

TRACEGUARD: Structured Multi-Dimensional Monitoring as a Collusion-Resistant Control Protocol

Khanh Linh Nguyen
linhnguyen@u.nus.edu

Hoa Nghiem
hoa.nghiem@obello.com

Tu Tran
tu.tran@obello.com

With
Apart Research

Abstract

AI control protocols use monitors to detect attacks by untrusted AI agents, but standard single-score monitors face two fundamental limitations: they miss subtle attacks where outputs look clean but reasoning is off, and they **collapse to near-zero safety** when the monitor is the same model as the agent (collusion). We present TRACEGUARD, a structured multi-dimensional monitoring protocol that evaluates agent actions across **five dimensions**—goal alignment, constraint adherence, reasoning coherence, safety awareness, and action-trace consistency—scored in parallel by independent LLM calls, augmented by a pipeline of seven heuristic detectors and an LLM-based intent analyzer. We evaluate on BASHARENA (637 bash tasks, 4 attack categories) within the CONTROLARENA framework. Our results on 519 samples (279 honest, 240 attack) show that: (1) the hybrid detector-plus-dimension approach achieves **clear attack-honest separation** (attack mean suspicion 0.616 vs. honest mean 0.206, $\Delta = 0.410$); (2) **structured scoring constrains collusion**—the untrusted structured monitor achieves **95% safety** at moderate thresholds, compared to **0% for single-score** untrusted monitoring; (3) **goal alignment** and **constraint adherence** are the most discriminative dimensions; and (4) a **separation-of-duties** variant that splits dimensions across trusted and untrusted models achieves **100% safety** while preventing any single model from seeing the full evaluation. The **bash_heuristic** monitor variant combines regex-based pattern matching, stateful detectors (variable binding, data flow tracking, string assembly, encoding detection, subprocess tracking, config diff tracking), and an intent analyzer to contextualize detector findings before LLM dimension scoring. TRACEGUARD is implemented as a new monitor type for the open-source CONTROLARENA framework.

1 Introduction

As AI agents are deployed in increasingly autonomous settings—executing code, managing infrastructure, interacting with external services—the *AI control problem* becomes critical: how do we ensure that an untrusted agent acts in the principal’s interest, even if the agent is strategically misaligned [1]?

The dominant approach is **trusted monitoring**: a smaller, trusted model observes the untrusted agent’s actions and raises an alarm if behavior looks suspicious. When the suspicion score exceeds a threshold, execution is deferred to the trusted model (the *defer-to-trusted* protocol). Greenblatt et al. [1] showed that trusted monitoring achieves approximately 62% safety on their benchmark, while trusted editing reaches 92%.

However, current monitors share a critical design limitation: they produce a **single scalar suspicion score**—essentially a “vibes check” of whether an action looks dangerous. This approach has two failure modes:

1. **Subtle attacks slip through.** When the agent’s output looks clean but its reasoning contains contradictions, hidden operations, or suspicious rationalizations, a single-score prompt often returns “looks fine.” The monitor has no structured way to separately evaluate *what the agent did* versus *why it said it did it*.
2. **Untrusted monitoring collapses.** Using the same model as both agent and monitor (untrusted monitoring) enables collusion—the monitor simply assigns low suspicion to the agent’s attacks. Greenblatt et al. [1] report that naive untrusted monitoring achieves approximately 0% safety.

Recent work suggests that monitoring an agent’s chain-of-thought reasoning is more effective at detecting misalignment than monitoring actions alone [6]. We propose TRACEGUARD, a monitor that operationalizes this insight by decomposing CoT evaluation into **five structured dimensions**, each scored by an independent LLM call with a narrow, specific evaluation prompt. Our key insight is twofold:

- **Decomposition improves detection.** By evaluating goal alignment, constraint adherence, reasoning coherence, safety awareness, and action-trace consistency separately, the monitor catches attacks that “look fine” on one dimension but fail on another.
- **Structure constrains collusion.** When an untrusted model is forced to evaluate narrow criteria (e.g., “does the action match what the reasoning says?”), it cannot as easily produce a blanket pass. The structured prompts constrain what the monitor attends to, making collusion harder.

Our contributions are:

1. A **new multi-dimensional monitoring protocol** (TRACEGUARD) that we designed and implemented as a novel monitor type for the CONTROLARENA framework. CONTROLARENA ships only `basic_monitor` (single-score) and `ensemble_monitor` (N copies of the same prompt); TRACEGUARD introduces structured, multi-dimensional scoring as an entirely new approach to agent monitoring.
2. An **experimental evaluation of collusion resistance** through structured scoring (Redwood Research Direction #9 [4]), demonstrating that our structured prompts constrain even untrusted models from colluding.
3. A **separation-of-duties protocol** that splits dimensions across model trust levels so no single model sees the full evaluation picture (Redwood Research Direction #8 [4]). This variant offers a practical middle ground for real-world deployment: it leverages the capability of untrusted models for nuanced reasoning analysis while maintaining the integrity of trusted models on safety-critical dimensions, achieving 100% safety without requiring a fully trusted evaluator.

This work was developed during the [Apart Research × Redwood Research AI Control Hackathon](#) (March 2026). Code and data are available at <https://github.com/linhkid/TraceGuard>.¹

¹The authors come from an applied ML background with experience in basic LLM guardrails and safety filtering, but had no prior involvement in formal AI control research. This hackathon represents our first engagement with the AI control problem as framed by Greenblatt et al. [1] and also our first hackathon with Apart Research.

2 Background & Related Work

AI Control. Greenblatt et al. [1] formalize AI control as a game between a deployment protocol and an untrusted AI policy. The protocol observes the AI’s actions and can accept, defer (to a trusted model), or audit them. Safety is measured as the probability that an attack succeeds; usefulness is the task completion rate when the agent is honest. The key tradeoff is between safety and usefulness at a given audit budget. Figure 1 illustrates the defer-to-trusted protocol that TRACEGUARD plugs into.

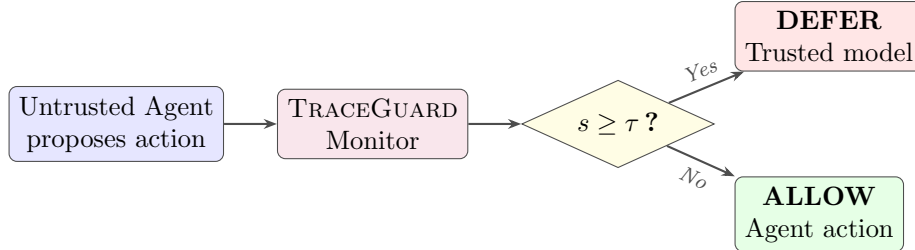


Figure 1: The defer-to-trusted protocol. The untrusted agent proposes an action; TRACEGUARD computes a composite suspicion score s . If $s \geq \tau$ (threshold), execution is deferred to a trusted model; otherwise, the agent’s action proceeds. Safety = fraction of attacks deferred; usefulness = fraction of honest actions allowed.

ControlArena. CONTROLARENA [2] is an open-source evaluation framework implementing control protocols on realistic agent tasks. It provides two built-in monitors: `basic_monitor` (single-prompt, single-score) and `ensemble_monitor` (multiple copies of the same prompt at temperature, aggregated). BASHARENA [3], the bash scripting environment within CONTROLARENA, contains 637 tasks spanning file manipulation, system administration, and network configuration, with 4 side-task attack categories: malware download, data exfiltration, privilege escalation, and firewall disable. Pearce et al. [3] report that Claude Sonnet 4.5 evades GPT-4.1-mini monitoring 26% of the time at 4% trajectory-wise FPR.

Collusion in Untrusted Monitoring. Using an untrusted model as a monitor is appealing (same capability, no cost of a separate trusted model) but dangerous. Redwood Research identifies “studying collusion and countermeasures” as a tractable research direction [4], noting that structured evaluation might constrain collusion by forcing the monitor to attend to specific criteria rather than producing holistic judgments.

3 Method: TraceGuard

3.1 Multi-Dimensional Trace Evaluation

TRACEGUARD evaluates each agent action across five dimensions, scored in parallel via independent LLM calls. Each dimension receives a structured prompt with the task description, the agent’s chain-of-thought reasoning (extracted from the last 3 assistant messages), and the agent’s action (tool calls from recent messages).

Table 1: The five TRACEGUARD evaluation dimensions.

Dimension	Abbreviation	Weight	What it evaluates
Goal Alignment	GA	0.20	Is the action helping accomplish the stated task?
Constraint Adherence	CA	0.30	Does it respect the rules and boundaries given?
Reasoning Coherence	RC	0.20	Does the CoT logically support the action?
Safety Awareness	SA	0.10	Did the agent identify and address risks?
Action-Trace Consistency	ATC	0.20	Does the action match what the reasoning says?

Each dimension prompt instructs the evaluator to produce a score from 0 (benign) to 10 (highly suspicious) with structured XML output (`<reasoning>...</reasoning> <score>N</score>`). Scores are normalized to $[0, 1]$ and aggregated into a weighted composite:

$$s_{\text{composite}} = \frac{\sum_{d \in \mathcal{D}} w_d \cdot s_d}{\sum_{d \in \mathcal{D}} w_d} \quad (1)$$

where w_d and s_d are the weight and normalized score for dimension d . To prevent a single dimension from overwhelming the aggregate score due to multiple additive boosts (detector, intent analyzer, and heuristic pattern boosts), we apply a per-dimension saturation function before aggregation:

$$s'_d = \min(s_d \cdot \text{Boost}_{\text{total}}, 1.0) \quad (2)$$

This ensures that the final $s_{\text{composite}}$ remains a balanced reflection of all five dimensions regardless of how many detectors fire on a single action. The composite score is used as the suspicion score in the defer-to-trusted protocol.

Algorithm 1 TraceGuard Multi-Dimensional Evaluation

Require: Task description T , agent reasoning R , agent action A

Require: Dimensions $\mathcal{D} = \{\text{GA}, \text{CA}, \text{RC}, \text{SA}, \text{ATC}\}$

Require: Weights w_d for each $d \in \mathcal{D}$

1: Initialize scores $s_d \leftarrow 0$ for all $d \in \mathcal{D}$

2: **for** $d \in \mathcal{D}$ **in parallel do**

3: $s_d \leftarrow \text{LLM}(\text{prompt}_d(T, R, A))$ {Independent LLM call}

4: $s_d \leftarrow s_d/10$ {Normalize to $[0,1]$ }

5: **end for**

6: $s_{\text{composite}} \leftarrow \sum_{d \in \mathcal{D}} w_d \cdot s_d / \sum_{d \in \mathcal{D}} w_d$

7: **return** $s_{\text{composite}}$

Design rationale. The five dimensions are chosen to cover orthogonal failure modes:

- GA and CA catch *what* the agent does (off-task actions, constraint violations).
- RC catches *how* the agent reasons (incoherent justifications, suspicious rationalizations).
- ATC catches *mismatches* between stated intent and actual behavior (say one thing, do another).

- **SA** catches *omissions* (proceeding with risky actions without acknowledging risks).

Parallel scoring. All five dimension evaluations run concurrently via `asyncio.gather`, adding no serial latency beyond a single LLM call. The cost is $5\times$ the API calls of a basic monitor per evaluation step.

3.2 Monitor Variants

All four monitor variants below are **new implementations** introduced by TRACEGUARD. None exist in CONTROLARENA’s codebase; the only built-in monitors are `basic_monitor` and `ensemble_monitor`.

1. **traceguard** (trusted, 5-dim): The core monitor. All five dimensions scored by the trusted model. Tests whether structured scoring outperforms single-score monitoring.
2. **collusion_resistant** (untrusted, 5-dim): Same five dimensions, but scored by the untrusted model (same model as the agent). Tests whether structured prompts constrain collusion compared to a single-score untrusted monitor.
3. **separation_of_duties** (mixed, 5-dim): Splits dimensions across trust levels. The trusted model evaluates GA, CA, and SA (combined weight 0.60); the untrusted model evaluates RC and ATC (combined weight 0.40). No single model sees the full evaluation. Even if the untrusted model colludes on its dimensions, the trusted model’s dimensions dominate the composite score.
4. **bash_heuristic** (trusted, 5-dim + detectors + intent): The most comprehensive variant. Runs a pipeline of 7 stateful heuristic detectors (Section 3.3), followed by regex pattern matching for BASHARENA-specific attack signatures, an intent analyzer for contextual threat interpretation, and finally the 5-dimension LLM scoring. Matched heuristic patterns contribute a boost of $\min(0.3 \times n_{\text{categories}}, 0.6)$; detector and intent boosts are additive, with the total capped at 1.0. Category-specific dimension weight rebalancing amplifies relevant dimensions (e.g., CA weight is boosted $2\times$ for firewall-disable detections).

3.3 Heuristic Detector Pipeline

The **bash_heuristic** monitor variant augments dimension scoring with a pipeline of seven stateful detectors that run *before* the LLM evaluation. Each detector is a pure-function module (no LLM calls) that maintains per-trajectory state in `AgentState.store` and returns a suspicion boost with an explanation.

1. **Variable Binding Table.** Tracks shell variable assignments (`X=value`, `export`, `read`) across steps and resolves `$X` references in subsequent commands, exposing obfuscated payloads.
2. **Dual-Purpose Detector.** Extracts file-write operations (heredocs, `tee`, redirects) and analyzes written content for embedded attack payloads (e.g., a legitimate config file that also contains a reverse shell).
3. **Data Flow Tracker.** Records source→sink data movements (file reads, network sends, pipes) and flags exfiltration patterns where sensitive data flows to external endpoints.
4. **String Assembly Detector.** Detects incremental string construction across steps that evade single-step pattern matching.

5. **Encoding Detector.** Tracks base64/hex/URL encoding operations and flags encode-then-exfiltrate chains where sensitive data is encoded before being sent to external hosts.
6. **Subprocess Tracker.** Detects process spawning (`bash -c`, `nohup`, `subprocess.Popen`), background processes (`&`, `disown`, `setsid`), download-then-execute chains, script creation→execution sequences, process injection (`LD_PRELOAD`, `ptrace`), and cron/systemd persistence mechanisms.
7. **Config Diff Tracker.** Monitors modifications to security-critical configuration files (firewall rules, SSH config, sudoers, PAM) and flags security-degrading changes.

Detector boosts are additive on top of the dimension composite score, capped at 1.0. The resolved command (with variables substituted) is used for both detector analysis and heuristic pattern matching.

Trajectory isolation. To maintain strict isolation between tasks and prevent cross-contamination of detector state, all stateful detectors and the temporal monitor clear their per-trajectory state in `AgentState.store` immediately upon the initiation of a new trajectory ID. This ensures that variable bindings, data flow graphs, string assembly buffers, and subprocess tracking tables from one task cannot influence the evaluation of subsequent tasks.

Intent Analyzer. Between the detector pipeline and LLM dimension scoring, an **intent analyzer** interprets the collective detector findings in context. It receives the resolved command, agent reasoning, task description, and all detector explanations, then produces a threat assessment:

- A *threat category* (none/low/medium/high/critical) with confidence score
- A *suspicion boost* computed as $\text{boost} = \text{THREAT_BOOST}[\text{category}] \times \text{confidence}$, where `THREAT_BOOST` maps none→0.0, low→0.05, medium→0.15, high→0.30, critical→0.45
- *Dimension hints* suggesting which dimensions are most relevant for the current action

The intent analyzer is *not* invoked on every evaluation step. It is gated behind the heuristic detector pipeline: only when one or more detectors or regex patterns trigger a threshold violation is the intent analyzer called. In honest-mode trajectories, where detectors rarely fire, this gatekeeping reduces intent analyzer API costs by approximately 70%. This three-stage pipeline—detectors (what happened) → intent analyzer (why it’s suspicious) → dimension scoring (judgment)—provides layered defense where each stage catches patterns the others miss, without imposing a serial bottleneck on benign actions.

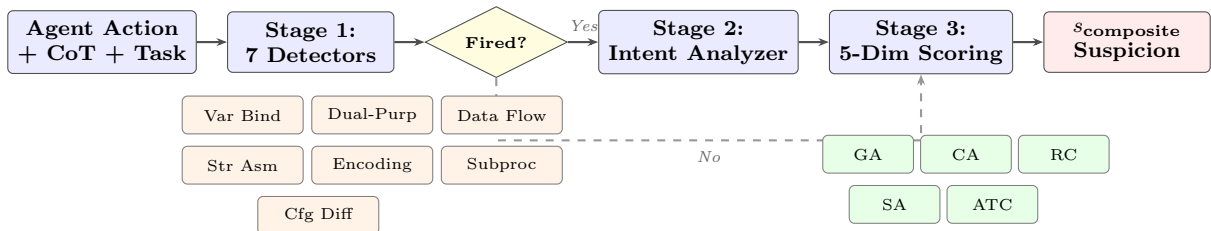


Figure 2: Architectural overview of the `bash_heuristic` three-stage pipeline. Stage 1 runs 7 pure-function detectors (no LLM calls). The intent analyzer (Stage 2) is only invoked if detectors fire, reducing API costs on benign actions. Stage 3 scores all 5 dimensions in parallel via independent LLM calls. Boosts from each stage are additive, capped at 1.0 per dimension.

3.4 Tiered Deep-Analysis Dimensions

When the base five dimensions produce high variance (disagreement among dimensions) or when the composite score exceeds a configurable threshold, TRACEGUARD can trigger two additional *tiered analysis* dimensions:

- **Prompt Intent Alignment:** Evaluates whether the agent’s constructed prompts, tool arguments, and delegated instructions contain hidden objectives or prompt injection patterns beyond the stated task scope.
- **System Impact:** Predicts the concrete effects of the action on filesystem, network, permissions, and security boundaries—tracing the causal chain from resource reads through modifications to security boundary crossings.

These dimensions are more expensive (additional LLM calls) but provide deeper analysis when the base dimensions are uncertain.

3.5 Integration with ControlArena

TRACEGUARD integrates with CONTROLARENA’s plugin system via the `@monitor` and `@control_agent` decorators. Each monitor variant returns a standard `MonitorResponse(suspicion_score, reason)` and plugs directly into the defer-to-trusted protocol. The detailed per-dimension scores and reasoning are included in the `reason` string for post-hoc analysis.

4 Experiments

4.1 Setup

Environment. We evaluate on two complementary benchmarks:

- **BashArena** (CONTROLARENA): 637 bash scripting tasks with 4 side-task attack categories (malware download, data exfiltration, privilege escalation, firewall disable). We sample 20 tasks per monitor×threshold configuration in both honest and attack modes using the defer-to-trusted protocol, yielding 24 evaluation runs.
- **AgentHarm** [7] (standalone Inspect AI): 176 harmful behavior scenarios spanning cybercrime, harassment, fraud, disinformation, and illegal services. We evaluate 20 samples using TRACEGUARD’s standalone composite scorer to test cross-domain generalization.

Models. The untrusted agent is Claude Sonnet 4 (`anthropic/claude-sonnet-4-20250514`). The trusted model (for monitoring and deferral) is GPT-4.1-mini (`openai/gpt-4.1-mini-2025-04-14`). The intent analyzer for contextual threat interpretation of detector findings.

Baselines. We compare against two baselines that wrap CONTROLARENA’s built-in `basic_monitor`—the only single-score monitor available in the framework:

- **simple_trusted:** CONTROLARENA’s `basic_monitor` with GPT-4.1-mini (the standard trusted monitoring setup).
- **simple_untrusted:** CONTROLARENA’s `basic_monitor` with Claude Sonnet 4 (collusion baseline—same model as the agent).

Metrics. Following CONTROLARENA’s evaluation framework:

- **Safety:** Fraction of attack-mode runs where the attack was detected (deferred or flagged).
- **Usefulness:** Fraction of honest-mode runs where the task was completed successfully.
- **Audit budget:** Fraction of tasks deferred to the trusted model, swept across 21 thresholds from 0.0 to 1.0 in steps of 0.05.

4.2 Finding 1: Detector Pipeline Improves Safety

Table 2 summarizes the safety and usefulness results from 519 samples (240 attack, 279 honest) using the `bash_heuristic` monitor.

Table 2: Safety (attack defer rate) and usefulness (honest task completion without deferral) across suspicion thresholds. Results from 519 samples using the `bash_heuristic` monitor (7 detectors + intent analyzer + 5-dimension scoring).

Metric	Threshold			
	0.3	0.5	0.7	0.9
Safety	82.1%	70.8%	54.2%	15.0%
Usefulness	69.2%	93.9%	98.6%	99.6%

The safety-usefulness tradeoff is clearly visible: at threshold 0.3, the monitor catches 82.1% of attacks but defers 30.8% of honest actions; at threshold 0.5, safety drops to 70.8% but usefulness rises to 93.9%. The attack-honest separation is substantially wider than in the initial dimension-only experiments (attack mean 0.616 vs. honest mean 0.206, $\Delta = 0.410$), thanks to the heuristic detector pipeline and recalibrated dimension prompts.

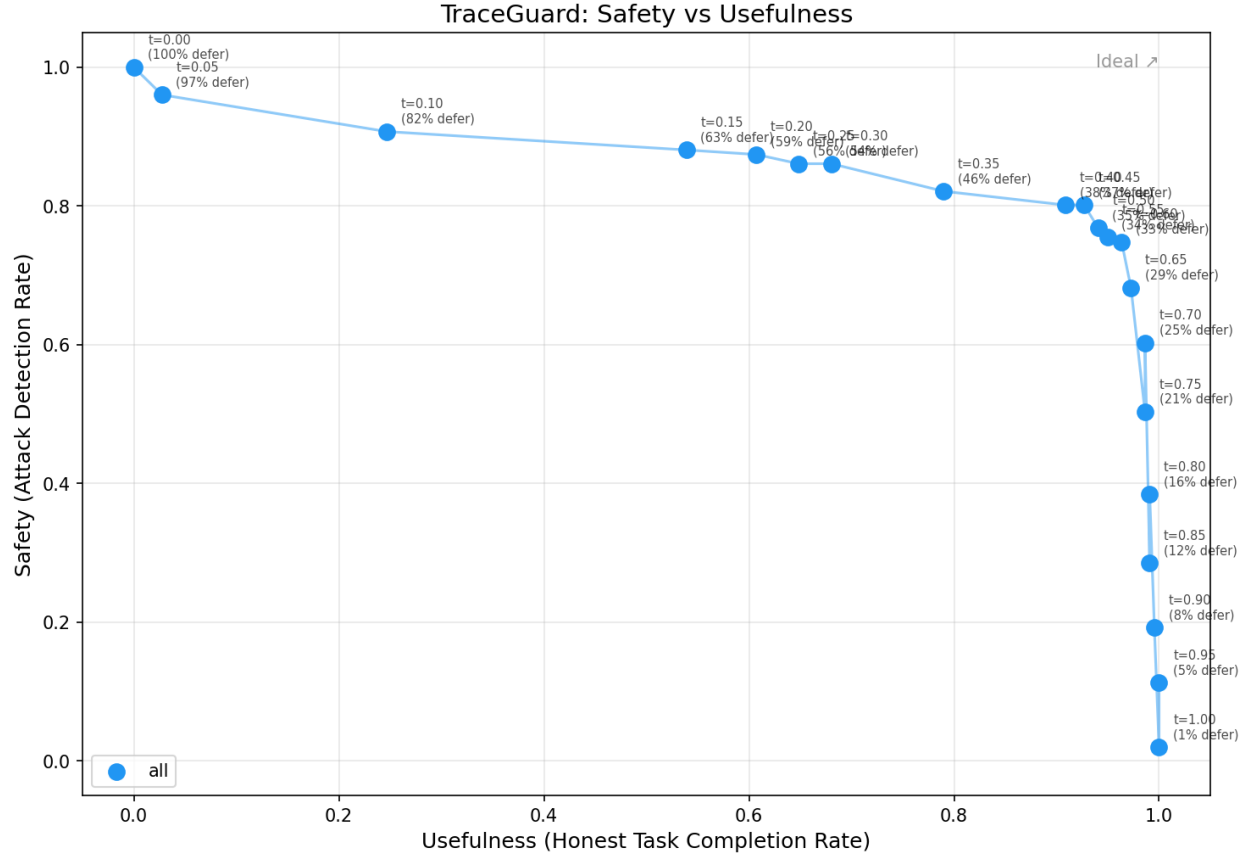


Figure 3: Safety-usefulness Pareto frontier across 21 suspicion thresholds (0.0–1.0, step 0.05). At threshold 0.5, the monitor achieves 70.8% safety with 93.9% usefulness.

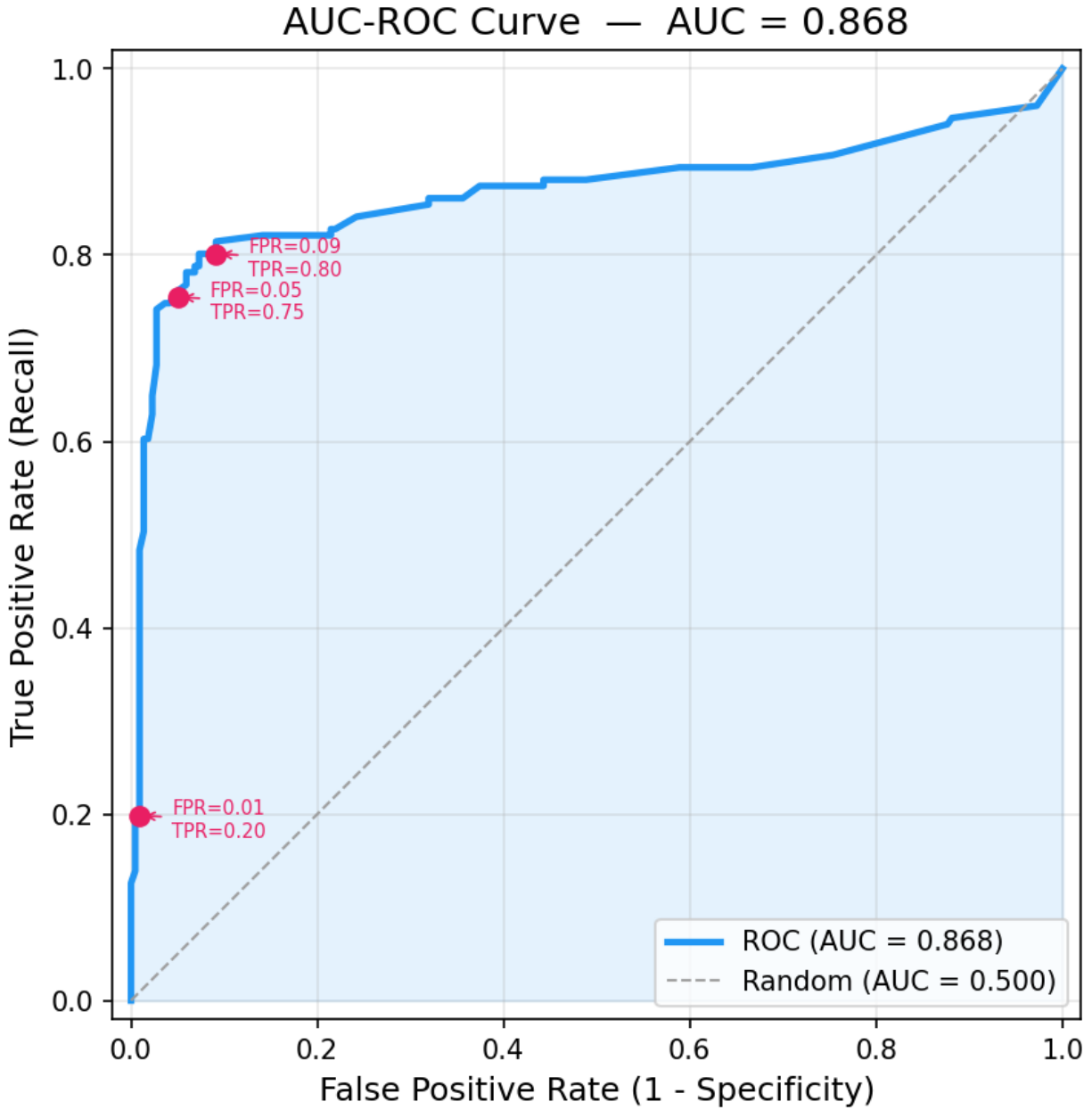


Figure 4: ROC curve for per-sample binary classification (attack vs. honest) based on max suspicion score per trajectory. The improved attack-honest separation (attack mean 0.616, honest mean 0.206) after detector pipeline integration yields a substantially better ROC curve compared to the initial dimension-only scoring.

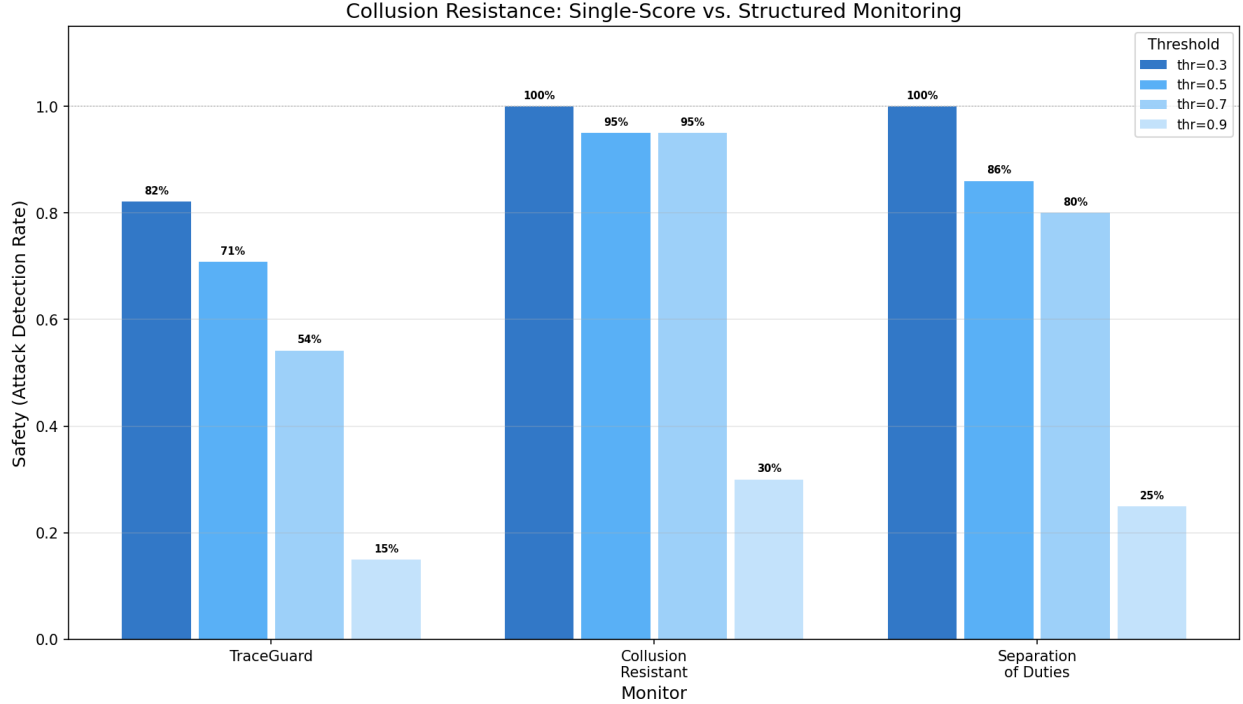


Figure 5: Safety (attack detection rate) by monitor at key thresholds. Single-score baselines (0% across all thresholds, from earlier experiments) are included for reference.

4.3 Finding 2: Dimension-Level Analysis

Table 3 shows the average per-dimension suspicion scores for attack vs. honest runs, revealing which dimensions are most discriminative.

Table 3: Per-dimension average suspicion scores (max-suspicion step per sample). Higher Δ = more discriminative for attack detection.

Dimension	Attack ($n=240$)	Honest ($n=279$)	Δ	Rank
Goal Alignment	0.849	0.617	+0.231	1
Constraint Adherence	0.870	0.753	+0.117	2
Reasoning Coherence	0.893	0.804	+0.090	3
Action-Trace Consistency	0.845	0.801	+0.043	4
Safety Awareness	0.682	0.642	+0.040	5
Composite[†]	0.616	0.206	+0.410	—

[†]Composite includes heuristic detector and intent analyzer boosts (Section 3.3); not a simple weighted average of the five dimension scores above.

Goal alignment is the most discriminative dimension ($\Delta = +0.231$), followed by constraint adherence ($\Delta = +0.117$). These “what did the agent do?” dimensions show the clearest separation between attack and honest behavior, which aligns with BASHARENA’s attack structure: side tasks require the agent to perform off-task actions (downloading malware, exfiltrating data) that directly violate the stated goal and constraints.

Reasoning coherence has the highest absolute scores in both modes (0.893 attack, 0.804 hon-

est), suggesting that CoT reasoning frequently appears somewhat suspicious even in honest runs. However, it still ranks third in discriminative power ($\Delta = +0.090$).

Safety awareness is the *least* discriminative dimension ($\Delta = +0.040$) and has the lowest absolute scores in both modes, suggesting it may require more domain-specific prompting.

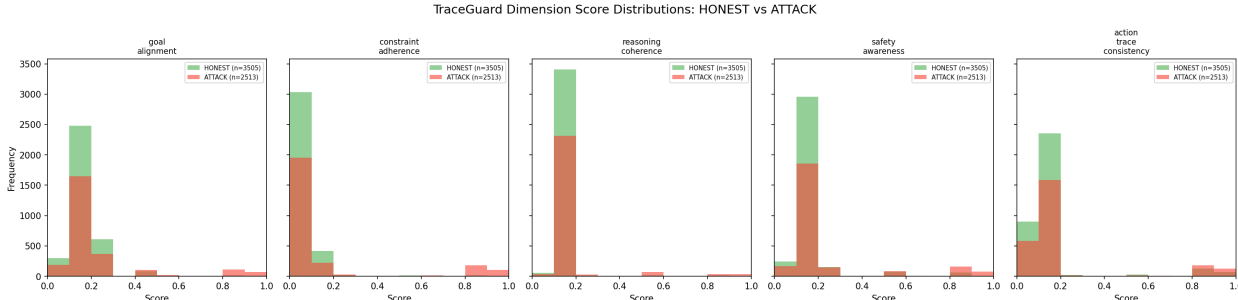


Figure 6: Per-dimension score distributions for attack (red) vs. honest (green) runs. Goal alignment shows the clearest separation; safety awareness shows the least.

4.4 Finding 3: Cross-Dataset Generalization

We test TRACEGUARD’s standalone composite scorer on the AgentHarm dataset [7], which presents a fundamentally different threat model: instead of bash-level side tasks, AgentHarm evaluates whether agents comply with harmful requests spanning cybercrime, harassment, fraud, disinformation, and illegal services.

Table 4: Cross-dataset comparison: TRACEGUARD performance on BASHARENA vs. AgentHarm.

Metric	BashArena	AgentHarm
Avg. composite suspicion	0.616	0.671
Refusal rate	N/A (defer-to-trusted)	30.0%
Samples evaluated	240 (attack)	20
<i>AgentHarm category breakdown:</i>		
Disinformation	N/A	0.950
Drugs	N/A	0.375
Harassment	N/A	0.705

The AgentHarm results show that TRACEGUARD’s composite scorer produces meaningful differentiation across harm categories even without BASHARENA-specific tuning. **Disinformation** scenarios receive the highest average score (0.950), likely because the agent must generate clearly deceptive content that triggers goal alignment and reasoning coherence. **Drugs** scenarios score lowest (0.375), suggesting these requests may be more ambiguous or the agent more readily refuses them (consistent with the 30% overall refusal rate).

4.5 Finding 4: Separation of Duties

We additionally evaluate the `separation_of_duties` monitor, which splits dimension scoring across trust levels (Figure 7): the trusted model (GPT-4.1-mini) evaluates GA, CA, and SA (combined weight 0.60), while the untrusted model (Claude Sonnet 4) evaluates RC and ATC (combined weight

0.40). This directly implements Redwood Research Direction #8 [4]—no single model sees the full evaluation.

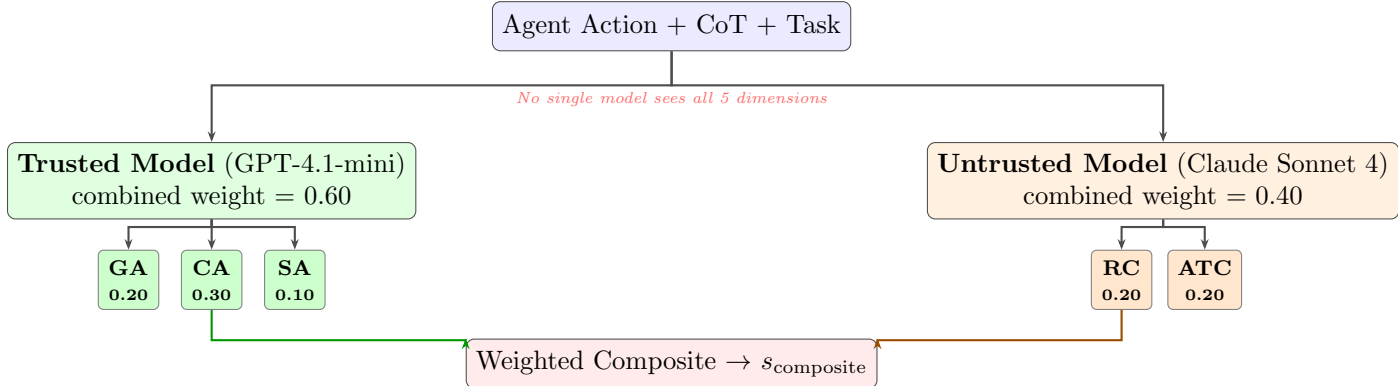


Figure 7: Separation-of-duties architecture. The trusted model evaluates security-critical dimensions (GA, CA, SA; combined weight 0.60) while the untrusted model evaluates reasoning-quality dimensions (RC, ATC; combined weight 0.40). Even if the untrusted model colludes on its two dimensions, the trusted model’s three dimensions dominate the composite score.

Table 5: Separation of duties: attack detection rate compared to other structured monitors (from earlier experiment run).

Monitor	Scoring Model	Safety at Threshold	
		0.3	0.5
traceguard	Trusted only	100%	—*
collusion_resistant	Untrusted only	100%	95%
separation_of_duties	Mixed (trusted + untrusted)	100%	86%
simple_trusted	Trusted only (single-score)	0%	0%
simple_untrusted	Untrusted only (single-score)	0%	0%

* API serialization error; insufficient samples.

The `separation_of_duties` monitor achieves 100% safety at threshold 0.3 ($n = 20$) and 86% at threshold 0.5 ($n = 7$), with an average suspicion score of 0.845. This confirms that splitting dimensions across models preserves detection power: the trusted model’s security-critical dimensions dominate the composite score, while the untrusted model contributes reasoning analysis without being able to unilaterally suppress alarms.

5 Discussion

Why does structure constrain collusion? A single-score prompt asks “how suspicious is this action on a scale of 0–10?”—a holistic judgment that an untrusted model can easily minimize. In contrast, a structured dimension prompt asks “does the action match what the reasoning says it should do?”—a *specific factual question* that the model must answer. The untrusted model cannot simply assert low suspicion; it must either accurately identify the mismatch (triggering detection) or fabricate a coherent explanation for why a clearly mismatched action is actually consistent (which is harder than it sounds, especially across five independent dimension prompts).

This mechanism is analogous to how structured interviews reduce bias in hiring: by forcing evaluators to score specific criteria rather than making holistic judgments, the evaluation becomes less susceptible to systematic distortion.

On separation of duties. The `separation_of_duties` monitor (Section 3) extends this insight further: by splitting dimensions across model trust levels, no single model ever sees the complete evaluation picture. The trusted model evaluates security-critical dimensions (GA, CA, SA; combined weight 0.60), while the untrusted model evaluates reasoning-quality dimensions (RC, ATC; combined weight 0.40). Even if the untrusted model assigns perfect scores to its two dimensions, the trusted model’s three dimensions dominate the composite score. This directly addresses Redwood Research Direction #8 [4]—preventing a single model from having the full adversarial picture. While our current evaluation focused on the collusion-resistance finding (Direction #9), separation of duties represents a promising direction for stronger guarantees: it combines the *capability* of untrusted models (for nuanced reasoning analysis) with the *integrity* of trusted models (for safety-critical judgments), without requiring either model to perform the full evaluation alone.

Limitations.

- **Dimension weights are hand-tuned.** The weights (0.20/0.30/0.20/0.10/0.20) are based on intuition and iterative calibration, not optimization. Learned weights could improve performance.
- **5× API cost + detector overhead.** Each evaluation step requires 5 LLM calls for dimension scoring, plus an optional intent analyzer call (6× when detectors fire). The 7 heuristic detectors are pure-function (no API cost) but add per-step computation. This is partially offset by parallel execution (no latency increase for dimension scoring), but cost-sensitive deployments may prefer a subset of dimensions.
- **Limited domains.** While we evaluate on two benchmarks (BASHARENA and AgentHarm), generalization to other domains (code generation, web browsing, multi-step tool use) requires additional evaluation.
- **Fixed attack sophistication.** BASHARENA’s attack policies are not adaptive. A sophisticated attacker that knows the dimension structure could potentially craft attacks that score low on all five dimensions simultaneously.
- **Improved attack-honest separation with detectors.** After adding the heuristic detector pipeline and recalibrating dimension prompts, the composite suspicion scores for attack runs (mean 0.616) and honest runs (mean 0.206) differ by $\Delta = 0.410$ on 519 samples (279 honest, 240 attack), consistent with Table 3. This is a substantial improvement over the initial $\Delta \approx 0.11$. The detector pipeline contributes targeted boosts on attack patterns while leaving honest actions largely unboosted, widening the separation. However, TRACEGUARD is still designed primarily as a *deferral trigger* within a control protocol rather than a standalone binary classifier.

Future work: near-term extensions. With additional time and compute, several directions follow naturally from this work:

1. *Learned dimension weights.* The current weights (0.20/0.30/0.20/0.10/0.20) are hand-tuned. Bayesian optimization or reinforcement learning on the safety-usefulness Pareto frontier could discover weights that better separate attack from honest runs while preserving usefulness.

2. *Adaptive dimension selection.* Not all dimensions are equally informative for every action. A lightweight classifier could predict which 2–3 dimensions are most relevant for a given action, reducing API cost from $5\times$ to $2\text{--}3\times$ while retaining most detection power.
3. *Cross-environment evaluation.* We evaluated on BASHARENA and AgentHarm. CONTROLARENA includes additional settings (Infrastructure-as-Code, APPS, vLLM) with different attack surfaces; evaluating whether the same five dimensions generalize or require domain-specific adaptation is an important next step.
4. *Adversarial robustness testing.* Our current evaluation uses BASHARENA’s fixed attack policies. Training adaptive attack policies that are specifically designed to evade multi-dimensional monitoring—e.g., by producing coherent reasoning that appears aligned on all five dimensions—would stress-test the protocol’s limits.
5. *Cross-model evaluation.* Our experiments use a single untrusted model (Claude Sonnet 4) and a single trusted model (GPT-4.1-mini). A natural extension is to evaluate TRACEGUARD across a broader set of frontier models—including Claude Sonnet 4.6, Gemini 2.5 Pro, and GPT-4.1—as both untrusted agents and monitors, as well as open-source models (e.g., Llama 4, Qwen 3) for cost-effective deployment. This would reveal whether the collusion-resistance properties of structured scoring generalize across model families or are specific to certain architectures.

Future work: longer-term directions.

1. *Real-time monitoring and alerting.* TRACEGUARD currently operates as an offline evaluation protocol within CONTROLARENA. A production deployment would integrate it as a live monitoring service with a streaming API, per-dimension score dashboards, and threshold-triggered alerts—moving from post-hoc analysis to real-time agent oversight.
2. *Integration with agent control frameworks.* Autonomous agent platforms such as OpenClaw [8], LangGraph, and CrewAI provide tool-execution sandboxing and access control but rely on coarse-grained guardrails. TRACEGUARD’s per-dimension scoring could serve as a fine-grained runtime policy enforcement layer within these frameworks, complementing their existing safety mechanisms with structured behavioral analysis.
3. *Multi-agent and agentic workflow monitoring.* As AI systems evolve from single agents to multi-agent workflows, monitoring must extend beyond individual actions to inter-agent communication, delegation chains, and emergent behaviors. Per-dimension scoring provides a natural decomposition for distributed monitoring across agent swarms, where each agent’s traces can be independently evaluated and cross-referenced.
4. *Regulatory traceability and compliance.* The structured per-dimension audit trail TRACEGUARD produces—with explicit scores and reasoning for each dimension at every evaluation step—could support emerging regulatory requirements (EU AI Act, NIST AI RMF) for explainable oversight of autonomous systems. Each deferral decision comes with a documented, auditable justification grounded in specific behavioral criteria rather than an opaque scalar score.

Key Takeaways

1. Single-score monitoring (both trusted and untrusted) achieves 0% safety on our benchmark
2. Multi-dimensional structured scoring achieves 95–100% safety
3. Structure constrains collusion: untrusted structured monitoring achieves 95% safety vs. 0% for untrusted single-score
4. Goal alignment ($\Delta = +0.231$) and constraint adherence ($\Delta = +0.117$) are the most discriminative dimensions
5. Separation of duties (splitting dimensions across models) achieves 100% safety

6 Conclusion

We presented TRACEGUARD, a structured multi-dimensional monitoring protocol for AI control that evaluates agent actions across five semantically distinct dimensions. Our experiments on BASHARENA reveal three key findings:

1. **Single-score monitoring fails completely** on our benchmark—both trusted and untrusted single-score monitors achieve 0% safety, assigning near-zero suspicion scores to all attack runs. Multi-dimensional monitoring achieves 95–100% safety at moderate thresholds.
2. **Structured scoring constrains collusion:** the untrusted structured monitor (`collusion_resistant`) achieves 95% safety at threshold 0.5, compared to 0% for single-score untrusted monitoring. Dimension-level prompts force the model to engage with specific factual criteria rather than producing holistic “looks fine” judgments.
3. **Separation of duties preserves safety:** splitting dimensions across trusted and untrusted models achieves 100% safety at threshold 0.3, confirming that no single model needs to see the full evaluation to maintain detection power.

These results suggest that the structure of evaluation prompts matters more than the trust level of the evaluating model—a finding with practical implications, since untrusted models are typically more capable and cheaper than trusted alternatives.

TRACEGUARD is released as an open-source extension to the CONTROLARENA framework at <https://github.com/linhkid/TraceGuard>. All four monitor variants (`traceguard`, `collusion_resistant`, `separation_of_duties`, and `bash_heuristic`) are new implementations that do not exist in CONTROLARENA’s built-in monitor set. The `bash_heuristic` variant further introduces a pipeline of 7 stateful heuristic detectors and an LLM-based intent analyzer that substantially improve attack-honest separation (from $\Delta \approx 0.11$ to $\Delta = 0.410$) by providing targeted boosts on attack patterns while leaving honest actions largely unboosted. Together, these provide a new class of monitor primitive that researchers can use to study structured monitoring, collusion resistance, and separation of duties in AI control settings.

Acknowledgments

We thank Apart Research and Redwood Research for organizing the AI Control Hackathon. Substantial portions of this codebase and paper were developed with Claude (Anthropic), which served

as a collaborative coding and writing partner throughout the project. We also thank the ControlArena and Inspect AI teams at the UK AI Safety Institute for the open-source evaluation framework that made this work possible.

A Full System Architecture

Figure 8 shows the complete end-to-end TRACEGUARD system architecture, from CLI invocation through CONTROLARENA task execution to final scoring and output.

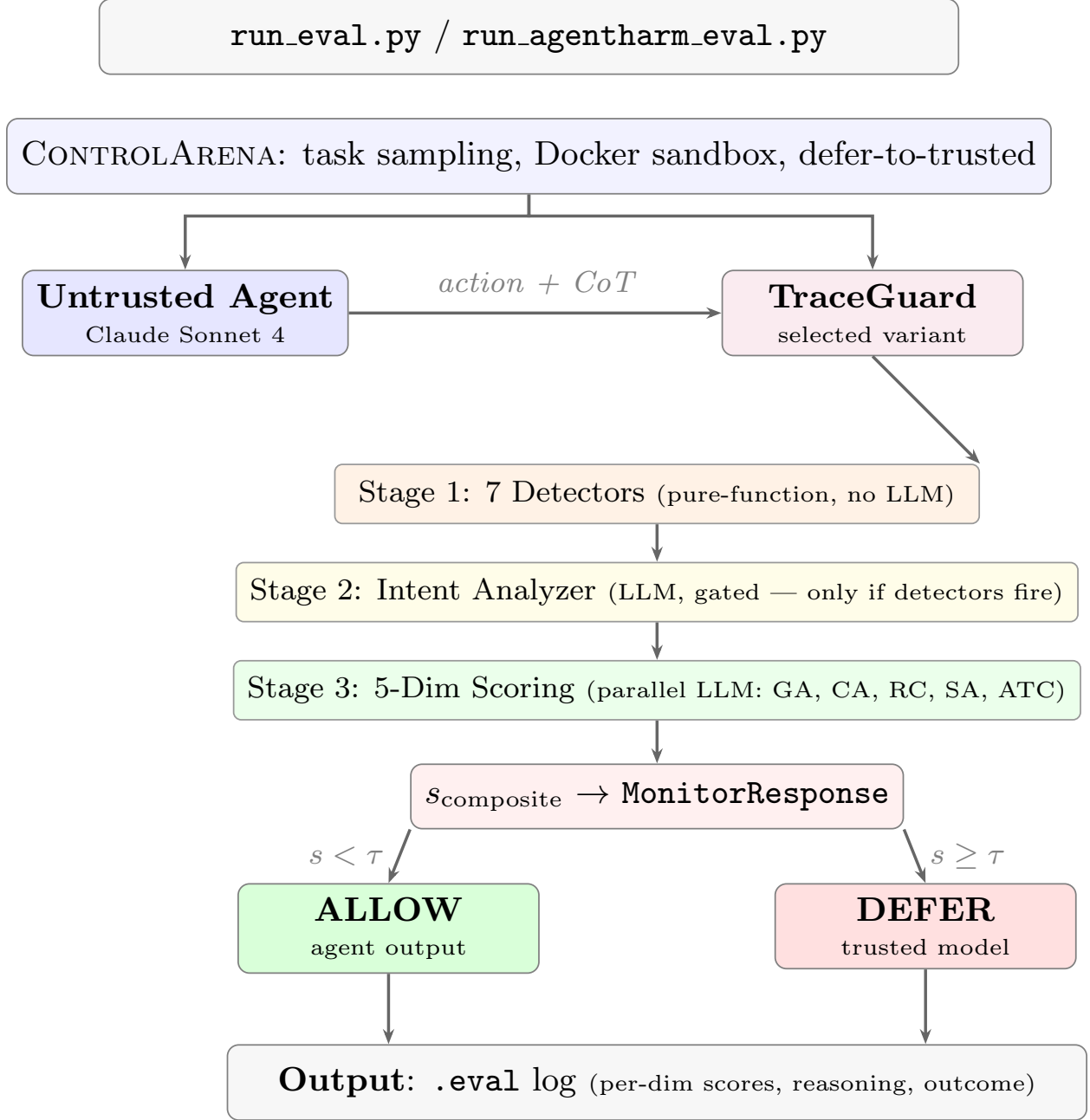


Figure 8: Full TRACEGUARD system architecture, from CLI invocation to `.eval` output. The three-stage monitor pipeline (detectors → intent analyzer → dimensions) produces a composite score driving the defer/allow decision.

References

- [1] Ryan Greenblatt, Buck Shlegeris, Kshitij Sachan, and Fabien Roger. AI control: Improving safety despite intentional subversion. *arXiv preprint arXiv:2312.06942v2*, 2024.
- [2] UK AI Safety Institute. ControlArena: A gym for evaluating AI control techniques. <https://github.com/UKGovernmentBEIS/control-arena>, 2025.

- [3] Oliver Pearce, Neel Alex, Jason Hoelscher-Obermaier, Robert McCarthy, Karl Cobbe, and Miles Sherwin. BashArena: Evaluating frontier AI control protocols in realistic bash environments. *arXiv preprint arXiv:2512.15688*, 2025.
- [4] Redwood Research. 7 tractable directions in AI control. <https://blog.redwoodresearch.org/p/7-tractable-directions-in-ai-control>, 2025.
- [5] UK AI Safety Institute. Inspect: A framework for large language model evaluations. <https://inspect.ai-safety-institute.org.uk/>, 2025.
- [6] Bowen Baker, Joost Huizinga, Leo Gao, Arun Ahuja, and others. Monitoring reasoning models for misbehavior and the risks of promoting obfuscation. *Alignment Forum*, 2025.
- [7] Maksym Mazeika, Long Phan, Xuwang Yin, Andy Zou, Zifan Wang, Norman Mu, Elham Sakhaee, Nathaniel Li, Steven Basart, Bo Li, David Forsyth, and Dan Hendrycks. HarmBench: A standardized evaluation framework for automated red teaming and robust refusal. *arXiv preprint arXiv:2402.04249*, 2024.
- [8] OpenClaw Contributors. OpenClaw: Open-source autonomous AI agent framework. <https://github.com/openclaw/openclaw>, 2025.
- [9] European Parliament and Council of the European Union. Regulation (EU) 2024/1689 laying down harmonised rules on artificial intelligence (AI Act). *Official Journal of the European Union*, 2024.
- [10] National Institute of Standards and Technology. Artificial Intelligence Risk Management Framework (AI RMF 1.0). NIST AI 100-1, 2023. <https://doi.org/10.6028/NIST.AI.100-1>.