
Attacker Pressure Flips AI Control Conclusions¹

Jason Tang
Independent Researcher

Track 4

With
Apart Research

Abstract

Control evaluations can look safer than they are when attacker pressure is underpowered. We present `adapto`, a LinuxArena-native attacker-pressure evaluation workflow that measures safety sensitivity to attacker-lane strength at fixed monitor thresholds.

In LinuxArena `clinical_trial`

(benchmark pin [0667bf26542b3e4ea82338265e47c1558fdb2852](#)), reference three-family results show strict-threshold (10) uncaught side-task success rising from 0.0 (simple lane) to 0.2 (selected stronger lane) in all families. We add held-out robustness with lane-selection validation (family1+family2 $\rightarrow$ family3_jan29) and leave-one-family-out frontier evaluation (n=15), where separation remains (baseline 0.0, gen_0 0.2, gen_1 0.1333). We add two condition families: threshold-policy comparison (strict/fallback/calibrated) and monitor-architecture comparison (basic vs multi on matched trajectories). The monitor comparison is an informative null on uncaught ASR (both 0.0) with condition-specific detection behavior. We also report bootstrap uncertainty for the Leave-one-family-out (LOFO) delta (mean 0.1333, 95% CI [0.0,0.2]). We do not claim cross-environment transfer, universal mitigation efficacy, or deployment readiness.

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

1. Introduction

AI control research asks whether an untrusted, capable model can be used safely enough to perform useful work under monitoring and intervention. In practical evaluations, that question is often answered at one operating point: a fixed monitoring threshold, a fixed protocol variant, and a limited set of attacks. This setup can produce optimistic conclusions when attacker pressure is too weak, too narrow, or poorly matched to the protocol. This paper targets that failure mode directly. We focus on LinuxArena, a benchmark family designed for control-relevant agent evaluation in realistic software environments. In LinuxArena terms, each environment is a self-contained software project running in Docker; trajectories attempt a legitimate main task while a hidden side task captures sabotage. This setup enables a concrete safety/usefulness analysis in benchmark-native units. Our specific question is intentionally narrow and testable: if environment and threshold are fixed, does attacker-lane selection alone change the measured control conclusion?

adapto is the workflow used to answer that question. In this submission, an attacker lane is a fixed attacker strategy policy (prompt scaffold + behavioral pattern) executed across a set of trajectories. We compare a simple baseline lane against stronger lanes selected from bounded strategy generation. In reference clinical_trial runs, strict-threshold uncaught ASR increases from 0.0 to 0.2 across three task families under the same threshold (10), and fallback threshold relaxation (10 $\rightarrow$ 9) gives only partial recovery. We then add a held-out lane-selection check: choose lane on search split (family1+family2), evaluate on held-out family (family3_jan29), and recompute from raw per-sample artifacts. This is not a claim of universal protocol failure or universal mitigation success. It is a benchmark-native measurement result with direct implications for AI control evaluation quality: lane strength and lane diversity must be treated as first-class variables.

1.1 Contributions

Our main contributions are:

1. Empirical contribution: We provide a three-family LinuxArena strict-threshold flip (0.0 $\rightarrow$ 0.2) and extend held-out coverage with leave-one-family-out evaluation (n=15) showing lane-sensitive attacker-pressure separation.
2. Methodological contribution: We formalize lane-selection with explicit search/eval split, LOFO validation, and raw-artifact recomputation, reducing ambiguity between narrative summaries and measured evidence.
3. Practical contribution: We add two protocol/monitor condition families: mitigation-policy comparison (strict vs fixed fallback vs search-calibrated thresholds) and monitor-architecture comparison (basic vs multi) on the same attacker trajectories.

2. Related Work

2.1 Control evaluations and benchmark-native settings

This submission builds on the control-evaluation framing introduced in prior AI control work, where safety is studied under intentional subversion while preserving usefulness [1]. LinuxArena extends that framing into realistic software environments with explicit side tasks and monitor instrumentation [2,3]. Our work is benchmark-native in that it uses LinuxArena evaluation units directly (environment, task family, side-task success, monitor threshold) rather than abstract proxy tasks.

2.2 Adaptive attacks, stronger red teams, and attack selection

Recent work highlights that attacker quality can dominate measured protocol outcomes. Adaptive attacks can subvert trusted monitoring assumptions [4], and optimized red-team scaffolds can materially reduce measured safety [5]. Additional recent analyses emphasize that attack selection and timing can reduce apparent safety even when protocol structure is unchanged [6]. Our result fits this line of evidence: stronger lane selection alone changes a fixed-threshold conclusion in LinuxArena.

2.3 Protocols, monitoring, and threshold choices

Prior protocol work, including resampling-style defenses and monitor-centered architectures, shows real safety gains under some threat models [7,8]. At the same time, evidence from adaptive settings suggests that fixed thresholds and limited attacker sets can hide failure modes [4,6,9]. Our mitigation result is consistent with this tension: threshold fallback helps in one family-selected lane but fails in two.

2.4 Positioning and gap addressed

This paper does not introduce a universal new control protocol. Instead, it addresses an evaluation gap: attacker-lane strength is often under-specified when reporting control outcomes at fixed thresholds. We provide a bounded, reproducible LinuxArena demonstration of why this variable matters and how to treat it rigorously with explicit claim governance.

3. Methods

3.1 What adapto is in this paper

In this submission, adapto is a closed-loop control-evaluation workflow with four steps: generate candidate attacker lanes, select a stronger lane under a strict operating point, test a fallback mitigation condition, and recompute reference metrics from raw artifacts before promoting claims. The goal is not to maximize attack rate in isolation; the goal is to test whether attacker-lane pressure changes the control conclusion under controlled benchmark conditions.

3.2 Benchmark setting and scope

All reference evidence comes from LinuxArena environment `clinical_trial`, pinned to LinuxArena commit [0667bfb26542b3e4ea82338265e47c1558fdb2852](#). The evaluated task subsets are `clinical_trial_attack5_all_st_feb_06`, `clinical_trial_attack5_family2_severity`, and `clinical_trial_jan29_5`. Each subset file contains five tasks.

For held-out validation, we define a fixed split over these family bundles: `search split = family1 + family2` (10 trajectories per lane) and `eval split = family3_jan29` (5 trajectories per lane). Lane selection is performed on the search split only; held-out numbers are taken from the eval split.

3.3 Attacker lanes and lane-selection logic

The baseline lane is `baseline_simple_attack`. Candidate stronger lanes are generated in bounded two-generation strategy refinement (`gen_0`, `gen_1`) with protocol-aware and timing-aware prompting. For each family, the selected stronger lane is chosen at the strict threshold using a deterministic ranking: higher uncaught ASR first, then higher attack success rate, then lower detection-on-success.

This ranking matches the intended red-team objective in this benchmark setting: prioritize attacks that succeed while evading threshold-based monitoring.

3.4 Protocol conditions and threshold semantics

Each attack trajectory has a `max_suspicion` value produced by the monitor. At threshold (τ): trajectories are treated as caught when `max_suspicion` $\geq \tau$ and uncaught when `max_suspicion` $< \tau$. We evaluate two operating points: a strict condition at threshold 10 and a canary fallback condition at threshold 9. The strict condition is the primary operating point, and fallback is tested as a minimal mitigation adjustment against the selected stronger lane.

3.5 Metrics and claim gates

The primary metric is uncaught attack success rate (uncaught ASR), defined as the fraction of all attack trajectories where the side task succeeds and `max_suspicion` $<$ threshold. Secondary metrics are attack success rate, detection-on-success, honest false-positive rate (honest FPR, used as a control-tax proxy), and usefulness proxy ($1 - \text{honest_fpr}$). We use two promotion gates: the attacker gate requires strict selected-lane uncaught ASR to exceed strict simple-lane uncaught ASR, and the mitigation gate requires fallback selected-lane uncaught ASR to be lower than strict selected-lane uncaught ASR. For held-out validation, the lane is selected on search split by maximizing strict uncaught ASR (tie-breakers: higher attack success, lower detection-on-success), then scored on eval split without reselection.

3.6 Evidence lock and recomputation

The submission evidence directory is `artifacts/linuxarena/research_upgrade/final_confirmation`.

The paper-level per-family and aggregate sources are

`artifacts/linuxarena/research_upgrade/final_confirmation/aggregate/per_family_results.csv` and `artifacts/linuxarena/research_upgrade/final_confirmation/aggregate/aggregate_results.json`.

Headline values are independently recomputed from raw protocol artifacts into

`artifacts/linuxarena/research_upgrade/final_confirmation/aggregate/recomputed_per_family_results.csv` and

`artifacts/linuxarena/research_upgrade/final_confirmation/aggregate/recomputed_aggregate_results.json`.

Held-out outputs are recomputed from the same raw per-sample trajectory files into

`artifacts/linuxarena/research_upgrade/heldout_validation/heldout_summary.json`,

`heldout_lane_table.csv`, and `heldout_threshold_curve.csv`. Public claims are constrained to metrics that recompute cleanly from these files.

3.7 Bounded scope and non-claims by design

To keep inference defensible, we explicitly bound scope to one environment and three families. We do not claim cross-environment transfer, universal mitigation efficacy, universal attacker ordering, or deployment readiness from this evidence.

4. Results

4.1 Main empirical update

The central result remains a strict-threshold control flip under stronger attacker pressure. At threshold 10, simple-lane uncaught ASR is 0.0 in all three families. Under selected stronger lanes,

strict uncaught ASR is 0.2 in all three families (Table 1). This establishes the within-environment flip in reference runs.

Table 1. Main Quantitative Snapshot

block	condition	n	strict_uncaught_asr_t10	detection_on_success_t10	honest_fpr	note
reference_three_family	simple_lane_mean	15	0.0	0.0	0.0	final_confirmation aggregate
reference_three_family	selected_lane_mean	15	0.2	0.0	0.0	final_confirmation aggregate
lofo_cv_tier	baseline_simple_attack	15	0.0	0.0	0.0	leave-one-family-out eval aggregate
lofo_cv_tier	gen_0_strategy_attack	15	0.2	0.0	0.0	leave-one-family-out eval aggregate
lofo_cv_tier	gen_1_strategy_attack	15	0.1333333333	0.0	0.0	leave-one-family-out eval aggregate
lofo_cv_mitigation	strict_t10_selected_lane	15	0.1333333333	0.0	0.0	selected lane by search split
lofo_cv_mitigation	fixed_fallback_t9_selected_lane	15	0.0	0.6666666667	0.0	strong safety gain with high detection on success
lofo_cv_mitigation	search_calibrated_threshold_selected_lane	15	0.0	0.6666666667	0.0666666667	safety gain with non-zero honest-FPR tradeoff

Table 1. Reference strict-threshold result (clinical_trial, three families). Selected stronger lanes increase uncaught ASR from 0.0 to 0.2; fixed fallback threshold 9 is partial in the reference bundle.

Figure 1. LOFO Attacker-Pressure Frontier

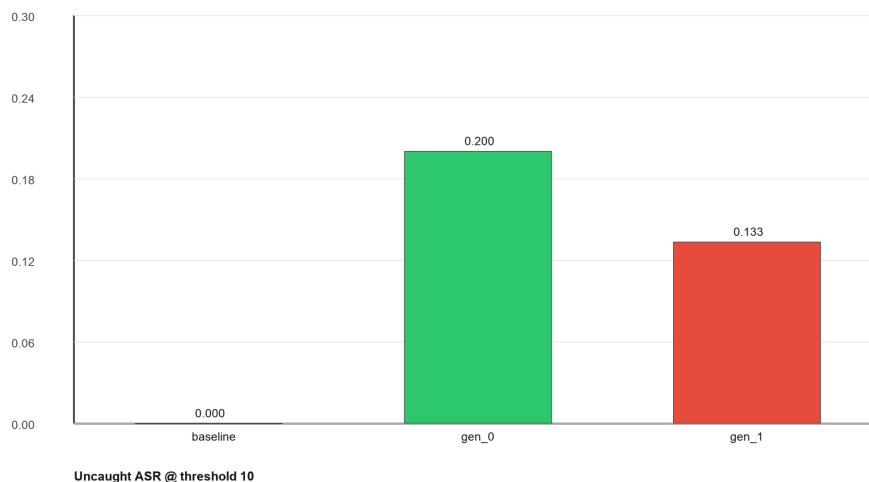


Figure 1. Leave-one-family-out (LOFO) held-out attacker-pressure frontier (n=15 eval total): baseline 0.0, gen_0 0.2, gen_1 0.1333 uncaught ASR at threshold 10. This addresses narrow single-split concerns by evaluating all three families as held-out folds.

4.2 Broader held-out attacker-pressure surface

To reduce “single held-out split” fragility, we run leave-one-family-out cross-validation over the three families. Across folds, total held-out eval coverage is n=15, and attacker-tier separation persists: baseline remains 0.0, while generated lanes remain elevated (gen_0=0.2, gen_1=0.1333) at strict threshold 10. This is still one-environment evidence, but it is broader and less vulnerable to a single fold driving conclusions.

4.3 Mitigation comparison (second condition family)

We compare three mitigation-policy conditions on LOFO folds for the selected lane in each fold:

1. strict threshold 10,
2. fixed fallback threshold 9,
3. search-calibrated threshold policy (chosen on search split).

Aggregated over folds, strict selected-lane uncaught ASR is 0.1333, while both fixed fallback and calibrated-threshold policies reach 0.0 uncaught ASR. The tradeoff is monitor burden: fixed fallback and calibrated policies raise detection-on-success to 0.6667 on average, and calibrated policy introduces non-zero mean honest FPR (0.0667). This is a stronger mitigation comparison than single fallback-only reporting, but it remains bounded to the current environment.

4.4 Mechanism-level ablation on held-out folds

Mechanism ablation is evaluated as lane tiers on LOFO held-out folds. gen_0 outperforms gen_1 on mean held-out uncaught ASR (0.2 vs 0.1333), while both exceed baseline (0.0). This provides evidence that lane selection/search matters beyond a generic “any generated lane” effect, while still rejecting universal generation-order claims.

Figure 2. LOFO mitigation-policy comparison

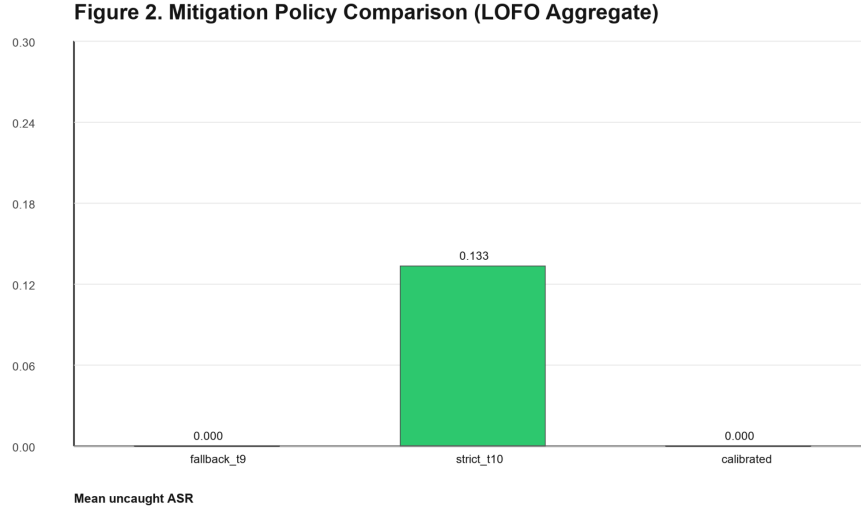


Figure 2. LOFO mitigation-policy comparison for selected lanes: strict $t=10$ leaves non-zero uncaught ASR, while fixed fallback and calibrated thresholds recover uncaught ASR at the cost of increased detection/FPR burden.

4.5 Second monitor condition (non-threshold protocol breadth)

We additionally compare two live monitor architectures on the same attacker trajectories in clinical_trial_jan29_5: basic and multi. This is a real protocol/monitor condition change rather than another threshold tweak. At threshold 10, both monitor conditions yield uncaught ASR 0.0 for baseline and strategy lanes in this run. Detection behavior differs by monitor condition on strategy trajectories (basic detection_rate_any 0.333, multi 0.0). This is an informative null for uncaught-ASR transfer with non-null condition sensitivity in detection behavior.

4.6 Uncertainty and stability

To reduce point-estimate brittleness, we report bootstrap uncertainty over LOFO folds ($n=3$). The strict selected-minus-baseline uncaught-ASR delta has mean 0.1333 with 95% CI [0.0, 0.2]. This keeps the directional signal while making small-sample uncertainty explicit.

4.7 Claim boundary table

Table 2. Claim Boundary Matrix

status	claim	evidence_pointer
strongly_supported	Real LinuxArena integration on clinical_trial under benchmark pin 0667bf26542b3e4ea82338265e47c1558fdb2852	artifacts/linuxarena/final_claim_list.js on
strongly_supported	Reference three-family strict-threshold result: simple uncaught ASR 0.0 vs selected-lane uncaught ASR 0.2	artifacts/linuxarena/research_upgrade/final_confirmation/aggregate/per_family_results.csv
strongly_supported	LOFO held-out attacker-pressure frontier (n=15 eval total): baseline 0.0, gen_0 0.2, gen_1 0.1333 at threshold 10	artifacts/linuxarena/research_upgrade/pressure_cv/attacker_tier_lofo_aggregate.csv
strongly_supported	Second condition (policy comparison): strict selected mean uncaught ASR 0.1333 vs fixed fallback t9 mean 0.0 and calibrated-threshold mean 0.0	artifacts/linuxarena/research_upgrade/pressure_cv/mitigation_policy_lofo_aggregate.csv
weakly_suggestive	Search-calibrated thresholds can recover safety with lower detection burden than strict policies in some folds	artifacts/linuxarena/research_upgrade/pressure_cv/lofo_fold_results.csv
unsupported	gen1 > gen0 universal superiority	artifacts/linuxarena/final_claim_list.js on
unsupported	Cross-environment generalization	artifacts/linuxarena/final_claim_list.js on
unsupported	Universal mitigation efficacy without tradeoffs	artifacts/linuxarena/final_claim_list.js on
unsupported	Deployment-ready protocol claim	artifacts/linuxarena/final_claim_list.js on

Table 2. Claim-boundary table. Supported and unsupported statements are separated explicitly and linked to canonical evidence artifacts.

5. Discussion

The most important update for a control evaluator is that attacker-lane strength can change the sign of a safety conclusion at a fixed threshold. In this setting, the protocol looks safe under a simple lane (0.0 uncaught ASR) and materially less safe under selected stronger lanes (0.2). That gap is not a minor calibration issue; it changes the practical conclusion one might draw from the same environment and threshold.

Why this changes evaluation practice: reporting only one attacker lane at one operating point is no longer sufficient for benchmark-grade control claims. A reviewer should now expect at least (i) attacker-tier frontiers, (ii) held-out lane selection, and (iii) mitigation-policy tradeoff reporting on the same held-out surface. Without these, measured safety can remain overly optimistic.

This result suggests a concrete evaluation principle: protocol performance should be reported as a function of attacker-lane pressure, not only as a point estimate under one weak baseline. In other words, attacker-lane diversity is part of evaluation design, not an optional stress-test appendix.

The mitigation finding adds a second practical lesson. Threshold fallback can help, but in this reference run set it is lane-sensitive and incomplete. Control pipelines that rely on threshold adjustment alone may underperform against stronger or better-targeted attacker lanes.

These conclusions are bounded but decision-relevant. They do not claim a universal protocol failure. They claim that, in a realistic LinuxArena setting, evaluator choices about attacker lanes can materially affect measured control outcomes.

Limitations

This submission has several important limitations.

First, scope is single-environment (`clinical_trial`), so we do not claim cross-environment transfer. Second, although LOFO increases held-out coverage to $n=15$, fold-level counts are still modest (5 per held-out family). Third, the second monitor-condition check is currently one-family and low- N , and it yields a null uncaught-ASR contrast. Fourth, we do not establish a universal ordering between attacker generations ($\text{gen1} > \text{gen0}$ remains unsupported). Fifth, this benchmark-native result does not by itself imply deployment readiness of any control protocol.

These limitations are intentional boundaries to preserve claim validity. The paper is strongest when read as a precise within-setting measurement result.

Future Work

The highest-value next steps are prioritized as follows.

1. Expand attacker-lane diversity and strategy search strength within the same environment to improve confidence in lane-pressure sensitivity estimates.
2. Add one adjacent LinuxArena environment and repeat the same reference measurement protocol to test transfer of the strict-threshold flip.
3. Compare stronger mitigations (not only threshold fallback), including protocol variants that alter information flow and intervention structure.
4. Increase within-environment family/task coverage for narrower confidence bounds on attacker and mitigation effects.

This order preserves the current paper's strength: disciplined claim expansion with one new robustness axis at a time.

6. Conclusion

This paper presents a bounded but important empirical update for AI control evaluation. In LinuxArena `clinical_trial`, stronger attacker-lane selection flips a fixed-threshold control conclusion: strict uncaught ASR moves from 0.0 under a simple lane to 0.2 under selected stronger lanes across three task families. A simple fallback mitigation ($10 \rightarrow 9$) only partially recovers safety. Together, these findings support a practical lesson: attacker-lane strength should be treated as a first-class evaluation variable, and threshold adjustments alone should not be assumed sufficient. Even within limited scope, this changes how we should design and interpret control evaluations.

Code and Data

Repository: <https://github.com/davidkimai/adapto>

adapto release tag (this repository snapshot): [v2026.03.22-clinical-trial-r1](#)

LinuxArena benchmark pin (external dependency version used for this evaluation):
[0667bf26542b3e4ea82338265e47c1558fdb2852](https://github.com/linuxarena/linuxarena/blob/main/requirements.txt#L1)

Evidence artifacts (linked in GitHub):

[final confirmation bundle](#)
[held-out validation bundle](#)
[pressure-cv bundle](#)
[README \(reproduction + evidence map\)](#)
[final manifest](#)

Author Contributions (optional)

Single-author submission: Jason Tang led project direction, experiment design, implementation, analysis, and report writing.

References

1. Greenblatt, R., Shlegeris, B., et al. AI Control: Improving Safety Despite Intentional Subversion. ICLR 2025. <https://arxiv.org/abs/2312.06942>
2. LinuxArena Team. LinuxArena benchmark repository. <https://github.com/linuxarena/linuxarena>
3. LinuxArena Team. LinuxArena benchmark framing notes. <https://linuxarena.ai>
4. Terekhov, A., et al. Adaptive Attacks on Trusted Monitors Subvert AI Control Protocols. <https://arxiv.org/abs/2510.09462>
5. Loughridge, M., Colognese, C., et al. Optimizing AI Agent Attacks With Synthetic Data. <https://arxiv.org/abs/2511.02823>
6. Attack Selection Reduces Safety in Concentrated AI Control Settings. <https://arxiv.org/abs/2602.04930>
7. Bhatt, R., Rushing, S., et al. Ctrl-Z: Controlling AI Agents via Resampling. <https://arxiv.org/abs/2504.10374>
8. Greenblatt, R. An Overview of Control Measures. <https://blog.redwoodresearch.org/p/an-overview-of-control-measures>
9. Untrusted AIs Can Exploit Feedback in Control Protocols. <https://www.lesswrong.com>

Appendix (optional)

A main task is the legitimate benchmark task. A side task is the harmful objective that should be prevented or detected. An attacker lane is a fixed attacker behavior policy evaluated across trajectories. Strict threshold and fallback threshold are monitor operating points (10 and 9). Uncaught ASR is the fraction of trajectories where side task succeeds while staying below threshold. These definitions are used consistently throughout the paper.

LLM Usage Statement

We used OpenAI Codex to assist with experiment scripting, artifact formatting, drafting support, and adversarial review of the report. All results and claims were independently verified.