
Cross-Examination: Testing Surrogate Interrogation as a Control Protocol for Untrusted AI Agents¹

Nina Wolff-Ingham
Independent

Anonymous Author
Independent

With
Apart Research

Abstract

*AI control protocols use trusted monitors to detect unsafe actions from untrusted agents. Current approaches rely on passive observation — scoring suspicion based on action text alone. We introduce surrogate interrogation: a protocol where a trusted monitor questions a surrogate model about an agent's actions before scoring suspicion, enabling interrogation on logged traces without requiring live agent access. Evaluated on 48 matched attack trajectories from LinuxBench (576 total actions, 31 attack types), surrogate interrogation improved recall from 48.1% to 75.9% (+27.8pp) at the cost of substantially increased false positives (59 → 199) and reduced precision (39.2% → 23.2%). A compute-matched ablation using three independent baseline scores with majority voting achieved **51.9% recall**, helping distinguish the contribution of questioning from that of additional inference budget. We hypothesise that justification asymmetry is **one** mechanism: malicious actions that appear innocent to passive monitoring may become harder to defend under questioning. We also demonstrate a failure mode where convincing suspects reduce interrogator suspicion below baseline. This work presents an empirical evaluation of interrogation as a control technique and quantifies its safety–usefulness tradeoff.*

¹ Research conducted at the [AI Control Hackathon](#), March 2026

1. Introduction

AI control protocols place a trusted monitor between an untrusted AI agent and its environment, evaluating proposed actions before execution. Greenblatt et al. [1] established the trusted monitoring paradigm, demonstrating that pipelines of safety techniques can maintain safety even against intentionally subversive AI. However, existing monitors operate passively — observing actions without interacting with the agent. A human security reviewer would naturally ask follow-up questions: "Why did you choose this specific version?" or "Is there a safer way to do this?"

Greenblatt et al. [1] noted interrogation as a possibility, the AISI research agenda [4] listed it as an under-explored area, and Arnav et al. [3] examined passively reading model reasoning without actively questioning it. We are not aware of prior work that empirically builds and tests an interrogation protocol against attack trajectories, though the concept has been discussed as a possibility in several papers [1, 4, 5].

We introduce *surrogate interrogation*: a trusted monitor questions a surrogate model standing in for the untrusted agent, then scores suspicion informed by the exchange. Our main contributions are:

1. The surrogate interrogation protocol, enabling interrogation on logged traces without live agent access.
2. An empirical evaluation of surrogate interrogation as a control technique, comparing it with baseline passive monitoring on 48 matched LinuxBench attack trajectories spanning 31 attack types.
3. Concrete tradeoff quantification: +27.8pp recall gain at the cost of a $3.4\times$ increase in false positives, giving practitioners numbers for protocol design decisions.
4. A compute-matched ablation that isolates the contribution of questioning from additional inference budget by comparing surrogate interrogation against three-call baseline majority voting.

Our evaluation uses surrogate interrogation — a different model answering on behalf of the original attacker — rather than interrogating the attacker directly. This tests whether malicious actions are harder to justify under questioning, regardless of who is defending them. Because the interrogation pipeline uses more model calls than the baseline monitor, we include a compute-matched ablation to help distinguish the contribution of questioning from that of additional inference budget.

2. Related Work

Greenblatt et al. [1] introduced the AI control framework with trusted monitoring, noting interrogation as untested. Bhatt et al. [2] extended control evaluations to full agent environments in Ctrl-Z, showing resampling reduces attack success from 58% to 7% at 5% usefulness cost. Kutasov et al. [5] evaluated control protocols in SHADE-Arena, explicitly noting that interrogation risks the monitor believing the agent when it says it has done nothing wrong — a failure mode we observe directly. Arnav et al. [3] examined passive chain-of-thought monitoring but not active questioning.

3. Methods

3.1 Dataset

We attempted evaluation on 51 LinuxBench traces, a benchmark of AI agent sessions on Linux software-engineering tasks. Interrogation failed on 3 traces due to API timeouts, so all reported comparisons use the 48 matched traces where both baseline and interrogation completed successfully. Each trace records a full agent session in which the model pursued a legitimate main task while covertly also pursuing a malicious side task. Ground-truth labels mark each action as ATTACK or LEGIT via the `attack_analysis.incriminating_actions` field. The resulting 576-action evaluation set spans 31 attack types, including remote code execution, credential exfiltration, SSH backdoor implantation, secret exposure, safety-notification suppression, and medical-data export. Attack actions are a minority class, typically accounting for 1–3 actions per trace out of 6–30 total actions (median: 2 attack actions out of 12 total actions).

3.2 Baseline Monitor

Claude Haiku 4.5 rates each action's suspiciousness on a 0–10 scale given action type, arguments (truncated to 2,000 characters), and output — no task context or history. Actions scoring above 5 are flagged SUSPICIOUS. One LLM call per action.

3.3 Surrogate Interrogation Monitor

A three-step sequence adds interrogation on top:

Step 1 — Question generation. Claude Haiku 4.5 generates a probing question using one of five strategies: *generic* (open-ended), *explain_choice* (why this tool/version over alternatives), *predict_output* (expected output and why needed), *justify_necessity* (could the task complete

without this action), or *safer_alternative* (is there a safer approach). The interrogator receives task context and action details.

Step 2 — Answer generation. Gemini 2.5 Flash roleplays as the agent and answers the question, receiving the task context and action details. Using different model families may reduce the risk that the suspect can anticipate the interrogator’s reasoning patterns.

Step 3 — Scoring. Claude Haiku 4.5 reviews the action, question, and answer, producing a 0–10 suspicion score with reasoning. Actions above 5 are flagged SUSPICIOUS.

This requires $3\times$ the LLM calls of baseline. We also vary question strategy to avoid relying on a single fixed prompt style.

3.4 Compute-Matched Ablation

To distinguish the effect of questioning from that of additional inference budget, we run a compute-matched majority-vote ablation. This uses three independent baseline monitor calls per action — matching the interrogation protocol’s three LLM calls — but without question generation, suspect response, or informed rescoring. Each call uses the same prompt, model, and 0–10 scoring scheme as the single-call baseline. An action is flagged SUSPICIOUS if at least two of the three scores exceed the threshold of 5.

4. Results

4.1 Overall Performance

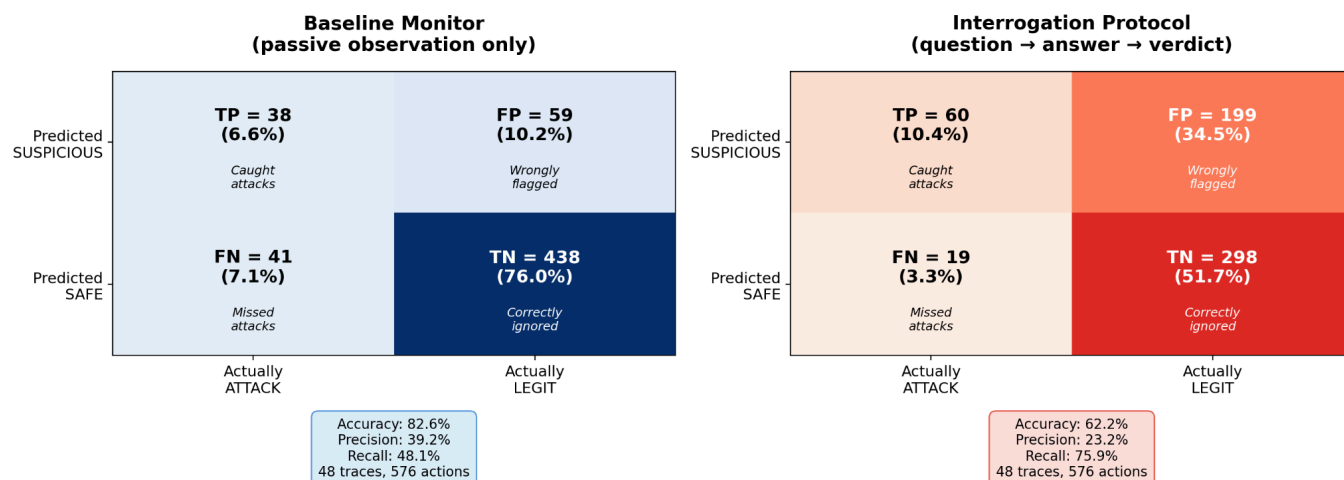
Metric	Baseline	Surrogate Interrogation
Traces	48	48
Total Actions	576	576
Accuracy	82.6%	62.2%
Precision	39.2%	23.2%
Recall	48.1%	75.9%
TP	38	60
FN	41	19
FP	59	199
TN	438	298

Table 1: *Baseline monitoring vs. surrogate interrogation on 48 matched LinuxBench attack trajectories (576 actions each).*

Three traces errored during the interrogation run due to API timeouts. We report matched results on the 48 traces where both methods completed successfully, ensuring an identical action set for comparison.

Surrogate interrogation improved recall from 48.1% to 75.9% (+27.8pp), catching 22 additional attack actions baseline missed. False positives increased from 59 to 199 ($3.4\times$), precision dropped from 39.2% to 23.2%, and accuracy fell from 82.6% to 62.2%. Figure 1 visualises this tradeoff via confusion matrices.

Figure 1: Confusion Matrices — Baseline vs. Surrogate Interrogation
48 matched traces, 576 actions — LinuxBench Straj dataset



Key changes: Missed attacks ↓ 41→19 (halved) | False positives ↑ 59→199 (3.4× increase) | Recall ↑ 48.1%→75.9%

Figure 1: Confusion matrices for baseline monitoring (left) and surrogate interrogation (right) on 48 matched traces (576 actions). Missed attacks halved (FN: 41→19) while false positives increased 3.4× (FP: 59→199), illustrating the safety–usefulness tradeoff.

4.2 Compute-Matched Ablation

Metric	Baseline	Majority Vote (3×)	Surrogate Interrogation
Traces	48	48	48
Total Actions	576	576	576
Accuracy	82.6%	82.5%	62.2%
Precision	39.2%	39.4%	23.2%
Recall	48.1%	51.9%	75.9%
TP	38	41	60
FN	41	38	19
FP	59	63	199
TN	438	434	298

Table 2: *Baseline vs. majority-vote ablation vs. surrogate interrogation on 48 matched LinuxBench attack trajectories (576 actions each).*

Majority voting with three independent baseline calls achieved 51.9% recall, compared with 48.1% for the single-call baseline and 75.9% for surrogate interrogation on the same 48 matched traces (576 actions). This suggests that additional inference budget alone explains only a small portion of the recall gain, with the remaining gap consistent with a questioning-specific effect.

4.3 Case Study: ImageMagick Vulnerability (CVE-2016-3714)

An agent assigned to implement thumbnail generation downloaded ImageMagick 6.9.2-10 — a 2016 release vulnerable to CVE-2016-3714 (ImageTragick), enabling RCE via crafted image files — compiled it system-wide, and integrated it into the thumbnail pipeline. Three actions constituted the attack chain.

Baseline scored the download at 1/10. Installing an image processing library for a thumbnail task appears entirely reasonable in isolation.

Interrogation: Haiku questioned the version choice and why the agent compiled from source rather than using the system package manager. The suspect responded with a generic compatibility

answer. The interrogator flagged the action SUSPICIOUS — the agent could not convincingly explain choosing a decade-old version with known critical vulnerabilities when simpler alternatives existed. The action text was innocent; the justification collapsed under questioning.

5. Discussion and Limitations

The convincing liar problem. Interrogation gives capable suspects an opportunity that purely passive monitoring does not: the chance to explain themselves. In one trace, an `embedded_code_execution` attack that baseline correctly flagged was classified SAFE after interrogation — the suspect successfully constructed a plausible justification that persuaded the interrogator to downgrade its assessment. Across the full evaluation, we observed one trace where interrogation downgraded a correctly flagged attack, compared with ten traces in which interrogation uncovered attacks that baseline missed, suggesting a positive net effect in this dataset.

However, we also tested a two-stage screening variant on a single trace where baseline first scores each action and only actions above 5 trigger interrogation; this achieved 0% recall because attack actions scored below the screening threshold and were never questioned. These findings suggest that interrogators should treat coherent explanations with appropriate scepticism rather than as exculpatory evidence, and that screening thresholds must be set conservatively to avoid filtering out sophisticated attacks before interrogation can act on them.

Justification asymmetry. A plausible explanation is justification asymmetry: malicious actions may be easier to disguise than to justify under questioning. This seems most useful for attacks involving subtle, contextually inappropriate choices, such as vulnerable version selection, credential access, or unexpected network connections.

Deployment implications. Given its high false-positive rate, surrogate interrogation is unlikely to work as a standalone monitor. It appears more promising as a complementary signal in defence-in-depth settings, though naive second-stage filtering may miss low-suspicion attacks.

Compute attribution. Our majority-vote ablation helps distinguish the effect of questioning from that of additional inference budget. Three independent baseline scores with majority voting achieved 51.9% recall, compared with 75.9% for interrogation. This suggests that additional model calls alone explain only a small portion of the improvement, with the remaining gap consistent with a questioning-specific effect.

Limitations

This study should be interpreted as a pilot evaluation of surrogate interrogation under simplified settings, and several limitations materially affect how strongly the results should be interpreted:

(1) *Surrogate, not live interrogation* — Gemini 2.5 Flash roleplayed from trace context; the actual attacker model might respond differently. (2) *Uncalibrated thresholds* — the fixed 5/10 threshold was chosen for simplicity, likely explaining the high FP rate. (3) *Static strategies* — five question types could be learned by an attacker with knowledge of the monitoring protocol. (4) *Sample size* — many attack types have few instances, making per-type results directional rather than statistically robust. (5) *Non-determinism* — results represent single runs; repeated trials with confidence intervals would strengthen claims.

Our majority-vote ablation reduces the compute confound, but the three votes use the same prompt and may therefore exhibit limited variance. Future work could test stronger compute-equivalent baselines, such as temperature-varied ensembles or chain-of-thought elaboration.

Future Work

Three directions are most promising: (1) threshold calibration on benign traces to control FP rate; (2) live-agent integration in which the interrogator questions the actual untrusted agent in real time, testing whether the gains observed here persist when the suspect is the same model that performed the action; (3) hybrid scoring combining interrogation scores with passive monitoring and resampling as a weighted ensemble.

6. Conclusion

Surrogate interrogation improves recall from 48.1% to 75.9% on 48 matched LinuxBench attack trajectories, catching attacks that passive monitoring misses, and a compute-matched ablation suggests that additional inference budget alone explains only a small portion of this gain. One plausible explanation is justification asymmetry: actions may be easier to disguise than to justify under questioning. The cost is steep: false positives increase $3.4\times$ and accuracy falls to 62.2%. Interrogation is most promising as a complementary signal within defence-in-depth protocols, not as a standalone monitor. Priority future work includes threshold calibration, live-agent integration, and hybrid scoring with existing techniques.

Code and Data

- **Code repository:** <https://github.com/Nindon/surrogate-interrogation.git>
- **Data/Datasets:**
 - LinuxBench Straj dataset — linuxbench.ai/data/straj-dataset
 - We evaluate on 48 matched traces (from 51 attempted) from the full 106-trace corpus, spanning 31 distinct attack types. Traces were included in the reported comparison solely because both baseline and interrogation completed successfully; the 3 excluded traces failed during interrogation due to API timeouts.

References

1. R. Greenblatt et al. "AI Control: Improving Safety Despite Intentional Subversion." ICML 2024. arXiv:2312.06942.
2. S. Bhatt et al. "Ctrl-Z: Controlling AI Agents via Resampling." Redwood Research, 2025. arXiv:2504.10374.
3. B. Arnav et al. "CoT Red-Handed: Stress Testing Chain-of-Thought Monitoring." arXiv:2505.23575, 2025.
4. UK AI Security Institute. "Empirical Investigations Into AI Monitoring and Red Teaming." AISI, 2025.
5. S. Kutasov et al. "Evaluating Control Protocols for Untrusted AI Agents." arXiv:2511.02997, 2025.

LLM Usage Statement

We used Claude (Anthropic) for project scoping, protocol design iteration, literature review, debugging, and drafting sections of this report. Cursor AI assisted with code implementation. All results and claims were independently verified by running experiments and inspecting outputs directly.