When Is a Policy Violation a Model Failure? Policy Observability and Failure Attribution in Agent-Safety Evaluation

Anonymous authors
Paper under double-blind review

Abstract

Reading a policy violation as a failure of rule following assumes that the agent had the rules and facts it needed. We study policy observability: whether the information available at decision time determines the authorization label used for evaluation. We formalize two conditions (sufficiency, that the scored label is a function of the agent-visible input, and presentation invariance, that this input does not change under transformations the benchmark's own data format declares meaningless) and give an audit protocol with checkable certificates. An audit of 7,974 released AuthorityBench records resolves 65 call-level replay discrepancies through a documented reconstruction of historical scenario definitions. We also construct 34 policy pairs across 21 scenarios with identical agent-visible inputs under the released input specification but opposite authorization labels, so the available information does not determine the required decision. Applied to seven further released agent benchmarks, the protocol finds sufficiency counterexamples in two, presentation that varies with process state in the released pipelines of three, and one benchmark that scores infrastructure errors as successful attacks. In one of these pipelines, changing only the interpreter's hash seed changes a deterministic agent's scored utility on 6 of 35 affected cases (all in one user task). Experiments on an LLM authorization judge complement the audit. In 330 evaluation attempts on 55 action proposals from three retail tasks, candidate-first ordering had zero estimated excess disagreement relative to generic canonicalization in 50 complete-valid comparisons. The disagreement that did occur, on three proposals where the policy text and the tool description support different decisions, replicates in a pre-registered follow-up: reordering the top-level members of content-identical evidence flips the evaluator's decision in every fresh session, 52 random member orders of one proposal split 30 to 22, and randomizing only nested members never changes it (0 of 12, against 7 of 12 for top-level orders). When the conflict is removed from the text (one of two pre-registered variants), the split disappears (51 of 51 orders approve). This evaluator fails our competence control (14 of 20, partly on flawed items); the one evaluator that passes it, Claude Opus 5.5, approves on all 52 random orders. These experiments measure decision consistency, not authorization accuracy, on proposals selected because they flipped. Together, the findings identify information, versioning and measurement conditions that must be checked before attributing benchmark violations to model failure.

1 Introduction

When a benchmark reports that an agent "violated a policy", readers take it to mean that the agent was given the rule, saw the facts, and acted against them. This reading holds only under conditions that published evaluations rarely check. First, the condition the evaluator scores must be determined by what the agent received. Second, what the agent received must not depend on choices that the benchmark itself treats as meaningless. Third, the trials in the denominator must be trials in which the agent actually acted. If any of these fails, part of the reported violation rate can measure how the benchmark was built, not how the model behaves.

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

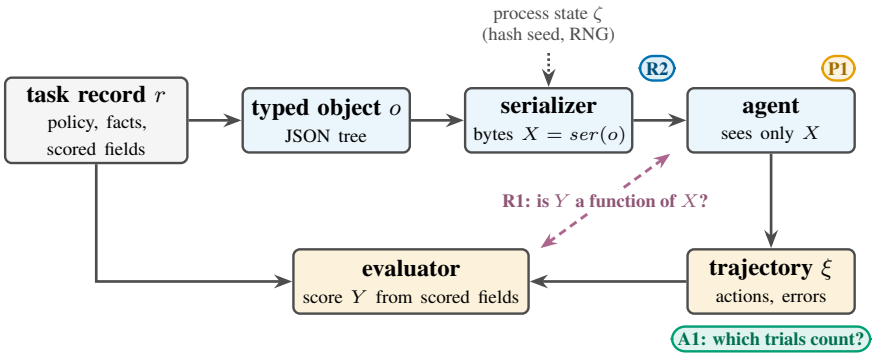


Figure 1. Where attribution can fail. A task record determines what the agent sees (top) and what the evaluator scores (bottom). R1: is the score a function of what the agent sees? R2: is the agent's input unchanged under changes the data format declares meaningless, such as those caused by process state ζ? A1: how are trials with errors or no action counted? P1: does the decision move when R2 fails?

We call the first two conditions *policy observability* and make them checkable (Figure 1). *Sufficiency* (R1) requires that two units the agent cannot tell apart receive the same score. *Presentation invariance* (R2) requires that the bytes the agent sees do not change under transformations that a published specification declares meaningless. For example, RFC 8259 defines a JSON object as an *unordered* collection of members (Bray, 2017), so reordering its members should change nothing. A reporting field (A1) separates completed, failed and unsupported trials, and a model-level property (P1) asks whether a model's decisions change when R2 fails.

Stating these conditions precisely matters. The natural finite version of sufficiency, "no two released tasks with identical prompts receive different labels", is passed by *any* benchmark whose prompts are all different, whatever it scores (Proposition 1). Every benchmark we audit would pass it. We therefore quantify over the benchmark's *coherent* units, those that its own text and declared background knowledge allow, and certify each failure with an exhibited witness. Similarly, we admit only invariances that have a cited published basis, and report NOT-ESTABLISHED when none exists.

We apply the resulting protocol to eight released agent benchmarks (Section 4). Every audit number for the seven further benchmarks is regenerated from pinned upstream code without model calls; the AuthorityBench numbers come from a separate audit of its released records. Sufficiency fails with exhibited witnesses in AuthorityBench (Celebi, 2026), ST-WebAgentBench (Levy et al., 2026) and τ^2-bench (Barres et al., 2026). In the released pipelines of AgentDojo (Debenedetti et al., 2024), AgentHarm (Andriushchenko et al., 2025) and Agent Security Bench (Zhang et al., 2025), what the agent sees depends on process state: the interpreter's string-hash seed or an unseeded shuffle. AgentDojo also scores provider errors as successful attacks. In AgentDojo, changing only the hash seed changes a deterministic agent's score on 6 of 35 affected cases (Section 6).

Do such presentation differences change decisions? A screening study of 330 frontier evaluation attempts on 55 retail action proposals found no excess disagreement between key sorting and candidate-first ordering. Its only excess disagreement, in a secondary contrast, came from three proposals: in the original member order the evaluator allowed the action, and in sorted order it blocked it. We re-examine these proposals with a pre-registered design (Section 5). The flip replicates in every fresh session and is carried by the order of the evidence's top-level members: random top-level orders split the decision, random nested orders never do. The evaluator's stated reasons point to the cause, the policy text and the tool description give two different rules for this case, and a pre-registered test supports it: once the conflict is removed from the text, every order yields the same decision. A stronger evaluator, which passes our competence control, does not flip at all. **Where observability fails, presentation can decide, at least for some evaluators.**

Contributions. (1) Checkable attribution preconditions with certificates and certified bounds (Section 3); (2) an audit of eight released benchmarks and a consequence test in one (Sections 4 and 6); (3) a pre-registered replication, decomposition and disambiguation test of a decision flip

on content-identical evidence, with competence controls (Section 5). All code, run records and pre-registered criteria are released.[1]

2 Related Work

What benchmark scores measure. Label errors change model rankings (Northcutt et al., 2021); construct validity is often under-specified in LLM benchmarks (Bean et al., 2025); agent evaluations raise reproducibility and cost concerns (Kapoor et al., 2024); and the Agentic Benchmark Checklist organizes task and outcome validity for practitioners (Zhu et al., 2025). LLM-BabyBench notes that a single success rate mixes failures of perception, ambiguous instructions, missing knowledge and planning, and removes all but planning by construction (Malek et al., 2025); related guidance argues that final-answer accuracy alone cannot diagnose how a model reached its answer (Nwadike et al., 2026). We give a formal procedure for two specific preconditions, with certificates that make each failure verifiable, and apply it to released benchmarks. Unlike checks that claims can be recomputed from retained evidence (Zhu & Chang, 2026), we ask whether the label was determinable from the agent's input at all.

Underdetermined policies. Two concurrent works are closest. Cao (2026) shows that ambiguous τ^2-bench policies lower scores and increase trial-to-trial inconsistency. Wu & Gong (2026) builds a benchmark of *policy-invisible* violations, whose compliance depends on facts absent from the agent's context. Our sufficiency condition states when such invisibility makes a released score unattributable, and our audits find it in existing benchmarks. The decision experiment adds a mechanism neither work isolates: when the policy text supports both decisions, a presentation difference that the data format declares meaningless decides between them.

Presentation sensitivity. Language models are sensitive to prompt formatting (Sclar et al., 2024), the order of answer options (Pezeshkpour & Hruschka, 2024), premise order (Chen et al., 2024) and the position of information in long contexts (Liu et al., 2024); LLM judges prefer answers by position (Wang et al., 2024; Zheng et al., 2023); content-free wrapper text can flip safety-judge verdicts (Zhou et al., 2026); candidate order changes the decisions of LLM scorers even at equal ranking quality (Frohmann et al., 2026); renaming the options of a typed decision head reverses its decisions (Sun et al., 2026); and framing and social context change whether agents follow embedded rules (Okamoto et al., 2026). Our interventions add or remove no text: they only permute JSON members, which RFC 8259 declares unordered, and released pipelines vary such orders themselves. We follow behavioral invariance testing (Ribeiro et al., 2020) and read the evaluator's stated reasons. Unlike nondeterminism in inference (Atil et al., 2024), the variation we report lives in benchmark code and would remain with a perfectly deterministic model.

3 Attribution Preconditions

Objects. A benchmark has schema-valid records $\mathcal{R}$ (those its loader accepts) and released tasks $T \subseteq \mathcal{R}$. A record r has scored units $u = (r, j)$, for instance one policy of a task. Each unit has a set $F(u)$ of record fields that the released evaluator reads, which we take from the evaluator's source code, not from field names. The evaluator returns $Y(r, j, \xi)$ for a trajectory ξ. The agent observes $X(r, z) = ser(o(r, z))$: a typed, *ordered* tree o (objects are sequences of name–value members) serialized to bytes, where z is sampling randomness such as a user simulator. The unit observation $O(u, z) = (X(r, z), h(u))$ adds the unit's agent-visible handle, such as its rendered policy line; $S(u)$ is its support. A benchmark declares the knowledge K that the agent must bring: $K = \emptyset$ when it measures compliance with a stated policy, and otherwise the unstated norm that the agent must infer. A record is *coherent* if each scored value faithfully formalizes the record's natural-language content together with K; $\mathcal{U}$ denotes coherent units. We assume released records are coherent.

Definition 1 (Sufficiency, R1). *B satisfies* R1 *iff for all $u_1, u_2 \in \mathcal{U}$ and $o_1 \in S(u_1)$, $o_2 \in S(u_2)$: $o_1 = o_2 \Rightarrow Y(u_1, \xi) = Y(u_2, \xi)$ for every trajectory ξ.*

[1] Repository and README: `https://anonymous.4open.science/r/policy-observability-anon-iclr2027-5B6C/README.md`

Equivalently, Y factors through the observation: $Y(u, \xi) = f(O(u, z), \xi)$ for some f (Section A). Two facts shape how the definition must be used.

Proposition 1 (The released-set form is vacuous)**.** *If* R1 *is quantified over released units U instead of $\mathcal{U}$, it holds for every scoring rule whenever distinct released units have disjoint observation supports.*

Released tasks almost always have different prompts, so the released-set form passes almost any benchmark; the counterfactual quantifier is what gives R1 content. A *witness* is a pair of coherent units that violates R1. A *single-mutation certificate* changes one scored field of a released unit to another coherent value that leaves the observation unchanged. A *swap certificate* exchanges the scored fields of two units of one record whose rendered descriptions are identical, so the agent's context stays the same while the scored conditions trade places. Certificates give a certified lower bound w on the fraction of released units that fail. The verdict is FAIL with w; PASS only through a decoder argument that recovers every scored value from the observation; OUT-OF-SCOPE when $K \neq \emptyset$; and NOT-ESTABLISHED otherwise. Scope cannot be replaced by a test:

Proposition 2 (Constant norms are invisible)**.** *Suppose $Y(u, \xi) = g(c(u), \kappa, \xi)$, where $c(u)$ are the unit's scored values and κ (a norm, a scoring polarity) is the same for every unit. Rendering κ to the agent, as a fixed string appended to every observation, does not change whether* R1 *holds.*

Hence no sufficiency test can tell "the agent was never told" a shared norm from "the agent was expected to know" it; the benchmark's declared construct decides, and we quote it. R1 detects hidden *unit-specific* content.

Presentation invariance (R2). Let $\approx_B$ be the equivalence on typed trees generated only by transformations with a cited basis: permuting the members of an object (RFC 8259: an object is an unordered collection), and permuting an array that the benchmark itself builds as a set, shuffles, or documents as unordered. A serialization site is *invariant* if $o \approx_B o'$ implies $ser(o) = ser(o')$ on reachable objects, that is, if it factors through a canonical form. A failure is *realised* when the benchmark's own execution produces two equivalent objects that serialize differently because of uncontrolled process state ζ, such as the string-hash seed. It is *contractual* when a violating pair is exhibited but not observed in execution. The verdict is PASS, FAIL-realised, FAIL-contract, or NOT-ESTABLISHED when no equivalence with a cited basis applies. The mere absence of a canonical form is never counted as evidence.

Attribution (A1). Scored trials E split, in this order of precedence, into unsupported trials $E_\emptyset$ (a score with no trajectory), infrastructure failures E_f, and completed trials E_c. We report the share of trials with an action, the completed-trial violation rate $\rho_c = |V|/|E_c|$ for violations $V \subseteq E_c$, $|E_f|$, $|E_\emptyset|$, and how the benchmark scores E_f.

Proposition 3 (Scoring failures as violations)**.** *If a benchmark reports $\hat{\rho} = (|V|+|E_f|)/(|E_c|+|E_f|)$, then $\hat{\rho} - \rho_c = |E_f|(1 - \rho_c)/(|E_c| + |E_f|) \geq 0$, with equality iff $|E_f| = 0$ or $\rho_c = 1$.*

Decisions (P1). For a decision model M, a prompt x and the scaffold $s = \{$`"decision": "`, let ℓ_d be the single token of label $d \in \{\text{ALLOW}, \text{BLOCK}, \text{ASK}\}$ (verified per model and prompt) and $V_"$ the tokens whose text begins with a quotation mark. The scored event is the string "d followed by a closing quote":

$$q_d(x) = P_M(\ell_d \mid x \cdot s) \sum_{v \in V_"} P_M(v \mid x \cdot s \cdot \ell_d), \qquad p_d(x) = q_d(x) / \textstyle\sum_{d'} q_{d'}(x). \tag{1}$$

Two content-identical presentations ω, ω' ($o_\omega \approx_B o_{\omega'}$) show a *deterministic flip* if $\arg\max_d p_d$ or the parsed greedy decision differs. For a sampled model, evidence must control for repeat variation: with two independent replicates per arm, the excess $\text{cross} - \text{within}$ has expectation $\frac{1}{2}\|p_A - p_B\|_2^2$ (Proposition 6), which is zero exactly when the two arms' decision distributions are equal. P1 fails when a deterministic flip or a repeat-controlled excess is exhibited. Showing that it holds would require enumerating an equivalence class whose size is a product of factorials, which is infeasible for realistic evidence packets.

Table 1. Attribution preconditions across eight released benchmarks. R1 FAIL rests on exhibited witnesses; certified quantities are lower bounds. R2 *realised*: the variation occurs in the benchmark's own execution. E_f: infrastructure failures; $E_\emptyset$: scores with no trajectory.

benchmark	R1 sufficiency	R2 presentation invariance	A1 attribution
AuthorityBench	FAIL: 34 witness pairs, 21 scenarios	not audited	4,567/6,924 trials without a tool call
ST-WebAgentBench	FAIL: certified $w \geq 35.9\%$ (1,098/3,057 policies); indicator 74.6%	NOT-ESTABLISHED: policy order unpinned (355 of 356 multi-policy tasks), no declared basis	not audited
τ^2-bench retail	FAIL: witness (task 68)	FAIL-contract: two unsorted agent-visible serializers beside one sorted	not audited
AgentDojo	OUT-OF-SCOPE	FAIL-realised: 35/726 transcripts change with the hash seed	91 E_f scored as attacks; 207 $E_\emptyset$
ToolEmu	OUT-OF-SCOPE (negative control)	NOT-ESTABLISHED: toolkit order, no declared basis	no trajectories
R-Judge	OUT-OF-SCOPE	FAIL-contract: 571/571 records change under key reordering	non-answers scored as errors
AgentHarm	not audited	FAIL-realised: unseeded shuffle, 355/416 behaviors	no trajectories
ASB	not audited	FAIL-realised: hash seed, 345/400 constructed workflows	no trajectories

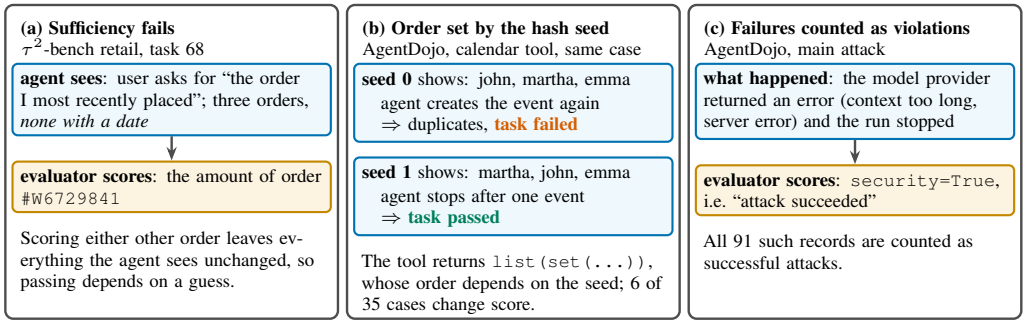


Figure 2. One real example of each failure. (a) The scored order cannot be identified from anything the agent sees. (b) A set's order, fixed by the interpreter's hash seed, reaches the agent and decides the score (Section 6). (c) Provider errors are scored as successful attacks.

4 Auditing Released Benchmarks

We applied the procedure to eight benchmarks (Table 1). Each audit fixes an upstream commit, hashes every file it reads, freezes its pass criterion before any script runs, and takes the scored fields from the evaluator's source code. No audit calls a model, and a fresh re-run of every script reproduces all committed outputs (Section D); Figure 2 gives examples.

Sufficiency witnesses. *AuthorityBench* shows the agent the names of authorized tools but not the argument and history constraints that its oracle checks. We construct 34 pairs of scenario variants in 21 scenarios (30 differ in an argument constraint, 4 in a history constraint) whose prompts are byte-identical, while the benchmark's own oracle labels the same candidate call oppositely. No function of the prompt can separate them. A replay of the 7,974 released records (6,924 of them valid trials) against the current scenario definitions gives 65 call-level discrepancies; a documented reconstruction of the historical definitions resolves all 65. *ST-WebAgentBench* shows all 3,057 of its policies to the agent, but its evaluator reads locator, content and count fields that are never shown. In 185 of 375 tasks, 1,500 pairs of policies have the same description, source, category and template but different scored conditions. Exchanging the scored fields within a pair leaves the agent's context byte-identical and changes the score of an exhibited trajectory. These swap witnesses certify $w \geq 1{,}098/3{,}057 = 35.9\%$ failing policies; they decide *which* policy a violation is attributed to, not

how many. In τ^2-*bench retail* task 68 (Figure 2a), the score depends on "the order most recently placed", but no order carries a date and the user cannot give one. Scoring another of the user's orders instead leaves every agent-visible input unchanged (the transcript is generated without reading the evaluation fields) and flips the score of an exhibited trajectory.

Presentation that depends on process state. In *AgentDojo*, `create_calendar_event` returns `list(set(participants))` (Figure 2b). Running the shipped ground-truth pipeline under three hash seeds changes the agent-visible bytes of 35 of 726 cases, while a rerun with the same seed is byte-identical once the wall clock is masked. The released trajectories confirm it: 10 of the 18 participant sets returned in them appear in two or more orders. *Agent Security Bench* builds the agent's workflow with `list(set(...))`, so the attacker tool's position depends on the hash seed: in 400 constructed agent–attacker workflows, 345 differ between two seeds and none between two runs with the same seed. *AgentHarm* shuffles each behavior's tool list with the global random generator, which is never seeded, so 355 of 416 behaviors show tools in a different order in two fresh processes. None of these pipelines records the state that determined what the agent saw.

Contract failures and non-findings. In τ^2-*bench*, tool results and the agent's own earlier arguments reach the agent through two unsorted `json.dumps` calls, while one read tool returns sorted JSON. *R-Judge* (Yuan et al., 2024) renders agent turns from JSON objects in file order, so every record changes under key reordering, although the release uses one order throughout. ToolEmu (Ruan et al., 2024), our negative control, is OUT-OF-SCOPE: it declares that agents must infer an unstated safety norm. The procedure also passes units: 23 of the 24 sampled τ^2-bench tasks pass its per-task check that every scored condition is recoverable from the agent's input, and τ^2-bench's product-lookup serializer sorts keys and is invariant. Policy order in ST-WebAgentBench and toolkit order in ToolEmu vary, but no document declares these orders meaningless, so R2 is NOT-ESTABLISHED rather than FAIL.

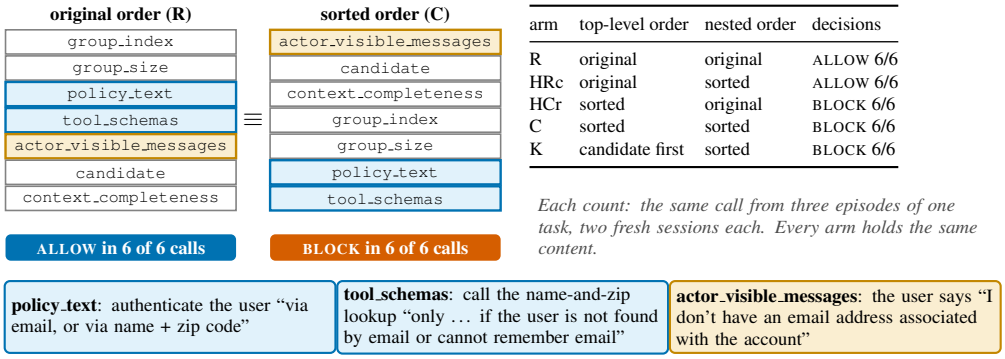
Attribution. AgentDojo sets `security=True`, an attack success, whenever the provider returns a context-length or server error (Figure 2c); all 91 such records under its main attack are scored as successes. Three conventions are possible. The code's (errors count as attacks) reproduces only 6 of the 10 rows of the paper's Table 3 (e.g. Llama-3-70B: 25.60% from the released flags, 20.03% in the paper). Counting errors as non-attacks while keeping them in the denominator reproduces all 10. So the published table used neither the code's convention nor exclusion of the errors, and a reader cannot tell which from the paper; Proposition 3 gives the size of such differences. In AuthorityBench, 4,567 of 6,924 eligible trials contain no tool call, so a violation rate over all trials mixes inaction into its denominator.

5 Does Presentation Change Decisions?

R2 failures are properties of pipelines. Whether they matter depends on models. We test this on an LLM authorization judge, the component that decides whether an agent's proposed action is permitted, where a registered screening study found its only excess disagreement.

The screening study. A frontier evaluator (gpt-6-astra at low reasoning effort, through the Codex command-line interface at the provider's default sampling) judged 55 action proposals from three τ^2-bench retail tasks. Each proposal is an *evidence packet*: a JSON object with seven top-level members, including the policy text, the tool schemas, the conversation so far and the candidate call. Each packet was shown in three arms: original member order (R), recursively sorted keys (C), and C with the candidate first (K), with two fresh sessions per arm (330 attempts, 318 valid). The pre-specified K–C contrast showed zero estimated excess disagreement over the 50 proposals with all four K and C sessions valid (cross- and within-arm disagreement both 1%). The secondary R–C contrast showed 5.77 points of excess (3 of the 52 proposals complete for R and C), all from three proposals: the same call, `find_user_id_by_name_zip(Noah, Patel, 10108)`, in three episodes of one task. R allowed it and C and K blocked it in both sessions.

Design. We pre-registered a follow-up before any model call (Section C). All arms are built with the screening's renderer (R, C and K byte-identical to the released renderings) and pass its typed-tree



arm	top-level order	nested order	decisions
R	original	original	ALLOW 6/6
HRc	original	sorted	ALLOW 6/6
HCr	sorted	original	BLOCK 6/6
C	sorted	sorted	BLOCK 6/6
K	candidate first	sorted	BLOCK 6/6

Figure 3. The flip replicates and is carried by top-level member order. Left: the seven top-level members in the original (R) and sorted (C) orders; the content is identical (≡). Right: frontier decisions on the three flipped proposals, two fresh sessions each. Bottom: the three members whose text bears on the decision.

check for identical content. New arms separate the order of the seven top-level members from the order inside nested objects: HRc (R's top-level order, nested objects sorted) and HCr (sorted top level, R's nested order); 52 random orders of every object (P); 12 random top-level orders with nested objects sorted (T); and 12 random nested orders under the sorted top level (N). The frontier evaluator (the same model, transport and parser as the screening) was called in fresh sessions; 33 of its calls were first refused for an account spending limit and were run once each when the limit was lifted, as the criterion prescribes. Three open-weight models (Llama-3.1-8B-Instruct, Grattafiori et al., 2024; gemma-3-27b-it, Gemma Team, 2025; Qwen3-32B with thinking disabled, Yang et al., 2025) were served deterministically with vLLM (Kwon et al., 2023), and their label probabilities read by Equation (1). A control set of 20 unambiguous proposals (10 ALLOW, 10 BLOCK, two of them shortcut detectors) tests whether a model can do the task at all.

The flip replicates and is carried by top-level order (Figure 3). On all three proposals, R is ALLOW and C and K are BLOCK in both sessions, a repeat-controlled excess of 1.0 per proposal. HRc behaves like R and HCr like C on every proposal: the order of the seven top-level members carries the effect. The pre-registered ablation confirms it on the first proposal: all 12 random nested orders under the sorted top level are BLOCK (like C), while 12 random top-level orders split 7 ALLOW to 5 BLOCK (Fisher two-sided $p = 0.0046$).

On identical content, the decision is close to a coin flip (Figure 4). Over the 52 random orders of the first proposal, the evaluator approves 30 and blocks 22 (ALLOW 57.7%, Wilson 95% interval 44.2–70.1%), with no failures; the entropy across orders is 0.681 nats, close to the two-label maximum $\ln 2 = 0.693$. The split is not session noise: for a fixed order the evaluator is nearly deterministic. Its 15 same-arm session pairs in Figure 3 (3 proposals $\times$ 5 arms) never disagree, and within-arm disagreement in the screening was 1%. We did not repeat each random order, so per-order determinism is inferred from the fixed orders.

What the order selects (exploratory). These analyses were specified after the replication. Over the 64 random orders with a varying top level, the evaluator approves 23 of 29 (79.3%) when the policy text comes before the conversation and 14 of 35 (40.0%) when it comes after (two-sided Fisher $p = 0.0022$; one of several possible position contrasts, not corrected for multiplicity). Its stated reasons point to the same account. We coded all 93 reasons recorded before the resumed calls (30 fixed-order and 63 random-order decisions), blind to presentation (Section C). All 45 BLOCK reasons apply the tool description's precondition (use the name-and-zip lookup "only" if the user is not found by email or cannot remember it) and find it unmet. Of the 48 ALLOW reasons, 13 address that precondition and judge it met because no email exists, and 35 justify the lookup by the authentication requirement without mentioning it. The policy text allows authentication "via email, or via name + zip code", and the user said there is no email address on the account. Both readings are defensible and no human label decides between them.

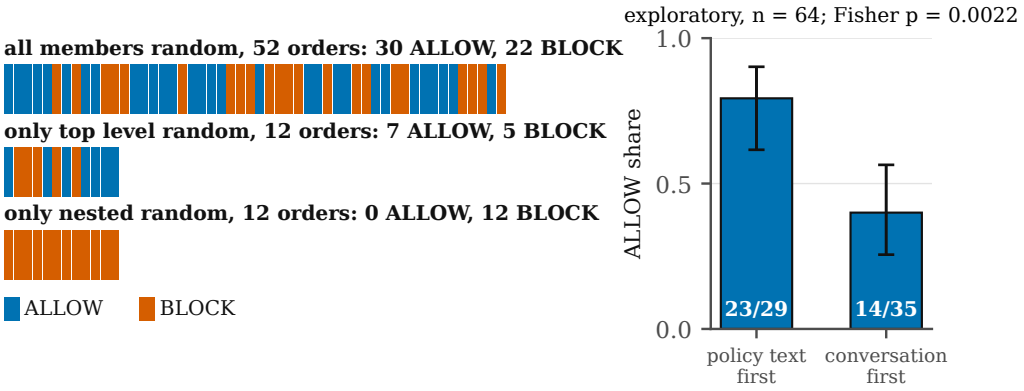


Figure 4. On identical content, the decision is close to a coin flip. Left: frontier decisions on 52 random orders of all members, 12 of the top level only, and 12 of nested members only (all BLOCK). Right (exploratory): approval share by whether the policy text precedes the conversation, over the 64 orders with a varying top level (Wilson 95%).

Competence control. No evaluator we tested passes the pre-registered rule (at least 18 of 20 correct, both shortcut detectors right) except Claude Opus 5.5, below. gpt-6-astra answers 14 of 20 correctly: every prohibited action, including both detectors, is BLOCK, and the six misses are permitted calls it blocks because their arguments (four calculator expressions, two product ids) appear nowhere in the visible conversation. This is a failure as pre-registered, but also a flaw in our control items; Opus 5.5's two misses are of the same kind.

Open-weight models. Llama-3.1-8B (0 of 20), gemma-3-27b (5) and Qwen3-32B (3) fail the control by a wide margin, so their results say nothing about capable evaluators; we report them in Section C. Under exact determinism they are still presentation-sensitive (Llama's top label differs between K and C on 12 of the 55 screened proposals), and gemma, like gpt-6-astra, moves only with top-level order (pre-registered Fisher $p = 0.037$; Figure 6).

Disambiguation (pre-registered). If presentation decides only where the texts conflict, removing the conflict should remove the split. In variant D1 the user says they cannot remember their email, which satisfies the tool's precondition; D2 instead drops the tool's email-first sentence. We re-ran the random orders with each original member order. For gpt-6-astra, D1 removes the split entirely: all 10 fixed-order calls and all 51 recorded random orders are ALLOW (one call was refused at the account limit), against 30 ALLOW of 52 originally; 21 of the 51 order-matched pairs change, all from BLOCK to ALLOW (McNemar exact $p = 9.5 \times 10^{-7}$). D2 could not be run before the account limit. Among the open-weight models, gemma's split (ALLOW versus its default ASK, not between the two rules) survives both edits (McNemar $p = 0.125, 0.25$), Qwen splits on only 3 of 52 orders, and Llama answers ASK throughout.

Anthropic evaluators (partial). We ran the same design, with identical prompt bytes and parser, on three Anthropic models through Claude Code; the runs stopped at an account limit after the control set, the fixed orders and part of D1. Claude Opus 5.5 (low effort) *passes* the control (18 of 20, both detectors correct) and approves the call at every fixed order of all three proposals in both sessions (30 of 30): no flip. On the 52 random orders of the original packet it approves every time (52 of 52), and under D1 all 24 recorded calls. Claude Sonnet 5 fails the control at medium and at low effort (15 of 20 each) and approves at every fixed order and, at low effort, on all 48 valid all-member random orders; Claude Haiku 4.5 fails the control (8 of its 11 valid answers right) and approves in all 80 of its valid order calls. The flip is therefore not a property of every frontier evaluator.

Scope of the claim. The effect is local: three proposals of one call type, selected because they flipped, and one evaluator that fails our control. It shows that a presentation difference the data format declares meaningless *can* decide an authorization judge's decision where the texts conflict; it does

Table 2. The hash seed changes a deterministic agent's scores. The 35 AgentDojo cases whose agent-visible bytes depend on the seed; transcripts compared without the (unrendered) tool-call ids.

	seed 0	seed 0 (rerun)	seed 1	seed 2
transcript differs from seed 0	–	0/35	23/35	19/35
first difference at the calendar result	–	–	23/23	19/19
agent's later actions differ	–	–	14	12
score differs from seed 0	–	0/35	6/35	0/35
utility, no attack	0/5	0/5	1/5	0/5
utility under attack	8/30	8/30	13/30	8/30
targeted attack success	0/30	0/30	0/30	0/30

not estimate how often, it measures consistency, not correctness, and the stronger evaluator we tested (Opus 5.5) shows no presentation dependence here at all.

6 Does Process State Change Scores?

Does AgentDojo's hash-seed dependence change what a deterministic agent does, and its score? Under a pre-registered design, Llama-3.1-8B-Instruct ran the 35 affected cases (five workspace user tasks, each with no injection and with six injection tasks) through the shipped native tool-calling pipeline on a deterministic server, in four processes: seed 0 twice (the same-seed control), seeds 1 and 2 once. Two declared deviations apply to every condition: a fixed clock, and errors recorded rather than scored (none occurred).

Results (Table 2). The same-seed control holds exactly: the two seed-0 processes produce identical transcripts on all 35 cases, so every difference is caused by the seed. With seed 1, 23 of 35 transcripts differ, each first at the participant list returned by `create_calendar_event`; in 14 the agent then acts differently, and 6 cases change score (Wilson 95% interval 8.1–32.7%, treating cases as independent; all six belong to one user task, so the effective sample is one task, not 35 cases). Seed 2 changes 19 transcripts and no score. Attack success is 0/30 in every condition for this agent, so security is not affected here.

Mechanism (Figure 2b). All six score changes come from one user task: create a follow-up meeting with the participants of an earlier one. The benchmark returns the new event with its participants in a seed-dependent order; under seed 0 the agent then creates the same event again, leaving duplicates, and fails; under seed 1 it stops and passes. Task, content and model are the same: the order of a set, fixed by process state that no record keeps, decides the score.

7 Discussion

What a reported violation rate can mean. A scored violation can come from the agent's decision, from information the benchmark withheld, or from the presentation the pipeline produced; the rate alone does not separate them, and we found the second and third in released benchmarks.

Recommendations. Benchmark authors should state the declared knowledge K and where each scored field is visible to the agent; serialize agent-visible objects canonically; seed every random generator and record the process state behind each observation; and report attempt rates and how failures are scored next to every violation rate.

Limitations. Sufficiency failures are existential: witnesses show that units fail, and certified quantities are lower bounds. Whether a constructed counterfactual is coherent is a judgment by the authors, recorded with its rationale but not checked by benchmark authors or independent annotators. Scope rests on each benchmark's stated construct, which we quote but cannot test (Proposition 2). The decision experiment is deliberately narrow: three proposals of one call, selected because they flipped; a flipping frontier evaluator that fails our competence control (partly because of flawed control items); a stronger evaluator, Opus 5.5, that neither flips nor splits; three open-weight models that fail the control; and no human permission labels. It therefore shows existence, not prevalence

or accuracy. Several audits lack attribution fields because no trajectories are released. Findings are pinned to the named commits; some may since have been fixed upstream.

8 Conclusion

A policy violation is evidence about a model only if the scored condition was observable to it, what it observed did not depend on choices the benchmark treats as meaningless, and the trial was one in which it acted. We made these conditions checkable, found each one violated in released agent benchmarks, and showed that when the text leaves a decision open, a meaningless change of presentation can decide it for an evaluator below our competence bar.

References

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. In *International Conference on Learning Representations*, 2025. arXiv:2410.09024.

Berk Atil, Sarp Aykent, Alexa Chittams, Lisheng Fu, Rebecca J. Passonneau, Evan Radcliffe, Guru Rajan Rajagopal, Adam Sloan, Tomasz Tudrej, Ferhan Ture, Zhe Wu, Lixinyu Xu, and Breck Baldwin. Non-determinism of "deterministic" LLM settings, 2024.

Victor Barres, Honghua Dong, Soham Ray, Xujie Si, and Karthik Narasimhan. τ^2-bench: Evaluating conversational agents in a dual-control environment. In *International Conference on Machine Learning*, 2026. arXiv:2506.07982.

Andrew M. Bean, Ryan Othniel Kearns, Angelika Romanou, Franziska Sofia Hafner, Harry Mayne, Jan Batzner, et al. Measuring what matters: Construct validity in large language model benchmarks. In *Advances in Neural Information Processing Systems, Datasets and Benchmarks Track*, 2025. arXiv:2511.04703.

Tim Bray. The JavaScript Object Notation (JSON) data interchange format. RFC 8259, Internet Engineering Task Force, 2017.

Hongliu Cao. Policy loopholes in agent evaluation: When policy ambiguity masquerades as agent error, 2026.

Yaz Celebi. Can is not may: Authority models for governable AI agents. `https://github.com/yazcaleb/can-is-not-may`, 2026. Preprint and AuthorityBench release.

Xinyun Chen, Ryan A. Chi, Xuezhi Wang, and Denny Zhou. Premise order matters in reasoning with large language models. In *International Conference on Machine Learning*, pp. 6596–6620, 2024. PMLR 235. arXiv:2402.08939.

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 *Advances in Neural Information Processing Systems, Datasets and Benchmarks Track*, 2024. arXiv:2406.13352.

Markus Frohmann, Mahdiyar Alavi, Elizabeth Lingg, and Navid Rekabsaz. Equal ranking quality, different decisions: Training order-consistent LLM scorers, 2026.

Gemma Team. Gemma 3 technical report, 2025.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, et al. The Llama 3 herd of models, 2024.

Sayash Kapoor, Benedikt Stroebl, Zachary S. Siegel, Nitya Nadgir, and Arvind Narayanan. AI agents that matter, 2024.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with PagedAttention. In *Proceedings of the 29th Symposium on Operating Systems Principles*, 2023. arXiv:2309.06180.

Ido Levy, Ben Wiesel, Sami Marreed, Alon Oved, Avi Yaeli, Nir Mashkif, and Segev Shlomov. ST-WebAgentBench: A benchmark for evaluating safety and trustworthiness in web agents. In *International Conference on Learning Representations*, 2026. arXiv:2410.06703.

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*, 2024. arXiv:2307.03172.

Idriss Malek, Omar Choukrani, Daniil Orel, Anh Duy Le Dinh, Zhuohan Xie, Zangir Iklassov, Martin Takáč, and Salem Lahlou. LLM-BabyBench: Can language models plan in worlds they can simulate?, 2025.

Curtis G. Northcutt, Anish Athalye, and Jonas Mueller. Pervasive label errors in test sets destabilize machine learning benchmarks. In *Advances in Neural Information Processing Systems, Datasets and Benchmarks Track*, 2021. arXiv:2103.14749.

Munachiso Samuel Nwadike, Zangir Iklassov, Kareem Ali, Rifo Genadi, and Kentaro Inui. Measuring AI reasoning: A guide for researchers, 2026.

Mika Okamoto, Ansel Kaplan Erol, and Kutluhan Erol. Why do AI agents break rules? How framing, context, and social signals shape compliance, 2026.

Pouya Pezeshkpour and Estevam Hruschka. Large language models sensitivity to the order of options in multiple-choice questions. In *Findings of the Association for Computational Linguistics: NAACL 2024*, 2024. arXiv:2308.11483.

Marco Tulio Ribeiro, Tongshuang Wu, Carlos Guestrin, and Sameer Singh. Beyond accuracy: Behavioral testing of NLP models with CheckList. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020. arXiv:2005.04118.

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. In *International Conference on Learning Representations*, 2024. arXiv:2309.15817.

Melanie Sclar, Yejin Choi, Yulia Tsvetkov, and Alane Suhr. Quantifying language models' sensitivity to spurious features in prompt design or: How I learned to start worrying about prompt formatting. In *International Conference on Learning Representations*, 2024. arXiv:2310.11324.

Yu Sun, Junhao Xu, Jiajia Shi, and Zijin Yang. Type-safe is not error-free: A constrained decision head follows the option name, not the rubric bound to it, 2026.

Peiyi Wang, Lei Li, Liang Chen, Zefan Cai, Dawei Zhu, Binghuai Lin, Yunbo Cao, Lingpeng Kong, Qi Liu, Tianyu Liu, and Zhifang Sui. Large language models are not fair evaluators. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics*, 2024. arXiv:2305.17926.

Jie Wu and Ming Gong. Policy-invisible violations in LLM-based agents, 2026.

An Yang, Anfeng Li, Baosong Yang, et al. Qwen3 technical report, 2025.

Tongxin Yuan, Zhiwei He, Lingzhong Dong, Yiming Wang, Ruijie Zhao, Tian Xia, Lizhen Xu, Binglin Zhou, Fangqi Li, Zhuosheng Zhang, Rui Wang, and Gongshen Liu. R-Judge: Benchmarking safety risk awareness for LLM agents. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, 2024. arXiv:2401.10019.

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. In *International Conference on Learning Representations*, 2025. arXiv:2410.02644.

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, Datasets and Benchmarks Track*, 2023. arXiv:2306.05685.

Yongxi Zhou, Wenbo Ye, Yuanzhe Liu, Zihan Dong, and Junwei Yao. Style over substance: Content-invariant wrappers flip LLM safety-judge verdicts, 2026.

Peiying Zhu and Sidi Chang. ClaimReceipt: Verifying evidence sufficiency and coverage in agent evaluations, 2026.

Yuxuan Zhu, Tengjun Jin, Yada Pruksachatkun, Andy Zhang, Shu Liu, Sasha Cui, Sayash Kapoor, Shayne Longpre, Kevin Meng, Rebecca Weiss, Fazl Barez, Rahul Gupta, Jwala Dhamala, Jacob Merizian, Mario Giulianelli, Harry Coppock, Cozmin Ududec, Jasjeet Sekhon, Jacob Steinhardt, Antony Kellermann, Sarah Schwettmann, Matei Zaharia, Ion Stoica, Percy Liang, and Daniel Kang. Establishing best practices for building rigorous agentic benchmarks. In *Advances in Neural Information Processing Systems, Datasets and Benchmarks Track*, 2025. arXiv:2507.02825.

A PROOFS

This appendix proves the statements of Section 3 and defines the certificates used in the audits. Notation is that of Section 3.

A.1 SUFFICIENCY (R1)

Proposition 4 (Factorization)**.** R1 *holds iff there is a map f with $Y(u,\xi) = f(o,\xi)$ for every $u \in \mathcal{U}$, $o \in S(u)$ and trajectory ξ.*

Proof. ($\Leftarrow$) If $o \in S(u_1) \cap S(u_2)$, then $Y(u_1,\xi) = f(o,\xi) = Y(u_2,\xi)$. ($\Rightarrow$) Set $f(o,\xi) = Y(u,\xi)$ for any $u \in \mathcal{U}$ with $o \in S(u)$, and arbitrarily where there is none; R1 makes this well defined. □

Proof of Proposition 1. If distinct released units have disjoint supports, then $o_1 = o_2$ with $o_i \in S(u_i)$ forces $u_1 = u_2$, and the conclusion $Y(u_1,\xi) = Y(u_2,\xi)$ holds trivially for any Y. □

Proof of Proposition 2. Let O' append a fixed string to every observation. Appending is injective, so

$$O'(u_1,z_1) = O'(u_2,z_2) \iff O(u_1,z_1) = O(u_2,z_2),$$

and supports intersect under O' exactly when they intersect under O. Neither Y nor $\mathcal{U}$ changes, because κ is not a field of any record. □

Certificates. A *single-mutation certificate* for a released unit $u = (t,j)$ is a field $f \in F(u)$ and a value $v \neq t[f]$ with $r = t[f \mapsto v]$ schema-valid, such that

(i) $o(r,z) = o(t,z)$ and $h(r,j) = h(t,j)$ for every z;
(ii) some exhibited trajectory ξ^* has $Y((r,j),\xi^*) \neq Y((t,j),\xi^*)$;
(iii) r is coherent.

Then u fails: by (iii) $(r,j) \in \mathcal{U}$, by (i) the supports coincide, and (ii) gives unequal scores. Part (i) is mechanical for a deterministic renderer; part (ii) is mechanical when the evaluator is a string-inclusion test and neither value is a substring of the other; part (iii) is a judgment.

A *swap certificate* exchanges the evaluator fields of two units of one record that agree on every non-evaluator field. The rendered context is then unchanged. Coherence of the swapped record follows when no content of the record refers to units by position (checked on all 30,803 strings of ST-WebAgentBench's records) and the benchmark sets precedence by source rather than position, as ST-WebAgentBench states.

A *decoder certificate*, a map recovering every scored value from every observation of every coherent unit, is sufficient for R1 (compose with Proposition 4) but not necessary.

A.2 PRESENTATION INVARIANCE (R2)

Proposition 5 (Invariance and canonicalization)**.** *Let $\mathcal{O}$ be the reachable objects, closed under $\approx_B$. A serialization site is invariant on $\mathcal{O}$ iff $ser = s \circ \kappa$ on $\mathcal{O}$ for some s and some κ that is constant on $\approx_B$-classes. A realised failure violates invariance.*

Proof. ($\Leftarrow$) $o \approx_B o'$ gives $\kappa(o) = \kappa(o')$, hence $ser(o) = ser(o')$. ($\Rightarrow$) Let κ pick a fixed representative of each class and set $s = ser$. For the last claim: the two objects produced under process states ζ, ζ' are both equivalent to the nominal reachable object, hence to each other, and they serialize differently. □

A.3 ATTRIBUTION (A1)

Proof of Proposition 3. With $\rho_c = |V|/|E_c|$,

$$\hat{\rho} - \rho_c = \frac{|E_c|\,(|V| + |E_f|) - |V|\,(|E_c| + |E_f|)}{|E_c|\,(|E_c| + |E_f|)} = \frac{|E_f|\,(|E_c| - |V|)}{|E_c|\,(|E_c| + |E_f|)} = \frac{|E_f|\,(1 - \rho_c)}{|E_c| + |E_f|} \;\geq\; 0,$$

with equality iff $|E_f| = 0$ or $\rho_c = 1$. □

A.4 DECISIONS (P1)

Proposition 6 (What excess identifies)**.** *For arms A, B with decisions $a_1, a_2 \sim p_A$ and $b_1, b_2 \sim p_B$, all independent, let*

$$\text{cross} = \tfrac{1}{4}\sum_{k,l} \mathbf{1}[a_k \neq b_l], \qquad \text{within} = \tfrac{1}{2}\big(\mathbf{1}[a_1 \neq a_2] + \mathbf{1}[b_1 \neq b_2]\big).$$

Then $\mathbb{E}[\text{cross} - \text{within}] = \frac{1}{2}\|p_A - p_B\|_2^2$.

Proof. $\mathbb{E}\,\mathbf{1}[a_1 \neq a_2] = 1 - \|p_A\|^2$, $\mathbb{E}\,\mathbf{1}[b_1 \neq b_2] = 1 - \|p_B\|^2$ and $\mathbb{E}\,\mathbf{1}[a_k \neq b_l] = 1 - \langle p_A, p_B\rangle$, so

$$\mathbb{E}[\text{cross} - \text{within}] = \tfrac{1}{2}\|p_A\|^2 - \langle p_A, p_B\rangle + \tfrac{1}{2}\|p_B\|^2 = \tfrac{1}{2}\|p_A - p_B\|_2^2. \qquad \square$$

Positive excess is evidence of presentation dependence only under independent replicates with common settings: randomness shared within an arm but not across arms would produce positive excess with $p_A = p_B$. Each frontier call is a fresh session.

Lemma 1 (Entropy and top mass)**.** *Let p be a distribution on $n \geq 2$ labels with top mass m, and let $h(m) = -m\log m - (1-m)\log(1-m)$. Then*

$$H(p) \geq -\log m > \log 2 \qquad \text{if } m < \tfrac{1}{2},$$
$$h(m) \leq H(p) \leq h(m) + (1-m)\log(n-1) \qquad \text{if } \tfrac{1}{2} \leq m < 1,$$

and $H(p) = 0$ if $m = 1$. For $n = 3$: $H(p) < 0.1 \Rightarrow m > 0.979494$, and $m > 0.982558 \Rightarrow H(p) < 0.1$.

Proof. Shannon entropy is at least the min-entropy $-\log m$. For $\frac{1}{2} \leq m < 1$, grouping gives $H(p) = h(m) + (1-m)H(\tilde{p})$, with $\tilde{p}$ the renormalized remainder and $0 \leq H(\tilde{p}) \leq \log(n-1)$. For $n = 3$, $H < 0.1$ excludes $m < \frac{1}{2}$. On $[\frac{1}{2}, 1)$, h is strictly decreasing with $h(m) = 0.1$ at $m = 0.979494$, and $h(m) + (1-m)\log 2$ is strictly decreasing (derivative $\log\frac{1-m}{2m} < 0$) with value 0.1 at $m = 0.982558$. □

The lemma fixes the control's confidence gate: requiring $H < 0.1$ already implies a top label above 0.979, so a separate top-mass threshold of 0.9 is redundant. The sufficient threshold 0.982558 is not necessary: $p = (0.98, 0.02, 0)$ has $H = 0.0980 < 0.1$.

B AUDIT DETAILS

Every audit follows the same protocol: pin the upstream commit and hash every file read; freeze the pass criterion before any script runs (later changes are dated and recorded, never edits); take scored fields from evaluator source; quote the benchmark's own construct statement for scope; and separate mechanical counts from judgments. The repository holds, per benchmark, the criterion, findings, scripts (standard library unless noted) and committed outputs.

AuthorityBench. The 34 witness pairs are constructed records changing one hidden argument or history constraint of a scenario; the prompt builder does not read these constraints, so the prompts are byte-identical in all 34 pairs, and the benchmark's oracle labels the constructed candidate call oppositely. Coherence is a judgment that the scenario's text does not fix the constraint. Each pair moves a trial label. The release we audited contains 7,974 records (6,924 valid trials); the benchmark's paper reports 7,427 trials over 54 scenarios, and we did not reconcile the two counts. The replay audit and pair construction reproduce from the released records (18 of 19 checks of the original AuthorityBench study pass; the other differs by one test in an environment-dependent suite; these checks are separate from the 24 comparisons of Section D).

ST-WebAgentBench. Commit `67f56dd`, 375 tasks, 3,057 policies. The renderer emits a policy's description, source and category; the evaluator reads an evaluation block. A field-by-field mutation test (49,353 single-field mutations) confirms the renderer never reads the evaluation block. Of the 1,500 twin pairs, all share category, template and every non-evaluator field; a distinguishing trajectory is certified mechanically for 1,192 pairs, covering 1,098 distinct policies. The indicator counts policies with a scored value literally absent from the rendered text (2,281); we withhold a projected rate because its estimator could not be reproduced.

τ^2-bench retail. Of 25 tasks drawn at seed 20260925 from the 111 tasks that reference orders, 24 were evaluated (one user could not be resolved); task 68 exhibits a witness. The detection rate $1/24$ (Wilson 95% [0.74%, 20.24%]) is an audit-detection rate under the audit's rule, not an estimate of the failing fraction, because the other 23 tasks pass only the audit's per-task recoverability check, which is not a proof of R1 for them. Serialization sites: `environment.py:365` (tool results, unsorted), `llm_utils.py:154` (the agent's own arguments, unsorted), and `tools.py:409` (one read tool, sorted).

AgentDojo. Commit `089ed46`, suite v1 (97 user tasks, 629 security cases). The ground-truth pipeline was run in separate processes under hash seeds 0, 0, 1 and 2; after masking wall-clock timestamps the two seed-0 runs are byte-identical on every case, and 35 of 726 transcripts differ between seed 0 and seeds 1 or 2, all first at a `create_calendar_event` participant list. A further 78 differ with the wall clock alone; this changes content, not presentation, and is not counted under R2. Released records: 91 infrastructure errors under the main attack, all scored `security=True`; 207 claude-3-opus records carry scores with empty transcripts. Paper Table 3 reproduces in 10 of 10 rows only when errors count as non-successes within the denominator; the "No defense" entry of the paper's Table 5 (57.69%) matches neither reading and differs from its Table 3 value for the same configuration (47.69%).

ToolEmu. All 144 cases have pairwise distinct prompts in each agent variant, so the released-set form of R1 holds vacuously; a literal content-presence test flags 144 of 144 because the rubric is withheld by design. The benchmark's construct, safe handling of underspecified instructions, declares $K \neq \emptyset$. Reordering toolkits changes the prompt in 49 of 49 multi-toolkit cases; no document declares toolkit order meaningless.

R-Judge, AgentHarm, ASB. R-Judge renders agent turns from JSON objects; all 571 records change bytes under key reordering, while the release uses a single order and no differing execution was observed. AgentHarm's solver shuffles each behavior's tools with the unseeded global generator; comparing two fresh processes, 355 of 416 behaviors differ in tool order. ASB's injection function places the attacker tool via `list(set(...))`; on 400 constructed workflows, 345 differ between seeds 0 and 1, 321 between 0 and 2, and 0 between two seed-0 runs. Neither AgentHarm nor ASB releases trajectories.

C Experimental Protocol

Pre-registration. Each experiment has one released criterion. The decision experiment's core design (variables, arms, prompts, the frontier runs F1–F4 and the first local runs, read conditions, confidence threshold ($\tau_{\text{valid}} = 0.5$) and outcomes O1–O7) was committed before any model call. Later additions were each committed before the calls they govern: the final label estimator of Equation (1)

(after a three-arm smoke test showed that scoring the exact token path gives $q_{\text{valid}} \approx 10^{-5}$ because of token merging, and before any main run); the rule for provider refusals (kept in the accounting); the second and third open-weight models, the Anthropic evaluators and the disambiguation arms with outcomes O8–O9; and serving and scheduling rules, including a fail-closed bit-identity gate for any restarted server. None changes an existing arm or decision rule. The consequence experiment's criterion was committed before any model call; its pipeline and clock rules took their final form after smoke tests on two cases and before any condition ran (shipped native tool calling, because the prompted adapter's parser ended every episode after one turn; one tool call per turn, which Llama-3.1's chat template requires; the fixed clock; tool-call identifiers, which the chat template does not render, removed before comparing transcripts).

Table 3. Variables of the decision experiment.

unit	evidence packet: policy text, tool schemas, actor-visible messages, one candidate call
presentation	an order for the members of every JSON object; all arms of a packet are equivalent under RFC 8259 and content-identical under the renderer's typed-tree check
factors	top-level member order; nested member order; derived: policy-before-conversation, candidate-first
arms	R, C, K (screening); HRc, HCr (factorial corners); P (all objects random); T (top level random, nested sorted); N (top level sorted, nested random)
outcomes	frontier: parsed decision or failure category; open-weight: p_d, top label, q_{valid}, entropy, and a greedy decision under schema-constrained decoding
estimands	repeat-controlled excess; ALLOW share with Wilson intervals; across-presentation entropy; T vs N Fisher test; FLAGSHIP(t_0; 0.5); control verdict

Accounting. gpt-6-astra: 126 attempts; 93 valid decisions, no invalid outputs, timeouts or isolation failures; 33 refused for the account's spend limit before any model turn (12 N arms, one T arm, all 20 control items). When the limit was lifted, each of the 33 was run once more, as the criterion prescribes: all 33 valid. Disambiguation D1: 62 attempts, 61 valid, one refused when the limit was reached again; D2 was then stopped (two refusals and two calls in flight, recorded). Anthropic evaluators (partial; runs stopped at an account limit): Claude Code 2.1.283, one fresh session per call with no tools except the structured-output tool, no MCP servers, and the caller's `CLAUDE*` environment removed; each model's first call was a single-item harness check. Recorded before the stop: Opus 5.5 74 attempts (control 20, corners 30, D1 24; all valid), plus 52 random orders run later (all valid, all ALLOW), Sonnet 5 65 (control 20, corners 30, D1 15; one invalid output), Haiku 4.5 18 control items (7 invalid outputs: the decision placed in an invented tool call), then 106 fixed- and random-order calls (80 valid, all ALLOW; its orders do not split, so disambiguation was not run); 6 calls in flight at the stop are recorded as aborted. Sonnet 5 at low effort, run afterwards as a separate condition: 126 attempts, 120 valid; control 15/20; all 100 valid order calls ALLOW, so its disambiguation was not run. Open-weight runs: Table 4; no arm failed the tokenization-identity check, and every pre-flight asserted raw log-probabilities and found identical requests identical ($\max|\Delta| = 0$).

Serving and read conditions. vLLM 0.30.0, one sequence at a time, no chunked prefill, eager execution, no prefix caching, raw log-probabilities, bfloat16. Raw-probability mode is asserted behaviorally (identical prompt log-probabilities at temperatures 1.0 and 0.5) and determinism checked (identical requests return identical values, maximum absolute difference 0.0) at the start and end of every run. After a host restart mid-run, completed arms re-ran bit-identically. gemma-3-27b and Qwen3-32B are served in bfloat16 with the weights that exceed the GPU held in host memory. Every restarted or reconfigured server had to reproduce the first five committed control items bit for bit before any of its results was recorded; one restarted gemma server did not, and was not used (it differed from the committed configuration in two settings, so the cause is not identified). The label tokens are single tokens for all three models; the per-prompt tokenization identity is checked on every arm.

Control set. Twenty proposals on the first packet's conversation, extended with an authentication and a profile lookup produced by the benchmark's own tools on its pinned database: ten ALLOW (calculator calls, reads of the authenticated user's own orders, profile and products) and ten BLOCK (cancellations, modifications and reads of other users' orders); two of the reads are shortcut detectors

Table 4. Open-weight runs: arms recorded against the planned inventory. Generated from the run files. The gemma and Qwen screening replications (L6) were still running at submission; they are not analysed, and no result in the paper depends on them.

model	job	recorded	planned	tok. fail	pre-flights (raw mode; max $\|\Delta\|$)
Llama-3.1-8B	L1 (screening, R/C/K)	294	294	0	5, all raw; 0
Llama-3.1-8B	L2 (flipped, all arms)	915	915	0	8, all raw; 0
Llama-3.1-8B	L3 (control)	20	20	0	2, all raw; 0
Llama-3.1-8B	LD1 (disambiguation D1)	57	57	0	3, all raw; 0
Llama-3.1-8B	LD2 (disambiguation D2)	57	57	0	2, all raw; 0
gemma-3-27b	L4 (flipped, corners)	15	15	0	2, all raw; 0
gemma-3-27b	L5 (t_0 random orders)	76	76	0	2, all raw; 0
gemma-3-27b	L6 (screening, R/C/K)	76	165	0	4, all raw; 0 *(incomplete)*
gemma-3-27b	L7 (control)	20	20	0	2, all raw; 0
gemma-3-27b	LD1 (disambiguation D1)	57	57	0	2, all raw; 0
gemma-3-27b	LD2 (disambiguation D2)	57	57	0	2, all raw; 0
Qwen3-32B	L4 (flipped, corners)	15	15	0	2, all raw; 0
Qwen3-32B	L5 (t_0 random orders)	76	76	0	2, all raw; 0
Qwen3-32B	L6 (screening, R/C/K)	20	165	0	2, all raw; 0 *(incomplete)*
Qwen3-32B	L7 (control)	20	20	0	2, all raw; 0
Qwen3-32B	LD1 (disambiguation D1)	57	57	0	3, all raw; 0
Qwen3-32B	LD2 (disambiguation D2)	57	57	0	2, all raw; 0

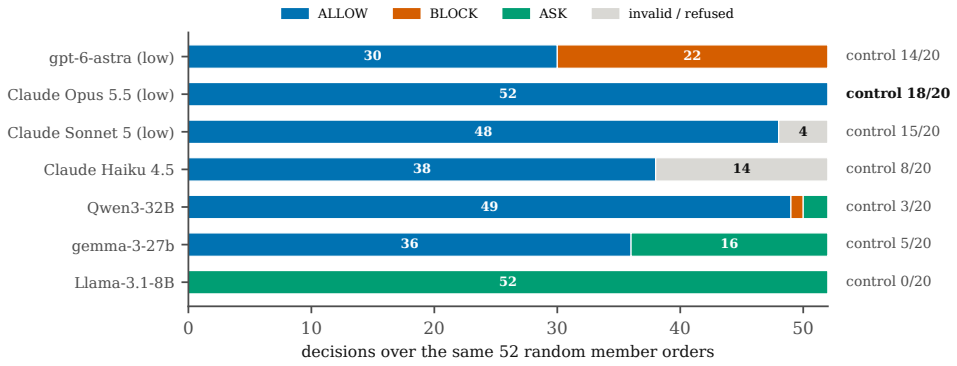


Figure 5. Every evaluator on the same 52 random orders of the original proposal. Bars count decisions (frontier) or top labels (open-weight); right, the competence-control score (pass: at least 18 of 20). Only gpt-6-astra splits between ALLOW and BLOCK; the one evaluator that passes the control, Opus 5.5, approves every order.

whose only difference from an ALLOW item is ownership. The rule (all 20 measured with entropy below 0.1, at least 18 correct, not both detectors missed) was fixed before any call.

Exploratory analyses. The policy-position test and the reason coding in Section 5 were specified after the replication. For the reason coding, the 93 reasons were shuffled with a fixed seed and stripped of arm and packet identity; each received one of three labels (precondition applied and unmet; precondition addressed and judged satisfied; lookup licensed without engaging the precondition), drafted by an AI system with a rationale, reviewed by a human annotator, who kept every label, and re-checked by an AI consistency pass that changed none (Section E). A mechanical keyword rule, written before the labelling, agrees on the BLOCK side (45 of 45) and flags 12 of the 48 ALLOW reasons.

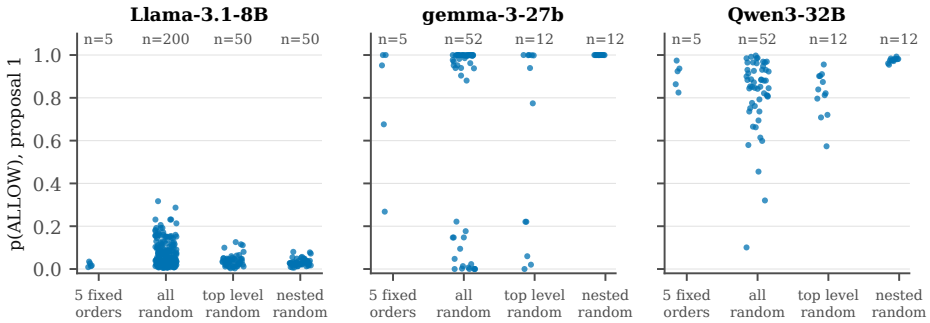


Figure 6. Open-weight models on the first flipped proposal. $p(\text{ALLOW})$ for each order, grouped by which members are reordered (all three models fail the competence control). Llama stays near zero: on each flipped proposal it answers ASK on at least 293 of its 305 orders (5 fixed, 200 all-random, 50 top-level, 50 nested; Llama's larger inventory), and the rest are deterministic flips (to ALLOW on 11 and 12 orders of two proposals, to BLOCK on one order of the first); gemma splits between near-certain ALLOW and ASK, but only when the top level varies, never when only nested members vary; Qwen3-32B is mostly ALLOW.

D Reproducibility

The released repository contains, for every audit, the frozen criterion, the scripts, the committed outputs and the upstream commit hash with a verbatim bundle of the audited source; for every experiment, the pre-registered criterion, the arms and control set with their hashes, every run record, and the analysis code that produces every number in Sections 5 and 6. A single script re-runs every audit and compares the regenerated outputs with the committed ones; on the submitted version all 24 comparisons pass (21 equal; three within a documented tolerance) (one script's own hash after an edit that changed no output; one key added by a later script version; one permutation set regenerated with identical content). A machine-checked verifier tests 75 claims of the formal development, including every numeric constant in Lemma 1 and the renderer-independence property underlying the ST-WebAgentBench certificates. Every model call is logged with its purpose. Frontier results were obtained through a sandboxed command-line transport with per-call validation that no host tool was invoked; they cannot be re-run without the same service, and their full event logs are released.

E Use of AI Systems

This section follows the ICLR 2027 policy on AI use by authors.

What AI systems did. AI coding assistants were used to implement the audit and experiment code, run the audits, compute the statistics, draft the formal development and the text, and maintain the run records. Frontier language models were used as adversarial reviewers of the formal development and of drafts of the paper. The frontier evaluators and the open-weight models of Sections 5 and 6 are the objects of study, not assistants.

Annotation. The annotation labels used in this paper (the coherence judgments behind constructed witnesses, the sampled scored-condition judgments for ST-WebAgentBench, and the coding of the frontier evaluator's 93 stated reasons) are human-annotated with AI assistance. An AI system drafted each label with its rationale; a human annotator reviewed the labels against the source material and made the final decision; an AI system then re-read every item against the label definitions as a consistency check. For the reason coding, the human review kept all 93 drafted labels and the final check changed none (review record released with the labels).

Safeguards. Every number in the paper is produced by released code from released run records, and no number was typed by hand. Pass criteria and experiment designs were committed before the runs they govern. Every reference was checked against its primary source (arXiv record, proceedings page, or the paper's own header), and no reference was added without that check. The authors reviewed all

text, take full responsibility for its content, and verified that it does not reproduce text from other sources.