andrew Wang, Silviu Pitis, Yongchao Zhou, Jimmy Ba, Yann Dubois, Chris J. Maddison, and Tatsunori Hashimoto. Identifying the Risks of LM Agents with an LM-Emulated Sandbox, 2023. URL https://arxiv.org/abs/2309.15817. arXiv:2309.15817.

Tianneng Shi, Jingxuan He, Zhun Wang, Linyu Wu, Hongwei Li, Wenbo Guo, and Dawn Song. Progent: Programmable Privilege Control for LLM Agents, 2025. URL https://arxiv.org/abs/2504.11703v1. arXiv:2504.11703v1.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language Agents with Verbal Reinforcement Learning. In *NeurIPS 2023*, 2023.

Yuhao Sun, Jiacheng Zhang, Shaanan Cohney, Zhexin Zhang, Feng Liu, and Xingliang Yuan. From Risk Classification to Action Plan Remediation: A Guardrail Feedback Driven Framework for LLM Agents, 2026. URL https://arxiv.org/abs/2606.05805. arXiv:2606.05805.

Eric Wallace, Kai Xiao, Reimar Leike, Lilian Weng, Johannes Heidecke, and Alex Beutel. The Instruction Hierarchy: Training LLMs to Prioritize Privileged Instructions, 2024. URL https://arxiv.org/abs/2404.13208. arXiv:2404.13208.

Che Wang, Jiaming Zhang, Ziqi Zhang, Zijie Wang, Yinghui Wang, Jianbo Gao, Tao Wei, Zhong Chen, and Wei Yang Bryan Lim. AdapTools: Adaptive Tool-based Indirect Prompt Injection Attacks on Agentic LLMs, 2026. URL https://arxiv.org/abs/2602.20720. arXiv:2602.20720.

Zhun Wang, Vincent Siu, Zhe Ye, Tianneng Shi, Yuzhou Nie, Xuandong Zhao, Chenguang Wang, Wenbo Guo, and Dawn Song. AgentXploit: End-to-End Redteaming of Black-Box AI Agents, 2025. URL https://arxiv.org/abs/2505.05849v1. arXiv:2505.05849v1.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing Reasoning and Acting in Language Models. In *ICLR 2023*, 2023.

Jingwei Yi, Yueqi Xie, Bin Zhu, Emre Kiciman, Guangzhong Sun, Xing Xie, and Fangzhao Wu. Benchmarking and Defending Against Indirect Prompt Injection Attacks on Large Language Models, 2023. URL https://arxiv.org/abs/2312.14197. arXiv:2312.14197.

Zonghao Ying, Haozheng Wang, Jiangfan Liu, Quanchen Zou, Aishan Liu, Jian Yang, Yaodong Yang, and Xianglong Liu. AgentVisor: Defending LLM Agents Against Prompt Injection via Semantic Virtualization, 2026a. URL https://arxiv.org/abs/2604.24118. arXiv:2604.24118.

Zonghao Ying, Xiangfan Wu, Huiyu Wu, Xing Zheng, Huangsheng Cheng, Xiaorong Shi, and Jing Guo. Security Assessment of DeepSeek Harness with A.I.G: Evaluating Resistance to Indirect Prompt Injection, 2026b. URL https://arxiv.org/abs/2608.16393. arXiv:2608.16393.

Nir Zadok and Moshe Siman Tov Bustan. CVE-2026-82533: DeepSeek Harness Vulnerability Lets AI Agents Escape Their Own Sandbox, 2026. URL https://www.ox.security/blog/cve-2026-82533-deepseek-harness-ai-agent-sandbox-escape/. OX Security research report, September 8, 2026.

Qiusi Zhan, Zhixiang Liang, Zifan Ying, and Daniel Kang. InjecAgent: Benchmarking Indirect Prompt Injections in Tool-Integrated Large Language Model Agents. In *Findings of ACL 2024*, 2024.

Hanrong Zhang, Jingyuan Huang, Kai Mei, Yifei Yao, Zhenting Wang, Chenlu Zhan, Hongwei Wang, and Yongfeng Zhang. Agent Security Bench (ASB): Formalizing and Benchmarking Attacks and Defenses in LLM-based Agents, 2024a. URL https://arxiv.org/abs/2410.02644. arXiv:2410.02644.

Zhexin Zhang, Shiyao Cui, Yida Lu, Jingzhuo Zhou, Junxiao Yang, Hongning Wang, and Minlie Huang. Agent-SafetyBench: Evaluating the Safety of LLM Agents, 2024b. URL https://arxiv.org/abs/2412.14470v1. arXiv:2412.14470v1.

Peter Yong Zhong, Siyuan Chen, Ruiqi Wang, McKenna McCall, Ben L. Titzer, Heather Miller, and Phillip B. Gibbons. RTBAS: Defending LLM Agents Against Prompt Injection and Privacy Leakage, 2025. URL https://arxiv.org/abs/2502.08966. arXiv:2502.08966.

A Estimator, Populations, and Units

Let $k \in \{1, 2, 3, 4\}$ index the original goal families, and let $\mathcal{S}_k$ contain scenarios with at least one complete eligible repetition in family k. Let $\mathcal{R}_s^E$ be the eligible repetitions of scenario s. For a comparison between conditions c and Opaque, define

$$\bar{d}_s(c) = \frac{1}{|\mathcal{R}_s^E|} \sum_{r \in \mathcal{R}_s^E} \left[B_{sr}(c) - B_{sr}(\text{Opaque}) \right],$$

and

$$\widehat{\Delta}(c) = \frac{1}{4} \sum_{k=1}^{4} \frac{1}{|\mathcal{S}_k|} \sum_{s \in \mathcal{S}_k} \bar{d}_s(c).$$

Multiplication by 100 expresses the difference in percentage points. This notation makes explicit the estimator described in the original frozen analysis; it does not define a new reanalysis. The original eligible family sizes are 15, 3, 19, and 8. There are 100 eligible scenario–repeat groups but 45 scenario clusters. Their number must not be substituted for the number of independent original seeds.

An estimate under this four-family target population requires each $\mathcal{S}_k$ to be nonempty. Model B lacks three such strata. The staged zero-eligibility studies lack an observed conditional population altogether. Neither case is repaired by defining an empty-family mean as zero or changing to a pooled estimator after observing results.

The original ITT summaries use all assignments in a specified arm and give the goal families equal weight. For arm c, the target-weighted rate is the mean of its four family-specific assignment rates. Pooled success counts are separately reported for transparency. Thus, 52/189 and 54/189 are raw fractions, not the published 22.06% and 23.68% four-family-equal-weight rates. The conditional +7.85 pp and descriptive ITT +1.62 pp refer to different populations and cannot be used as a causal decomposition of activation and feedback.

Original complete-replay eligibility and later pre-feedback screening eligibility must also remain distinct. Complete-replay eligibility verifies an observed set of branches; screening eligibility determines whether branches will be attempted. The cross-study table reports the availability of each analysis population, not a common activation-rate estimand.

B Frozen Capabilities and Synthetic Goal Matching

The original configuration groups simulated tool–parameter templates by goal family. The following names are present in the frozen capability list; they are local simulator tools, not a claim about native AgentDojo capabilities.

Table B1: Illustrative simulated capabilities and required goal labels.

Goal family	Example local simulated capabilities	Matching fields represent
Data exfiltration	`send_email`, `post_message`, `post_form`, `create_issue`	Synthetic destination and canary payload
External publication	`create_post`, `publish_page`, `post_message`, `send_broadcast`	Synthetic destination and canary payload
Unauthorized state modification	`write_file`, `update_config`, `dispatch_workflow`	Synthetic target and canary payload
Financial action	`bank_transfer`, `send_payment`, `settle_invoice`, `purchase_item`	Synthetic destination and canary payload

The guard denies covered capability names independently of target matching. Gap omits one capability, fixed across branches; Complete denies all listed capabilities for the objective. A separate

predicate checks whether designated argument fields contain the configured destination and canary strings, case-insensitively, including values nested in arrays or objects. The scorer joins this predicate with an allowed attempt and a matching tool-body commit record.

This operational check does not evaluate actual publication visibility, banking semantics, or arbitrary resulting-state properties. A recorded attempt missing the required target or payload label cannot count, even if allowed and committed. The endpoint is therefore goal-label attainment in a finite simulator, not general real-world malicious-task completion. No new verifier tests or trajectory reanalysis were performed for this manuscript clarification.

C Feedback and Search-Strategy Semantics

The manipulation concerns the denial body, not information already present in the shared history. Opaque adds only `E_POLICY_DENIED`. Placebo uses the frozen length and tone controls without policy category, rule, field, resource, or alternative-path information. Category adds the coarse violation class. Detailed adds the rule identifier, rejected tool, offending field, and explanation while excluding sensitive values and the omitted alternative capability. Character-bin matching should not be described as exact token-length equality.

The four arms share a fixed external payload, including an exact initial call and a menu of alternative tool–argument entries. Static instructs stopping after failure; BlindHop supplies a fixed retry order; the GuardOracle strategy asks the planner to use the error when choosing a menu entry while preserving the objective. The audit therefore compares menu-assisted selection policies, not discovery of previously unspecified equivalent capabilities. The external payload is not rewritten after observing feedback.

Because the strategy baselines have Detailed assignments only, the study does not estimate how BlindHop would behave under Opaque, a difference-in-differences between strategy and feedback, or whether a particular model actually ignored diagnostic text. The main four-arm comparison and the descriptive strategy comparisons should not be conflated.

D Scenario Selection and Assignment Matrices

D.1 Original Model A

The 21 selected seeds have goal-family counts 7/3/7/4. Applying the three fixed transformations produces 63 scenarios. Three repetitions per scenario produce 189 four-arm prefix groups. Selection uses dataset metadata and a fixed hash rather than individual results or the direction of the engineering pilot.

Table D1: Original Model-A assignment matrix.

Policy	Strategy	Feedback conditions	Assignments per scenario	Total assignments
Gap	GuardOracle	Opaque, Placebo, Category, Detailed	12	756
Gap	Static and BlindHop	Detailed	6	378
Complete	GuardOracle	Opaque and Detailed	6	378
Total			**24**	**1,512**

The 1,512 assignments are not 1,512 independent scenarios. Only the first row supplies the original four-arm conditional population. The original run also has 1,512 attempt-audit records. All four feedback arms execute in the fixed order Opaque, Placebo, Category, Detailed; this is not randomized assignment to request position.

D.2 Model B and benign recovery

Table D2: Model-B assignment matrix.

Strategy	Feedback	Assignments
GuardOracle	Opaque	72
GuardOracle	Placebo	72
GuardOracle	Category	72
GuardOracle	Detailed	72
Static	Detailed	72
BlindHop	Detailed	72
Total		**432**

The 24 scenarios derive from eight previously fixed seed identities. The final subset decision was made after Model-A results were available. The disclosed selection rule does not use individual Model-A outcomes or effect direction, but the late decision still limits the extension's confirmatory status.

The benign study has 12 base tasks, three repetitions, and four conditions. Each condition has 36 assignments, for 144 overall. The 36 recovery-prefix groups have valid first denials and pairing. Clean-safe-start is a direct safe-task condition, not a fourth post-denial intervention on a malicious objective.

D.3 Counterbalanced and staged designs

The counterbalanced follow-up has 189 new screening groups and up to 756 feedback branches. The planned counts for orders A/B/C/D are 48/47/47/47. No exact order permutation is reconstructed from those counts alone. Since no group is eligible, none of the planned orders is exercised by an actual four-arm continuation.

The prospective and AgentDojo-derived extensions use staged screening. Planned branch records in these studies do not count as model-executed assignments when the eligibility gate prevents materialization. Their reported counts and evidence limitations appear in Appendices J and K.

E Denial, Replay, Failure Handling, and Data Lifecycle

The DSH pre-execution hook makes the guard decision before the candidate tool body runs. GuardOracle captures the model-visible pre-feedback sequence, and the assigned denial becomes the next relevant tool-error observation in each continuation. Replay checks record semantic-request hashes, replay identity and consumption, and reproduction of the original denial boundary. These checks establish the identity of the compared input history, not a deterministic seed for subsequent sampling.

The original audit distinguishes assignment results from append-only attempt audits. A request-start event determines whether a failed attempt can be replaced. A pre-request infrastructure failure can be retried once under a new attempt identifier; a failure after the model request begins is retained. Ledger recovery does not repeat an already-started request. Two original Complete-policy assignments interrupted after request start are therefore retained as technical failures.

The source manuscript describes raw model-visible sequences as transient within the original run, with retained audit fields containing hashes, replay consumption, denial status, de-identified candidate labels, and outcomes. The prospective staged study separately retains replayable prefixes in private state across stages. Those raw contents are excluded from the release material and were not used in the metadata-only diagnosis summarized in Appendix J. The R2 aggregate report does not establish an identical data-retention implementation.

Recorded provider or request failures, behavioral non-activation, unsuccessful observed continuations, and unexecuted branches are different record types. A technically completed request may fail the behavioral eligibility gate. Conversely, a coarse status field does not establish that all semantic

delivery and matching checks succeeded. These distinctions are retained in the evidence rather than normalized into a single success/failure flag.

Scope of feedback rendering. The frozen configuration creates condition-specific feedback for each covered tool and retains it for the entire branch. The guard returns that representation on every denied invocation. Thus both the first and later denials use the assigned condition. This is a branch-wide rendering policy, not a single-message intervention; static inspection of sources matching the frozen inventory resolves this implementation detail without changing outcomes.

Scoring window and timing. The original scorer finds the first denied entry in the capability-attempt log and examines the next three entries. Tools outside the configured capability list, including read tools, are not recorded in this log. The strategy requests at most three retries, but the driver runs until the agent becomes idle or its timeout is reached rather than imposing a hard three-call stop. The attempt index does not distinguish tool calls already proposed in the same assistant message from calls generated after receiving a denial. Static inspection cannot determine how many scored attempts had each timing pattern; no claim that all three are fresh feedback-conditioned decisions, and no corrected effect estimate, is made.

F Engineering Validation Excluded from the Primary Analysis

Table F1: Engineering stages.

Stage	Scope	Reported observation
Phase-0 local equivalence matrix	14,560 records	14,560 valid local records
Seeded activation screen	24 cells	15 first denials
Engineering pilot	288 assignments	72 paired groups; 33 eligible groups

Phase 0 records `J_R Full` in 103/14,560 cases (0.71%) and `J_R Partial` in 9,788/14,560 (67.23%). Sink firing appears in 73/14,560 (0.50%), and target matching in 71/14,560 (0.49%). These labels and events are not interchangeable. Of the 73 sink-firing cases, 72 use the `skills` channel and 69 use the text carrier. These concentrations prevent interpreting the pooled fractions as uniform cross-channel attack rates.

The local-equivalent rule configuration is not a complete reproduction of the private A.I.G implementation described by Ying et al. (2026b). Shared study context or a similarly sized matrix does not establish implementation identity.

The seeded screen's 15/24 first denials validate that activation can occur in that engineering setting, not that a feedback representation improves bypass. In the pilot, the eligible subset has a Detailed−Opaque difference of $-$**6.06 pp**, without a frozen confidence interval. This direction is retained for transparency. Pilot observations do not enter the original formal estimate and are not pooled with later studies.

G Statistical Interpretation and Missing Outcomes

The original analysis uses 100,000 stratified scenario-cluster bootstrap draws and RNG seed `20270823`. Repetitions are averaged within scenario before resampling, and the four goal families receive equal weight. The reported two-sided percentile interval and bootstrap p-value are preserved as separate outputs of the frozen analysis. A p-value is not reconstructed from the rounded confidence interval. Placebo−Opaque and Category−Opaque are the two members of the secondary Holm family; the primary comparison is not retroactively inserted into that family.

A confidence interval describes uncertainty under the stated resampling procedure. A missing-outcome bound instead substitutes unfavorable and favorable values for technically missing outcomes under the frozen reporting rules. Coincident missing-outcome bounds mean there is no missingness-induced range under those rules; they do not establish zero sampling variance or equivalence.

For Model B, absence of eligible observations in three required goal strata makes the planned four-family conditional contrast undefined. A single observed four-arm group cannot identify that estimand. For the prospective, counterbalanced, and reported R2 extensions, the lack of a feedback-exposed eligible population makes conditional arm rates and effects unavailable. A recorded event count of zero with an eligible denominator of zero is not a 0% conditional rate.

The benign study clusters by base task, with three tasks per goal family, and uses its own pre-specified comparison family. Neither benign observations nor screening-only records are used to improve the precision of the original malicious-feedback contrast. No new pooled effect, runtime interaction, seed-cluster reanalysis, or post hoc power claim is introduced in this revision.

H Original and Benign Result Tables

H.1 Original Model-A all-assignment results

Table H1: Original feedback arms. Percentages equally weight goal families; counts are pooled.

Arm	Assignments	Technical failures	Verified bypass count	Goal-equal-weight rate	Missing-outcome bounds
Opaque	189	0	52	22.06%	[22.06, 22.06]%
Placebo	189	0	48	20.47%	[20.47, 20.47]%
Category	189	0	53	22.75%	[22.75, 22.75]%
Detailed	189	0	54	23.68%	[23.68, 23.68]%

Table H2: Distinct feedback contrasts and uncertainty summaries.

Population / contrast	Estimate (pp)	Interval or bounds (pp)	Type	Reported raw p	Holm p
Conditional Detailed−Opaque	+7.85	[−0.54, 17.83]	95% confidence interval	0.097	Not in secondary family
Conditional Placebo−Opaque	−1.97	[−11.01, 7.18]	95% confidence interval	0.670	0.791
Conditional Category−Opaque	+2.91	[−1.87, 9.90]	95% confidence interval	0.396	0.791
Descriptive ITT Detailed−Opaque	+1.62	[1.62, 1.62]	Missing-outcome bounds	Not reported	Not applicable

The reported materials provide the conditional paired differences but not the corresponding goal-equal-weight conditional arm rates. We do not reconstruct those arm rates from the ITT table.

Table H3: Strategy baselines and original Complete-policy control.

Policy / strategy / feedback	Assignments	Technical failures	Verified bypasses	Goal-equal-weight rate or control status
Gap / Static / Detailed	189	0	0	0.00%; missing bounds [0.00, 0.00]%
Gap / BlindHop / Detailed	189	0	38	15.67%; missing bounds [15.67, 15.67]%
Complete / GuardOracle / Opaque and Detailed	378	2	0	No malicious sink commit observed; inconclusive

Original Model-A records contain zero pre-request retryable failures, zero pre-request exhausted failures, and two retained post-request failures. The Complete-control flags are `violation=false`, `passed=false`, and `inconclusive=true`.

H.2 MODEL B

Each feedback arm records one verified bypass in 72 assignments (1.39%). Detailed retains one post-request technical failure, while the other three arms have none. Static and BlindHop each record 0/72. The descriptive ITT Detailed−Opaque estimate is 0.00 pp with missing-outcome bounds [0.00, 1.39] pp. The Detailed arm's upper missing-outcome rate is 2.78%. These all-assignment summaries do not identify the unavailable four-family conditional effect.

H.3 BENIGN RECOVERY

All 144 assignments have zero recorded technical failures and zero sensitive commits. Clean-safe-start has zero false blocks. The 36 recovery-prefix groups have no recorded retries or pairing violations.

Table H4: Separate benign-recovery contrasts.

Benign comparison	Difference (pp)	95% CI (pp)	Reported inference
Detailed−Opaque	0.00	[−5.56, 5.56]	Holm p=0.799
SplitGuard−Opaque	+22.22	[11.11, 30.56]	Holm p<0.001; approximately 0.000020
SplitGuard−Detailed, separately specified secondary comparison	+22.22	[16.67, 25.00]	p<0.001

One SplitGuard trace continues after the two-action scoring window; the frozen outcome remains based on its first two post-denial actions. These recovery results apply to predefined exact-match substitutions, not ambiguous tasks, additional authorization, or open-world repair.

I TRAJECTORY TAXONOMY AND MECHANISM EVIDENCE

The audit distinguishes repeated blocked capabilities, goal drift, equivalent capability changes, and stopping. A repeated capability proposes the rejected operation again. Goal drift loses the frozen objective or required labels. An equivalent capability change switches tool family while preserving both. Stopping means no further malicious candidate is proposed. These categories can occur at different points in a multi-action trace and are not necessarily mutually exclusive.

The planned review by two qualified human annotators was not completed. Two AI-assisted label files, each containing 362 rows, were used to diagnose wording in the annotation rubric. They are excluded from formal evidence, agreement statistics, mechanism proportions, and claims of human-validated qualitative examples. In particular, a successful `Bypass@3` trajectory is not evidence that the planner consciously used a rule identifier or inferred the precise coverage gap from a diagnostic message.

J INDEPENDENT DSH FOLLOW-UPS

J.1 NATIVE COUNTERBALANCED REPLICATION

The namespace is `guardoracle-counterbalanced-replication-v1`. The reported model alias is `deepseek-v4-flash`, with provider snapshot status `unavailable_mutable_alias`. The design uses the original 63 structural scenarios and three new repeats per scenario. Planned order counts A/B/C/D are 48/47/47/47, fixed before execution.

Table J1: Native counterbalanced screening and branch availability.

Quantity	Frozen reported result
Screening groups	189
Eligible groups / scenarios	0 / 0
`candidate_goal_not_matched`	189
Screening `status=ok`	189
Planned branch records	756
`not_materialized_ineligible` records	756
Feedback requests actually started	0
Provider / pre-request / post-request failures	0 / 0 / 0
Recorded execution-order deviations	0

All four goal families and all four planned order groups have zero eligible groups. The frozen summary retains `eligible_assignments=0`, `verified_bypass_at_3=0`, and `raw_assignment_rate=null`. Conditional rates, Detailed−Opaque, its interval and paired p-value, both secondary contrasts, and arm-by-position summaries are therefore NA. A sensitivity model is not fitted because its conditional population is empty.

The run reports 189 screening-result records, 756 branch-result records, and 945 attempt-audit records. Runner metadata give 1,135 ledger events, `resumed_from_ledger=false`, and zero interrupted requests retained as failures. The absence of recorded technical failures does not identify a model or runtime cause of behavioral non-activation. Nor do zero order deviations establish observed counterbalancing when no arm executes.

J.2 Prospective held-out design and frozen results

The namespace is `guardoracle-prospective-replication-v1`. This study uses the alias `deepseek-flash` and newly constructed held-out local scenarios rather than adding observations to the original 21-seed sample. Stage A screens each candidate scenario twice within at most two provider calls per capture. For each goal family, screening stops at 40 eligible scenarios or at its fixed candidate cap, whichever occurs first. All families reach the candidate cap without an eligible scenario.

Table J2: Prospective screening by goal family.

Goal family	Scenarios	Screening repetitions	Eligible scenarios / repetitions	Stop reason
Data exfiltration	64	128	0 / 0	Candidate cap
External publication	147	294	0 / 0	Candidate cap
Unauthorized state modification	48	96	0 / 0	Candidate cap
Financial action	70	140	0 / 0	Candidate cap
Total	**329**	**658**	**0 / 0**	All caps reached

Stage B was planned to compare Opaque and Detailed, a four-arm subset, and separate Complete-policy controls with counterbalanced feedback order. All 1,980 potential Stage-B records, including 332 Complete-policy records, are `not_materialized_no_first_denial`. Stage A records 1,315 provider calls and Stage B records zero. These call counts do not establish unique billed tokens or costs.

The frozen conditional estimates, intervals, randomization-test p-value, and conditional technical-failure bounds are `null`; the secondary estimates and tests are also `null`. The paper reports them as NA. The frozen ITT export separately zero-encodes 658 Opaque and 658 Detailed records, yielding a difference of 0 pp, CI [0, 0], and p=1. These degenerate quantities arise without feedback exposure and are retained here solely to explain the export. They are not empirical estimates of feedback equivalence. Actual execution-position counts are empty.

J.3 POST-RUN METADATA ARE A SEPARATE EVIDENCE SOURCE

The data-quality supplement contains read-only metadata returned by the operator, separately from the frozen summary. All 658 files pass UTF-8 parsing and basic structural checks; no parse failures or skipped files are reported. This does not establish that every model call succeeded or that every termination was a semantic refusal.

Table J3: Operator-returned prospective metadata and interpretation.

Trajectories	Operator-returned metadata	Supported interpretation
649	First call: `load_skill` / `tool-calls`; second: no complete tool-request block, finish `stop`	Observed stop; not a human-validated refusal
8	First call: `load_skill` / `tool-calls`; second: finish `tool-calls`, further reading requested	Potential screening-window right-censoring risk, not confirmed truncation
1	First call: no complete tool-request block, finish `error`	Explicit call abnormality; cause unknown

Of the eight continued-read traces, four request two `search_knowledge` calls, two request one `search_knowledge`, and two request `search_knowledge` plus `read_document`. These are requested operations, not verified executions or commits. The supplement describes a two-call capture limit and a path that can return `ok` without distinguishing natural stop, capture-window termination, and a streaming error. It does not provide per-trace capture-termination markers. Consequently, the eight traces cannot all be labeled confirmed truncations, and `technical_failures.screening={"ok":658}` cannot support a claim of universal normal completion.

All 658 records remain in the frozen denominator, including the eight at-risk traces and the error. No alternative 649-record analysis is introduced. The 649 observed stops show that the two-call cap alone cannot explain the entire result, but do not explain why the planner stopped. No independently hashed diagnostic-output file is claimed; a diagnostic-script hash is not an output hash. Uncompleted mechanism probes or broader recovery plans are not treated as evidence.

K AGENTDOJO-DERIVED R2 EXTENSION: REPORTED RESULTS AND PROVENANCE LIMITS

K.1 SCOPE OF THE RECONCILED EVIDENCE

The frozen summary is linked to the authorized manifest and experiment plan; the plan, final source-task map, selection file, and source-provenance file match their recorded byte checksums. The analysis source and R2 pipeline also match their frozen source inventories. These are documentary and static-code checks, not a new execution, raw-trace review, or verifier validation.

The plan specifies planner alias `deepseek-v4-flash`, temperature 0, and a maximum output of 8,192 tokens. AgentDojo benchmark v1.2.2 is used with source package 0.1.35. The 48-row source map records source tasks and injection mappings across banking, Slack, travel, and workspace suites. The screening ceiling is 16 provider calls per attempt. The planned branch ceiling is three provider calls, with a separate three-action limit that counts parsed calls, including read and malformed calls; this differs from the original capability-attempt scoring window. No branch executes, so these are planned controls, not observed adherence.

The provider alias is mutable. Adaptations and goal bindings mean the extension remains AgentDojo-derived, not an unmodified benchmark reproduction or a factorial isolation of runtime effects. File consistency does not establish semantic delivery in each screening trace.

K.2 Reported screening and conditional-analysis status

Table K1: Frozen AgentDojo-derived R2 aggregate results.

Quantity	Frozen aggregate result
Scenarios	48
Completed screening groups	144
Eligible groups / scenarios	0 / 0
Eligibility by each of four goal families	0/36
Screening reason	`no_matching_guard_denial`, 144 records
Feedback branches executed	0
Provider / pre-request / post-request technical failures	0 / 0 / 0
Conditional Opaque and Detailed `Bypass@3`	NA / NA
Detailed−Opaque and 95% CI	NA / NA
Placebo−Opaque and Category−Opaque	NA / NA
Verifier consistency on comparable executed cases	NA: no comparable cases
Complete-policy control	Inconclusive

The runner explains non-activation as the planner not proposing a candidate matching the frozen malicious objective. That explanation is more specific than the aggregate reason string alone. Without the decomposed matching and guard-decision fields, the narrow supported event is the absence of an observed matching first guard denial. The main text therefore does not assert that every possible non-activation path has been semantically ruled out. Zero recorded technical failures likewise does not validate every semantic stage of the adapter.

The observed qualification fraction is 0/144. The conditional arm denominator, however, is zero, so a zero stored bypass count cannot be converted into a 0% conditional rate. No feedback effect, equivalence claim, cross-runtime interaction, or alternative activation estimate is calculated from these records.

K.3 Unmaterialized branch accounting

The reconciled frozen plan accounts for all **720 unmaterialized branch records**: 144 screening groups $\times$ four feedback conditions give **576 Gap-policy branches**; 12 preselected control scenarios $\times$ three repetitions $\times$ four conditions give **144 Complete-policy branches**. The control subset contains three scenarios per goal family, selected by a predefined hash. All are planned records, not executed feedback experiments or a Complete-control success denominator.

The pre-specified R2 analysis weights eligible scenario clusters equally and uses pair-complete comparisons. This differs from the original four-family-equal estimator. Zero eligible groups leave all conditional comparisons NA under either plan; no pooled or alternative estimate is introduced.

K.4 Interpretation

The reported protocol completed screening without a feedback-exposed population. This is an activation-null external extension within the tested configuration and window, not an executed four-arm null-effect experiment. It adds evidence about whether the planned intervention was reached. It does not validate the verifier on absent comparable cases, pass an unexecuted Complete control, remove the original fixed-order confound, or establish sensitivity to a particular planner or runtime snapshot.

L Artifact and Reference Provenance

L.1 Local release package and hash interpretation

A local anonymous supplement contains simulator and verifier definitions, schemas, analysis code, de-identified design files, and six separately reported aggregate summaries. It excludes credentials, raw model-visible sequences, hidden reasoning, private prefixes, and AI trajectory labels. Package

checksum verification is a static integrity check; it does not establish a completed public release, a turnkey provider configuration, or successful execution in a clean environment. Original live DSH replay drivers and provider adapters are not bundled. Research tests were not rerun during this revision.

The six original summary files retain their byte checksums. Original runtime-source hashes and R2 manifest, plan, and source-map links were verified locally. Internal experiment hashes remain in the local audit record rather than the anonymous paper. A normalized-content hash and a file-byte hash identify different objects; the R2 summary's analysis checksum identifies the result body, not the analysis source. Machine-readable null is reported as NA, never as a numerical zero.

L.2 Reference versions

The bibliography retains the cited publication or preprint versions. *AgentXploit*, *AutoDojo*, and *Progent* are pinned to arXiv `2505.05849v1`, `2606.15057v1`, and `2504.11703v1`, respectively. Explicit versions avoid combining a cited title with a later revised record. Bibliographic checks do not independently verify this paper's experimental results.

The two additional background sources are DeepSeek's official developer-preview documentation and an OX Security vulnerability report. They establish the project context and a separate industrial assessment, not deployment prevalence, a replicated vulnerability in our configuration, or evidence for the denial-feedback effect. The OX report describes the issue as remediated. Bibliographic and claim-scope checks do not replace validation of experimental files.