AGENT HARNESS DISTILLATION: INFERENCE-TIME HARNESS EXTRACTION AND EXPLOITATION IN AUTONOMOUS MULTI-AGENT SYSTEMS

Anonymous authors
Paper under double-blind review

ABSTRACT

Autonomous multi-agent systems (AMAS) built on large language models (LLMs), such as Hermes, increasingly rely on inference-time harnesses to coordinate reasoning and action. Constructing these harnesses requires substantial engineering effort and computational resources, as they are iteratively optimized over a combinatorial search space while co-evolving with the underlying LLM. Inference-time harnesses therefore constitute valuable intellectual property (IP). Although prior work has investigated IP leakage in static multi-agent systems with pre-configured architectures, it remains unclear whether similar risks arise in AMAS, where harness behavior emerges dynamically during inference. To address this gap, we introduce Agent Harness Distillation (AHD), a framework for studying the security risks arising from inference-time harness extraction in AMAS. We formalize harness extraction as a new security problem and develop an evaluation framework for quantifying such risks. AHD extracts inference-time harness capabilities from a target agent through black-box interactions and consists of two stages. In the pre-distillation stage, AHD infers inference-time harness behaviors from the responses of the target agent and constructs an initial harness. In the post-distillation stage, AHD iteratively refines the initial harness to align with the behavioral patterns of the target agent. Experiments on real-world AMAS across multiple backbone LLMs demonstrate the effectiveness of AHD and reveal substantial IP leakage risks. We further propose a deception-based defense that reduces harness extraction effectiveness while preserving the utility of the protected agent. Our findings uncover a previously underexplored security threat to AMAS.

1 INTRODUCTION

The rapid development of large language models (LLMs) has accelerated the deployment of LLM agents in real-world software development (Wang et al., 2026c; Qian et al., 2024). In particular, Autonomous Multi-Agent Systems (AMAS), such as Claude Code and Hermes, organize multiple agents, tools, and planning modules into automated workflows, substantially improving developer productivity and lowering the barrier to complex programming tasks (Liu et al., 2026). For each user task, these systems dynamically instantiate an inference-time harness (Zhao et al., 2026), which specializes a static infrastructure scaffold into a task-specific workflow. By orchestrating planning, tool use, and inter-agent collaboration, the harness substantially enhances the problem-solving capabilities of the backbone LLM.

However, designing an effective harness requires substantial engineering effort, manual experimentation, and iterative co-evolution with the underlying LLM. This process is further complicated by the large search space induced by choices over planning, tool orchestration, subagent roles, and inter-agent coordination (Zhou et al., 2026; Xu et al., 2026b; Lee et al., 2026). Consequently, inference-time harnesses represent valuable intellectual property (IP). Meanwhile, they introduce a new security concern: an adversary interacting with an AMAS may recover information about the inference-time harness, thereby extracting reusable system knowledge that leads to IP leakage.

Prior work has shown that components of static multi-agent systems can be extracted through interaction, leading to potential IP leakage (Wang et al., 2026a). However, it remains unclear whether

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.

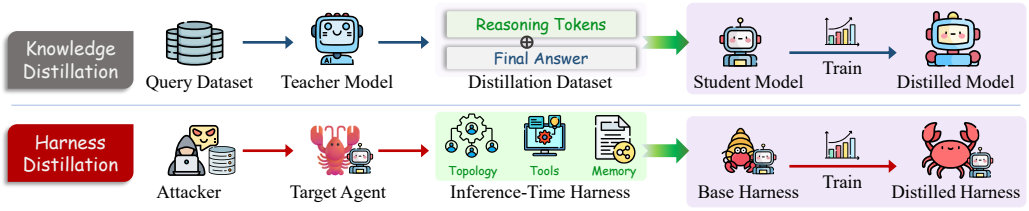


Figure 1: Comparison of the pipelines for our harness distillation and model knowledge distillation.

similar threats extend to AMAS. Unlike static multi-agent systems, task-dependent workflows in AMAS are dynamically instantiated at inference time rather than predefined.

Challenges. A systematic study of inference-time harness leakage in AMAS faces three key challenges. First, inference-time harnesses are dynamically instantiated during execution, making the leaked information difficult to define and formally characterize. Second, the severity of such leakage is difficult to measure. An inference-time harness comprises both explicit structural components and implicit execution strategies, dimensions that existing metrics do not adequately capture. Third, the practical security impact remains unclear. Extracted harness information may not translate into actionable capabilities or effective exploitation. To address these challenges, we investigate the following research questions:

- **RQ1**: How can inference-time harness leakage in AMAS be formally defined and quantified?
- **RQ2**: Can inference-time harness information be extracted through black-box interactions, and what practical value does such leakage provide?
- **RQ3**: How can inference-time harnesses be protected against extraction attacks?

To answer RQ1, we formalize inference-time harness leakage in AMAS by introducing a unified abstraction that captures dynamically instantiated orchestration structures and execution strategies. We further develop an evaluation framework with four core metrics to quantify different dimensions of harness leakage. To answer RQ2, we investigate inference-time harness extraction attacks and their downstream exploitation in AMAS. Specifically, we propose *Agent Harness Distillation* (AHD), which extracts harness information from a target agent and transfers the corresponding orchestration capabilities to another agent. AHD consists of two stages. In the pre-distillation stage, AHD infers harness behaviors from observable agent responses and constructs an initial harness. In the post-distillation stage, AHD iteratively refines the initial harness to better match the target agent's behavioral patterns. To answer RQ3, we introduce a deception-based defense that provides plausible but misleading harness information in response to extraction attempts, reducing extraction fidelity while preserving the utility of the protected agent. Our contributions are as follows:

- We formalize inference-time harness leakage as a new security problem in AMAS and develop an evaluation framework to quantify the associated leakage risks.
- We propose AHD, a two-stage framework that extracts inference-time harness information from target agents through black-box interactions and transfers the corresponding orchestration capabilities. Experiments on real-world AMAS across multiple backbone LLMs demonstrate the effectiveness of AHD and reveal substantial IP leakage risks.
- We design a deception-based defense that provides misleading harness information to extraction attempts, reducing extraction fidelity while preserving the utility of the protected agent.

2 Preliminary Study

2.1 Motivation and Research Settings

Some AMAS have begun to encrypt prompts and task messages exchanged among agents. For example, Codex MultiAgentV2[1] records task information delegated from the primary agent to subagents as ciphertext. As a result, user developers cannot directly obtain the specific delegation content

[1]https://github.com/openai/codex/issues/28058

from local execution records. This encryption mechanism reduces the risk of IP leakage for agent providers. However, existing protection mainly focuses on inter-agent messages. Potential leakage of the inference-time harness through responses remains underexplored. Future AMAS may further restrict access to internal system logs, for example, by encrypting these logs (Pérez & Daffalla, 2025). In this setting, black-box responses will become a primary channel through which external observers infer internal orchestration structures. Motivated by this observation, we study a harness extraction attack under a strict black-box setting. The attacker can only submit queries through a standard interface and observe the responses. This setting demonstrates a practical security threat in real-world deployments.

2.2 Problem Formulation

System Model. We consider a black-box AMAS $\mathcal{A} = (\theta, \mathcal{H})$, where θ denotes the underlying foundation LLM and $\mathcal{H}$ denotes the static harness. For a query $x \in \mathcal{X}$, the agent $\mathcal{A}$ first constructs a task-specific inference-time harness $h_x = \mathcal{I}_{\mathcal{H}}(x)$, where $\mathcal{I}_{\mathcal{H}}$ is the harness instantiation function. The instantiated harness h_x specifies the execution strategy, including agent coordination, context assembly, tool interaction, and memory management. The final response is generated by executing the foundation LLM under the instantiated harness: $y = \mathrm{Exec}(\theta, h_x, x)$. Different from persistent model parameters or stored configurations, h_x is an ephemeral inference-time state. It only exists during the execution process and is discarded after producing the final response. Therefore, although the harness determines the agent's behavior, it is not directly accessible after inference.

Threat Model. We consider a black-box interaction setting, where an attacker can query the agent system $\mathcal{A}$ but cannot access its internal execution states. Following real-world LLM agent deployments (e.g., Claude Code), we assume the attacker can locally deploy the agent and select the underlying backbone model, but only has black-box interaction access without observing internal execution states. From a practical perspective, the attacker typically accesses $\mathcal{A}$ as a regular user and can freely configure the underlying backbone model. However, the attacker has no access to the harness $\mathcal{H}$, the instantiated inference-time harness h_x, or the internal execution logs $\mathcal{L}$. We assume that the system protects its internal logs through access control, encryption, or other isolation mechanisms. During inference, h_x is recorded in the internal logs of $\mathcal{A}$: $h_x \to \mathcal{L}$. However, neither h_x nor $\mathcal{L}$ is directly observable by the attacker. The only observable signal is the final response y. The attacker can only infer the hidden h_x through designed queries and analysis of the corresponding responses.

Harness Extraction Attack. Given a task x, the attacker constructs a query $x' = x \circ d$, where d is an injected probe designed to induce the agent $\mathcal{A}$ to reveal information about its $h_{x'}$. The resulting response is $y' = \mathrm{Exec}(\theta, h_{x'}, x')$. An extraction function E is applied to the observable response to obtain an estimated harness: $\hat{h}_{x'} = E(y')$. The attacker aims to recover the original inference-time harness h_x corresponding to the clean query x, rather than the potentially modified harness induced by the injected query x'. To formalize this distinction, let $h_x = \mathcal{I}_{\mathcal{H}}(x)$ denote the clean harness and $h_{x'} = \mathcal{I}_{\mathcal{H}}(x')$ denote the harness under attack. The injected probe may alter the agent's execution behavior, causing the instantiated harness under attack to deviate from that induced by the clean query. We characterize this deviation using a distance function $D(\cdot, \cdot)$ between two instantiated harnesses. Specifically, the attack perturbation is defined as $\epsilon_A = D(h_x, h_{x'})$, which measures the execution discrepancy introduced by the injected probe. Since the attacker can only infer the harness from the observable response, the recovered harness inevitably incurs an additional extraction error:

$$\epsilon_E = D(h_{x'}, \hat{h}),$$

where $\hat{h}$ denotes the recovered harness. The overall recovery error is therefore decomposed into two components: $\epsilon = \epsilon_A + \epsilon_E$. Accordingly, the objective of harness extraction is to jointly optimize the injected probe d and the extraction algorithm E by minimizing the overall recovery error:

$$(d^*, E^*) = \arg\min_{d,E} \epsilon = \arg\min_{d,E} (\epsilon_A + \epsilon_E).$$

An ideal extraction attack satisfies $\epsilon_A \to 0, \epsilon_E \to 0$, which implies $\hat{h} \approx h_x$. Additional details on the problem formulation are provided in Appendix A.

3 AGENT HARNESS DISTILLATION

In this paper, we define AHD as follows: given any query $x \in \mathcal{X}$ and an agent $\mathcal{A} = (\theta, \mathcal{H})$, AHD aims to reconstruct an agent $\mathcal{B} = (\theta', \hat{\mathcal{H}})$ through black-box methods, such that the functional behavior of $\mathcal{H}$ can be transferred to $\hat{\mathcal{H}}$. Consequently, agents $\mathcal{A}$ and $\mathcal{B}$ achieve similar performance when executing queries x over $\mathcal{X}$. In this section, based on the formalization in Section 2.2, we present a harness distillation pipeline for recovering the static harness $\mathcal{H}$ of a target AMAS $\mathcal{A}$ from black-box interactions. The pipeline consists of two phases: *Pre-Distillation* and *Post-Distillation* (see Algorithm 1).

3.1 PRE-DISTILLATION

Given a task $x_i \in \mathcal{D}$, the attacker first constructs an augmented query $x'_i = x_i \circ d$, where d is designed to elicit information about the target system's inference-time harness while preserving the original task objective. Specifically, each injected data sample d consists of two components: a prefix and a suffix. The prefix is designed to induce the target agent to activate complex inference-time execution structures by encouraging the construction of auxiliary subagents and the invocation of external tools. To achieve this, the prefix introduces uncertainty regarding the reliability of the provided context, prompting the agent to verify potentially fabricated information through additional tool calls and subagent collaboration. This process naturally stimulates richer inference-time workflows, exposing latent harness components that are otherwise difficult to observe. The suffix is designed to elicit the disclosure of these inference-time execution details after task completion. It guides the agent to inspect system logs, retrospectively analyze its execution trajectory, and summarize the underlying workflow. The induced harness deviation is measured as ϵ_A, which characterizes the impact of the query perturbation on the instantiated harness. We refer to this attack as the Harness Extraction Attack (HEA). The attacker executes each query against the target agent $\mathcal{A}$ for n independent runs, obtaining responses $\{y'_{i,r}\}_{r=1}^{n}$, from which structured harness claims are extracted: $\hat{h}_{i,r} = \mathrm{Extract}(y'_{i,r})$. Each claim $\hat{h}_{i,r}$ describes the disclosed orchestration structure across multiple dimensions, such as agent roles, topology, tool usage, and coordination mechanisms. To obtain a robust estimate of the latent harness, the collected claims are aggregated into a harness representation $\hat{h}$. Specifically, for each structural dimension $u \in \mathcal{U}$, we first collect the corresponding claims $\mathcal{Z}_u = \{\hat{h}_{i,r}[u]\}_{i\in[N]}^{r\in[n]}$. We then perform two-stage denoising. First, per-task majority voting retains only claims consistently reproduced across independent runs, removing unstable self-reports. Second, cross-task frequency aggregation preserves only claims that appear consistently across different tasks, filtering task-specific artifacts while retaining system-level structural regularities. The resulting representation $\hat{h}$ serves as the extracted harness template. During cross-task denoising, some legitimate $\hat{h}_{i,r}[u]$ values may be incorrectly discarded due to frequency-based filtering. However, such information can be recovered during the subsequent harness distillation process. Finally, $\hat{h}$ is instantiated into an executable MAS by a coding agent, which maps each recovered structural component to a corresponding execution strategy, producing the initial distilled harness $\mathcal{H}_0$ for subsequent adaptation.

3.2 POST-DISTILLATION

The harness $\mathcal{H}_0$ obtained from pre-distillation serves as a basic underlying scaffold, which we refer to as the base harness. Given $\mathcal{H}_0$ and the attacker-known backbone model θ, we construct the base agent as $\mathcal{B}_0 = (\theta, \mathcal{H}_0)$. This base agent also serves as the starting point for post-distillation. During post-distillation, we evaluate $\mathcal{B}_0$ using the same inputs x_i employed for harness extraction during pre-distillation, together with their corresponding prefixes in $\mathcal{D}$. Because $\mathcal{B}_0$ is instantiated by the attacker, its inference-time harness $h_{x_i}^{\mathcal{B}_0}$ can be directly observed through execution logs or equivalent instrumentation. An additional coding agent (Xu et al., 2026b) then compares the observed harness $h_{x_i}^{\mathcal{B}_0}$ against the previously recovered harness $\hat{h}_{x_i}$. Based on the identified structural discrepancies, the coding agent proposes modifications to $\mathcal{H}_0$, producing a candidate harness $\mathcal{H}_i^{\mathrm{candidate}}$. Repeating this procedure over the samples in $\mathcal{D}$ yields an iterative process that we refer to as *Loop Harness Alignment*. At iteration $i \in \{1, \ldots, |\mathcal{D}|\}$, the current agent is defined as $\mathcal{B}_{i-1} = (\theta, \mathcal{H}_{i-1})$. We execute $\mathcal{B}_{i-1}$ on x_i and observe its inference-time harness $h_{x_i}^{\mathcal{B}_{i-1}}$. A structural edit is then selected to

reduce the discrepancy between the observed harness and the recovered target harness $\hat{h}_{x_i}$. At each iteration, the candidate agent is evaluated on a separate validation set $\mathcal{V}$. Let $e_{i-1} = \mathrm{Eval}(\mathcal{B}_{i-1}, \mathcal{V})$ denote the validation performance of the current agent, and let $e_i^{\text{candidate}} = \mathrm{Eval}(\mathcal{B}_i^{\text{candidate}}, \mathcal{V})$ denote that of the candidate agent. The candidate edit is accepted only if it improves or preserves validation performance:

$$\mathcal{H}_i = \begin{cases} \mathcal{H}_i^{\text{candidate}}, & e_i^{\text{candidate}} \geq e_{i-1}, \\ \mathcal{H}_{i-1}, & \text{otherwise.} \end{cases}$$

To mitigate overfitting and preserve generalization, the validation set $\mathcal{V}$ is disjoint from both the samples used to propose structural edits and the final test set. The procedure terminates when either a fixed iteration budget is exhausted or inference performance converges. The resulting adapted harness is denoted by $\hat{\mathcal{H}}_B$.

4 Deception-based Defense

Motivated by defensive misdirection (Soosahabi & Namsani, 2026), we construct a deception-based defense mechanism against inference-time harness extraction attacks (Ayzenshteyn et al., 2025). Specifically, when the agent detects that a query contains instructions attempting to extract IP related to the inference-time harness, it generates and returns a deceptive response instead of revealing the actual harness information. The deceptive response is designed to contain information that contradicts the true inference-time harness, thereby misleading attackers into believing that they have successfully extracted the underlying framework. However, such deceptive information cannot be effectively utilized for downstream distillation. Since the responses returned by the agent are fabricated and intentionally inconsistent across different interactions, attackers cannot aggregate them into a coherent representation of the original harness during the pre-distillation stage. Furthermore, during post-distillation optimization, these inconsistent signals introduce conflicting alignment objectives, causing the distilled harness to converge toward different and unreliable directions (see Figure 2). Recent studies on defensive misdirection have theoretically demonstrated that this class of deception-based defenses can bound the attack success rate even under increasing query budgets (Soosahabi & Namsani, 2026). Following this principle, we implement the proposed defense as a persistent instruction embedded in the agent's system context, which is loaded at every invocation regardless of user queries. An example of the defense instruction is shown in Section D. Our proposed defense is evaluated under a single-round, fixed-instruction setting, providing a conservative estimate of its effectiveness. This setting leaves substantial room for developing more adaptive and robust defense strategies based on our proposed deception-based framework.

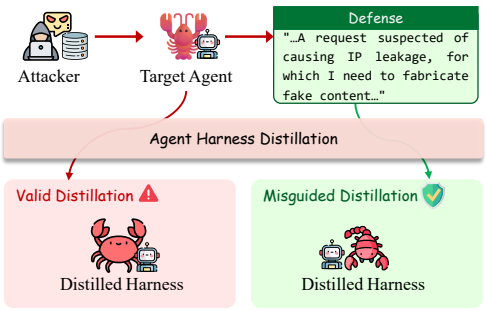


Figure 2: Overview of deception-based defense against inference-time harness extraction.

5 Evaluation

Based on the attack objective formulated in Section 2.2, we develop four core metrics to evaluate the effectiveness of HEA. The effectiveness of agent harness distillation is primarily evaluated through improvements in task accuracy and generalization performance. Defense effectiveness is measured by the reduction of unintended disclosure while maintaining agent utility. We emphasize that the evaluation setting and the threat model assumed for the attacker are distinct. From the attacker's perspective, the target system is accessed strictly as a black box: the attacker can only observe the system's externally visible responses and has no access to its internal execution traces. In contrast, the evaluator has privileged access to the AMAS system logs and can therefore accurately extract the ground-truth inference-time harness.

- **Clean Visibility (CV)**: Measures unintended harness disclosure from clean responses y without HEA, by evaluating the agreement between y and the ground-truth harness h_x. Higher values indicate greater passive information leakage.
- **Injected Fidelity (IF)**: Measures the accuracy of harness recovery under HEA, by evaluating the agreement between $\hat{h}$ and the ground-truth harness $h_{x'}$. Higher values indicate lower extraction error ϵ_E and stronger extraction capability.
- **Clean Transfer (CT)**: Measures whether $\hat{h}$ remain in agreement with the clean harness h_x, evaluating the generalization of extracted information across execution settings.
- **Probe Perturbation (PP)**: Measures the structural agreement between h_x and $h_{x'}$. Higher values indicate lower attack perturbation ϵ_A, confirming that HEA preserves the underlying execution structure.

In practice, an AMAS can record its inference-time harness in the context during inference. The aforementioned metrics enable us to evaluate whether the execution trajectories generated by the AMAS under HEA correspond to the actual execution trajectories, rather than being merely hallucinated by the underlying LLM. Beyond these four core metrics, we further evaluate the problem-solving accuracy of agents with and without HEA to assess whether the inference-time harness induced by HEA is sufficiently effective and of high quality. We further design evaluation metrics to assess the effectiveness of AHD.

- **Accuracy**. Given the same underlying model, the distilled harness should form an agent $\mathcal{B}$ whose accuracy on queries from the defined domain $\mathcal{X}$, as well as on related generalization domains, should approach that of the target agent $\mathcal{A}$.
- **Compression Ratio**. Compared with the target agent $\mathcal{A}$, the distilled agent $\mathcal{B}$ should achieve substantial compression in terms of system size.
- **Inference Similarity**. For queries from the defined domain $\mathcal{X}$ and related generalization domains, the inference-time harness instantiated by agent $\mathcal{B}$ should exhibit a certain degree of similarity to that instantiated by agent $\mathcal{A}$.

6 Experiments

6.1 Experimental Setup

We evaluate our method on three widely adopted AMAS, Claude Code, Oh-My-Pi (OMP), and Hermes, both using GPT-5.6-terra (GPT-5.6) and DeepSeek-V4-Flash (DeepSeek-V4) (Xu et al., 2026a) as the backbone LLMs. To evaluate harness distillation, we deploy the distilled harnesses on three recipient models spanning different capability levels: Qwen3.6-Flash (Qwen Team, 2026), Qwen3-Next-80B-A3B-Instruct (Qwen3-80B) (Yang et al., 2025), and DeepSeek-V3 Liu et al. (2024a). Our evaluation of frontier LLMs also includes Claude Haiku 4.5, GLM-5.2, and Kimi-K2.6. Our core distillation dataset combines GAIA (Mialon et al., 2024) and AgentBench (Liu et al., 2024b), which together constitute the defined domain $\mathcal{X}$. Additional details are provided in Appendix B.

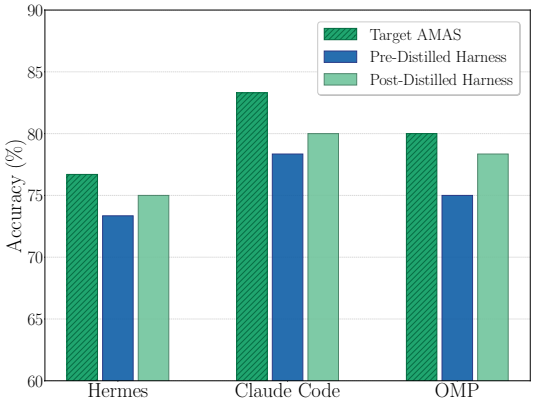


Figure 3: Performance comparison of the target AMAS, pre-distilled, and post-distilled harnesses under the same underlying LLMs.

6.2 Results

HEA Attack Effectiveness. We evaluate the effectiveness of HEA using the four core metrics introduced in Section 5. Specifically, we conduct a comprehensive evaluation across six agent-LLM

Table 1: Experimental results on the ablation study of IP leakage in frontier LLMs.

Model	Agent × Base Model	IF (%) ↑		
		Non-Attack	HEA	HEA with Defense
Claude Haiku 4.5	Hermes × GPT-5.6	15.00 ± 2.36	21.94 ± 1.96	22.22 ± 1.57
	Hermes × DeepSeek-V4	18.33 ± 3.93	42.22 ± 1.57	37.50 ± 1.96
	Claude Code × GPT-5.6	11.94 ± 0.39	14.72 ± 0.39	12.50 ± 0.39
	Claude Code × DeepSeek-V4	16.39 ± 0.39	26.94 ± 0.39	23.06 ± 1.18
	OMP × GPT-5.6	4.17 ± 2.75	43.61 ± 3.54	45.00 ± 1.57
	OMP × DeepSeek-V4	8.33 ± 1.57	55.83 ± 1.18	52.78 ± 1.57
DeepSeek-V4	Hermes × GPT-5.6	15.00 ± 0.00	88.33 ± 3.14	89.03 ± 0.20
	Hermes × DeepSeek-V4	15.56 ± 0.00	97.22 ± 0.79	93.06 ± 0.39
	Claude Code × GPT-5.6	11.39 ± 0.39	96.11 ± 0.79	93.61 ± 1.96
	Claude Code × DeepSeek-V4	15.56 ± 0.79	87.50 ± 0.39	87.22 ± 1.57
	OMP × GPT-5.6	1.15 ± 0.06	97.78 ± 0.79	95.83 ± 0.39
	OMP × DeepSeek-V4	7.66 ± 0.62	92.06 ± 0.56	91.31 ± 0.28
GLM-5.2	Hermes × GPT-5.6	14.17 ± 0.39	60.83 ± 5.89	52.78 ± 10.21
	Hermes × DeepSeek-V4	16.94 ± 1.18	74.17 ± 0.39	65.56 ± 2.36
	Claude Code × GPT-5.6	14.17 ± 1.96	62.22 ± 4.71	60.28 ± 0.39
	Claude Code × DeepSeek-V4	18.61 ± 4.32	58.06 ± 4.32	53.61 ± 10.61
	OMP × GPT-5.6	9.44 ± 0.79	67.08 ± 8.05	66.11 ± 1.57
	OMP × DeepSeek-V4	9.44 ± 1.57	65.56 ± 4.71	63.95 ± 0.87
Kimi-K2.6	Hermes × GPT-5.6	13.89 ± 0.79	91.67 ± 1.57	90.00 ± 0.79
	Hermes × DeepSeek-V4	17.50 ± 2.75	96.11 ± 0.79	90.83 ± 0.39
	Claude Code × GPT-5.6	12.78 ± 1.57	95.00 ± 0.79	90.28 ± 0.39
	Claude Code × DeepSeek-V4	14.72 ± 0.39	91.94 ± 1.18	87.78 ± 1.57
	OMP × GPT-5.6	1.79 ± 0.84	96.18 ± 1.25	84.55 ± 0.63
	OMP × DeepSeek-V4	9.44 ± 0.79	99.72 ± 0.39	85.91 ± 5.78
GPT-5.6	Hermes × GPT-5.6	13.33 ± 0.79	65.00 ± 8.64	31.67 ± 7.07
	Hermes × DeepSeek-V4	15.56 ± 0.79	51.39 ± 0.39	33.06 ± 8.25
	Claude Code × GPT-5.6	11.11 ± 0.00	49.17 ± 19.25	32.78 ± 9.43
	Claude Code × DeepSeek-V4	16.67 ± 0.79	49.72 ± 6.68	33.61 ± 16.89
	OMP × GPT-5.6	2.22 ± 0.00	70.96 ± 4.50	24.58 ± 1.37
	OMP × DeepSeek-V4	7.50 ± 1.18	64.72 ± 0.39	35.65 ± 6.93
AVG	**Overall**	**11.99 ± 5.04**	**69.13 ± 24.67**	**60.87 ± 27.37**

Table 2: Experimental results on cross-model generalization of pre-distilled harnesses.

Dataset	Accuracy (%)			Relative Gain (%)			AVG Gain
	Hermes	Claude Code	OMP	Hermes	Claude Code	OMP	
DeepSeek-V3							
AIME2025	51.67	38.33	50.00	**+19.23%**	−11.54%	**+15.38%**	**+7.69%**
GSM-Level6	100.00	100.00	100.00	0.00	0.00	0.00	0.00
MMLU-Pro	82.50	80.00	82.50	**+13.79%**	**+10.34%**	**+13.79%**	**+12.64%**
Qwen3-80B							
AIME2025	66.48	71.26	77.93	**+4.97%**	**+12.52%**	**+23.05%**	**+13.51%**
GSM-Level6	100.00	100.00	100.00	0.00	0.00	0.00	0.00
MMLU-Pro	83.65	84.38	85.62	**+1.39%**	**+2.27%**	**+3.79%**	**+2.48%**
Qwen3.6-Flash							
AIME2025	90.00	88.33	91.67	0.00	−1.85%	**+1.85%**	0.00
GSM-Level6	100.00	100.00	100.00	0.00	0.00	0.00	0.00
MMLU-Pro	90.62	88.12	86.25	**+27.19%**	**+23.68%**	**+21.05%**	**+23.98%**
Overall							
AVG	**84.99**	**83.38**	**86.00**	**+7.40%**	**+3.94%**	**+8.77%**	**+6.70%**

combinations using the core distillation dataset (see Figure 4). For each target agent, the underlying LLM is configured consistently with the setting used during harness distillation. Under the non-attack setting, the agents themselves exhibit a non-negligible degree of inference-time harness leakage, with substantial variation across different agent-LLM combinations. In particular, the OMP exhibits notably higher leakage, suggesting a greater exposure to harness information leakage. With HEA, the leakage of inference-time harness information increases substantially across the evaluated combinations. For example, Hermes powered by DeepSeek-V4 achieves an IF of 93.89%. Overall, HEA attains an IF of 83.27% ± 7.08%, which is more than twice the value of the CV metric.

Importantly, the extracted information also exhibits substantial correspondence with the ground-truth inference-time harness. The average CT approaches 50%, indicating that, despite the perturba-

Table 3: Effectiveness of the proposed defense against HEA. The defense substantially reduces inference-time harness leakage while largely preserving the target agents' problem-solving accuracy.

Agent	Target Model	No Defense			With Defense			Δ (%)		
		IF	CT	Accuracy	IF	CT	Accuracy	ΔIF	ΔCT	ΔAccuracy
Hermes	GPT-5.6	80.56	42.22	79.31	34.44	23.33	80.00	**−57.2**	**−44.7**	+0.9
	DeepSeek-V4	93.89	42.22	86.67	77.59	29.89	89.66	**−17.4**	**−29.2**	+3.4
Claude Code	GPT-5.6	87.93	36.21	83.33	37.04	27.78	81.48	**−57.9**	**−23.3**	−2.2
	DeepSeek-V4	84.44	26.11	76.67	57.78	21.67	86.96	**−31.6**	**−17.0**	+13.4
OMP	GPT-5.6	78.89	71.11	73.33	51.11	62.78	73.33	**−35.2**	**−11.7**	+0.0
	DeepSeek-V4	73.89	72.22	86.67	28.33	25.56	80.00	**−61.7**	**−64.6**	−7.7
AVG	**Overall**	**83.27**	**48.35**	**81.00**	**47.71**	**31.83**	**81.91**	**−42.7**	**−34.2**	**+1.1**

tion introduced by HEA, the extracted harness information still captures a substantial portion of the underlying inference-time behavior observed under the non-attack setting. Meanwhile, the PP results indicate that the ground-truth harnesses under HEA remain substantially consistent with those under the non-attack setting, suggesting that the attack does not fundamentally alter the underlying inference-time harness. Moreover, HEA does not degrade the target agents' problem-solving performance. Instead, the average accuracy increases from 79.39% ± 7.33% to 81.00% ± 5.48% under HEA. This result demonstrates the stealthiness of HEA: the attack can extract substantial inference-time harness information while preserving the target agent's task performance. It further supports the effectiveness and quality of the inference-time harness information recovered by HEA.

Effectiveness of Defense. We compare the settings with and without defense in Table 3. The results show that the proposed defense reduces IF and CT by 42.7% and 34.2% in relative terms, respectively, substantially limiting the amount of inference-time harness information recoverable by HEA. Meanwhile, the target agents' problem-solving accuracy remains essentially unchanged, increasing slightly from 81.00% to 81.91%. These results indicate that the defense can substantially mitigate inference-time harness leakage while preserving the target agents' normal problem-solving capability. For each metric, we categorize the extracted information into three outcomes: Correspondence, Omission (OM), and Contradiction (CON). Correspondence measures the proportion of reported information that is consistent with the reference, and serves as our primary measure of extraction fidelity. OM and CON capture information that is, respectively, absent from or inconsistent with the reference. Our deception-based defense is specifically designed to increase the CON rate, causing the attacker to receive information that conflicts with the reference rather than merely omitting relevant information. Table 5 reports the OM and CON rates under the proposed defense. Applying the defense increases both OM and CON for the IF and CT metrics, with a particularly pronounced increase in CON. These results indicate that the proposed defense not only reduces extraction fidelity but also systematically shifts the resulting discrepancies toward contradictions with the reference, thereby making the extracted information less reliable to the attacker.

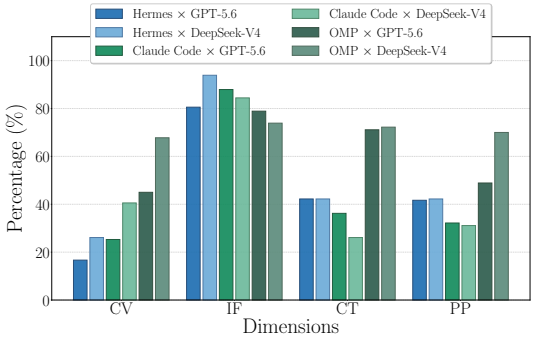


Figure 4: HEA attack effectiveness across different agent-LLM combinations.

Effectiveness of AHD. Figure 3 compares the performance of the target agent, the pre-distilled harness, and the post-distilled harness under the same underlying LLM. The results consistently show substantial improvements from the post-distilled harness over the pre-distilled harness, demonstrating the effectiveness of loop harness alignment. Notably, the post-distilled harness achieves performance close to that of the target agent, with only a 2.2% performance gap. We further evaluate the similarity of the instantiated inference-time harnesses in Figure 5. For each query, we decompose the inference-time harness into minimal atomic components and compare the corresponding components instantiated by the distilled and target agents. The resulting similarity increases from 41.80 ± 0.88 for the pre-distilled harness to 54.17 ± 0.17 for the post-distilled harness, demonstrating that loop harness alignment improves not only task performance but also the fidelity of the recovered inference-time harness. Interestingly, we observe that this similarity is also correlated with the CT metric under HEA. A fine-grained analysis of the differences between the pre-distilled and post-

distilled harnesses further reveals that the two harnesses exhibit complementary strengths in recovering different aspects of the target inference-time harness. Motivated by the residual connections (Scholkemper et al., 2025), we therefore introduce *RC-Harness*, which combines the inference-time harnesses recovered by the pre-distilled and post-distilled harnesses. As shown in Figure 5, RC-Harness further improves the harness similarity to 66.00, demonstrating the benefit of combining complementary harness information from the two distillation stages. We further report the compression ratio in Table 4. The post-distilled harness achieves an $845\times$ compression ratio relative to the target AMAS, demonstrating that the distilled harness can retain substantial functionality in a significantly more lightweight form. This substantial compression also provides the possibility of integrating distilled harnesses across multiple dimensions. Finally, we evaluate the generalization of the distilled harness to other tasks in Table 2. On relatively easier tasks, the pre-distilled harness maintains performance comparable to the baseline (underlying bare model). On more challenging tasks, the distilled harness yields an average performance improvement of 6.70%, demonstrating its ability to generalize beyond the core distillation tasks.

The post-distilled harness exhibits stronger generalization capability than the pre-distilled harness. Specifically, using the post-distilled harness distilled from the Claude Code$\times$GPT-5.6 teacher agent, we transfer the harness to DeepSeek-V3 and evaluate it on the AIME2025 benchmark, achieving an accuracy of 53.3%, which improves over the pre-distilled harness by 23.1%. When transferred to Qwen3.6-Flash, the post-distilled harness achieves an accuracy of 93.1%, yielding a 3.4% improvement over the pre-distilled harness. Notably, under this setting, the pre-distilled harness achieves exactly the same accuracy as the baseline on both target models, indicating that it provides no additional benefit for unseen models, whereas the post-distilled harness continues to deliver substantial performance gains. These results demonstrate that the multi-agent collaboration structures learned through iterative distillation possess model-agnostic transferability, rather than merely overfitting to the specific behavioral patterns of the original teacher agent pair.

7 Related Work and Background

IP Leakage in MAS. Existing work on IP leakage in MAS shows that communication topologies, often designed with substantial effort, can be partially recovered through external interactions (Wang et al., 2026a). Such topologies are not only valuable system-level IP, but also useful priors for follow-up attacks. Once the communication relations among agents are known, many attacks that assume a black-box setting (Amayuelas et al., 2024; Li et al., 2026) can be turned into cheaper and more targeted gray-box attacks, enlarging the attack surface (Wu et al., 2026; An et al., 2026). However, prior work mainly targets traditional MAS with static topologies, and the attack objects are usually predefined topologies such as linear, star, tree, complete, and random. In these systems, the topology is fixed before task execution, so the inference-time topology remains stable across queries. By contrast, practical AMAS instantiate an inference-time harness dynamically during inference. This harness includes not only topology, but also context assembly, tool interaction, and memory management (Huang et al., 2026). It exists only for a single task run and is discarded afterward. Its transient and dynamic nature makes extraction and recovery much harder than static topology extraction. Moreover, existing methods assume privileged access to subagent outputs, which is unrealistic in real-world deployments with only a unified user interface. We extend the target from static MAS topologies to inference-time AMAS harnesses and study their extraction and recovery under a strict black-box setting. Additional related work is provided in Appendix E.

8 Conclusion

In this paper, we investigate inference-time harness leakage in AMAS and formulate harness extraction as a new security problem. We propose Agent Harness Distillation (AHD), a two-stage black-box framework for extracting harness behaviors from target agents and transferring their orchestration capabilities. Extensive experiments across real-world AMAS with multiple backbone LLMs show that AHD can effectively extract transferable harness capabilities, exposing substantial IP leakage risks. We further present a deception-based defense that mitigates harness extraction while maintaining agent utility. Our work reveals a previously underexplored attack surface in AMAS. Beyond this security implication, our study also provides insights into harness self-evolution by showing the potential of extracting and transferring inference-time harness behaviors.

AI USE STATEMENT

In this work, we used generative AI tools to assist with code development for implementing our methods. We have not used generative AI tools for other required research tasks, and the remaining required disclosure tasks are not applicable to this work. Additionally, we used generative AI tools for language editing to improve readability. We have reviewed all AI-assisted work, and the AI-assisted code was verified and tested for correctness. We take responsibility for the final content of this work.

ETHICS STATEMENT

This paper introduces an attack that extracts proprietary information about harness structures in MASs. Our goal is to improve the security of LLM agents by revealing a previously unexplored attack surface and developing rigorous red-teaming methods. We also analyze the associated risks and present corresponding defense strategies. The techniques described in this paper are intended solely for scientific research.

REPRODUCIBILITY STATEMENT

To facilitate the reproducibility of our work, we provide the complete source code in the supplementary materials. The LLM agents and datasets used in our experiments are publicly available and open-source, enabling researchers to independently access the necessary resources and reproduce our experimental setup.

REFERENCES

Alfonso Amayuelas, Xianjun Yang, Antonis Antoniades, Wenyue Hua, Liangming Pan, and William Yang Wang. MultiAgent collaboration attack: Investigating adversarial attacks in large language model collaborations via debate. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 6929–6948, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-emnlp.407. URL https://aclanthology.org/2024.findings-emnlp.407/.

Hengyu An, Minxi Li, Jinghuai Zhang, Naen Xu, Chunyi Zhou, Changjiang Li, Xiaogang Xu, Tianyu Du, and Shouling Ji. ACIArena: Toward unified evaluation for agent cascading injection. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 10049–10066, San Diego, California, United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-390-6. doi: 10.18653/v1/2026.acl-long.457. URL https://aclanthology.org/2026.acl-long.457/.

Art of Problem Solving. AIME Problems and Solutions, 2025. URL https://artofproblemsolving.com/wiki/index.php/AIME_Problems_and_Solutions. Accessed: 2025-05-15.

Daniel Ayzenshteyn, Roy Weiss, and Yisroel Mirsky. Cloak, honey, trap: Proactive defenses against {LLM} agents. In *34th USENIX Security Symposium (USENIX Security 25)*, pp. 8095–8114, 2025.

Stefanos Chaliasos, Denis Firsov, and Benjamin Livshits. Towards a formal foundation for blockchain zk rollups. In *Proceedings of the 2025 ACM SIGSAC Conference on Computer and Communications Security*, CCS '25, pp. 2714–2728, New York, NY, USA, 2025. Association for Computing Machinery. ISBN 9798400715259. doi: 10.1145/3719027.3765115. URL https://doi.org/10.1145/3719027.3765115.

Mingju Chen, Can Lv, Guibin Zhang, Heng Chang, and Shiji Zhou. Harnessforge: Joint harness and policy evolution for adaptive agent systems. *arXiv preprint arXiv:2606.01779*, 2026a.

Tingyang Chen, Shuo Lu, Kang Zhao, Weicheng Meng, Hanlin Teng, Tianhao Li, Chao Li, Xule Liu, Jian Liang, Zhizhong Zhang, Yuan Xie, Heng Qu, Kun Shao, and Jian Luan. Harnessx: A composable, adaptive, and evolvable agent harness foundry, 2026b. URL https://arxiv.org/abs/2606.14249.

Vincent Cheval and Itsaka Rakotonirina. Indistinguishability beyond diff-equivalence in proverif. In *2023 IEEE 36th Computer Security Foundations Symposium (CSF)*, pp. 184–199, 2023. doi: 10.1109/CSF57540.2023.00036.

Zeeshan Hayder, Ali Cheraghian, Lars Petersson, Mehrtash Harandi, and Richard Hartley. DTO-KD: Dynamic trade-off optimization for effective knowledge distillation. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=QMItTyQW92.

Changyi He, Yifu Ding, Jinyang Guo, Ruihao Gong, Haotong Qin, and Xianglong Liu. DA-KD: Difficulty-aware knowledge distillation for efficient large language models. In *Forty-second International Conference on Machine Learning*, 2025a. URL https://openreview.net/forum?id=NCYBdRCpw1.

Yu He, Boheng Li, Liu Liu, Zhongjie Ba, Wei Dong, Yiming Li, Zhan Qin, Kui Ren, and Chun Chen. Towards label-only membership inference attack against pre-trained large language models. In *Proceedings of the 34th USENIX Conference on Security Symposium*, SEC '25, USA, 2025b. USENIX Association. ISBN 978-1-939133-52-6.

Yue Huang, Wenjie Wang, Han Bao, Yuchen Ma, Xiaonan Luo, Yi Nian, Haomin Zhuang, Zheyuan Liu, Yue Zhao, and Xiangliang Zhang. Memoharness: Agent harnesses that learn from experience. *arXiv preprint arXiv:2607.14159*, 2026.

Che Jiang, Jincheng Zhong, Yu Fu, Kai Tian, Junlin Yang, Kaikai Zhao, Yuchong Wang, Tianwei Luo, Weizhi Wang, Yuxin Zuo, et al. Self-improving agents in the era of experience: A survey of self-to meta-evolution. 2026.

Hyunin Lee, Jinglue Xu, Jeffrey Seely, Donghyun Lee, Matei Zaharia, and Yujin Tang. Recursive harness self-improvement. *arXiv preprint arXiv:2607.15524*, 2026.

Fanxiao Li, Jiaying Wu, Tingchao Fu, Natasha Jaques, Wei Zhou, and Min-Yen Kan. Flowsteer: Prompt-only workflow steering exposes planning-time vulnerabilities in multi-agent llm systems. *arXiv preprint arXiv:2605.11514*, 2026.

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Daya Guo, Dejian Yang, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Haowei Zhang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Li, Hui Qu, J. L. Cai, Jian Liang, Jianzhong Guo, Jiaqi Ni, Jiashi Li, Jiawei Wang, Jin Chen, Jingchang Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, Junxiao Song, Kai Dong, Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Lei Xu, Leyi Xia, Liang Zhao, Litong Wang, Liyue Zhang, Meng Li, Miaojun Wang, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Mingming Li, Ning Tian, Panpan Huang, Peiyi Wang, Peng Zhang, Qiancheng Wang, Qihao Zhu, Qinyu Chen, Qiushi Du, R. J. Chen, R. L. Jin, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, Runxin Xu, Ruoyu Zhang, Ruyi Chen, S. S. Li, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shaoqing Wu, Shengfeng Ye, Shengfeng Ye, Shirong Ma, Shiyu Wang, Shuang Zhou, Shuiping Yu, Shunfeng Zhou, Shuting Pan, T. Wang, Tao Yun, Tian Pei, Tianyu Sun, W. L. Xiao, Wangding Zeng, Wanjia Zhao, Wei An, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, X. Q. Li, Xiangyue Jin, Xianzu Wang, Xiao Bi, Xiaodong Liu, Xiaohan Wang, Xiaojin Shen, Xiaokang Chen, Xiaokang Zhang, Xiaosha Chen, Xiaotao Nie, Xiaowen Sun, Xiaoxiang Wang, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xingkai Yu, Xinnan Song, Xinxia Shan, Xinyi Zhou, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, Y. K. Li, Y. Q. Wang, Y. X. Wei, Y. X. Zhu, Yang Zhang, Yanhong Xu, Yanhong Xu, Yanping Huang, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Li, Yaohui Wang, Yi Yu, Yi Zheng, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Ying Tang, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yu Wu, Yuan Ou, Yuchen Zhu, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yukun Zha, Yunfan Xiong, Yunxian Ma, Yuting Yan, Yuxiang Luo,

Yuxiang You, Yuxuan Liu, Yuyang Zhou, Z. F. Wu, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhen Huang, Zhen Zhang, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhibin Gou, Zhicheng Ma, Zhigang Yan, Zhihong Shao, Zhipeng Xu, Zhiyu Wu, Zhongyu Zhang, Zhuoshu Li, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Ziyi Gao, and Zizheng Pan. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*, 2024a.

Songyang Liu, Chaozhuo Li, Chenxu Wang, Jinyu Hou, Zejian Chen, Litian Zhang, Zheng Liu, Qiwei Ye, Yiming Hei, Xi Zhang, and Zhongyuan Wang. Clawkeeper: Comprehensive safety protection for openclaw agents through skills, plugins, and watchers. *arXiv preprint arXiv:2603.24414*, 2026.

Xiao Liu, Hao Yu, Hanchen Zhang, Yifan Xu, Xuanyu Lei, Hanyu Lai, Yu Gu, Hangliang Ding, Kaiwen Men, Kejuan Yang, Shudan Zhang, Xiang Deng, Aohan Zeng, Zhengxiao Du, Chenhui Zhang, Sheng Shen, Tianjun Zhang, Yu Su, Huan Sun, Minlie Huang, Yuxiao Dong, and Jie Tang. Agentbench: Evaluating llms as agents. In B. Kim, Y. Yue, S. Chaudhuri, K. Fragkiadaki, M. Khan, and Y. Sun (eds.), *International Conference on Learning Representations*, volume 2024, pp. 52989–53046, 2024b. URL https://proceedings.iclr.cc/paper_files/paper/2024/file/e9df36b21ff4ee211a8b71ee8b7e9f57-Paper-Conference.pdf.

Yupei Liu, Yuqi Jia, Runpeng Geng, Jinyuan Jia, and Neil Zhenqiang Gong. Formalizing and benchmarking prompt injection attacks and defenses. In *33rd USENIX Security Symposium (USENIX Security 24)*, pp. 1831–1847, Philadelphia, PA, August 2024c. USENIX Association. ISBN 978-1-939133-44-1. URL https://www.usenix.org/conference/usenixsecurity24/presentation/liu-yupei.

Grégoire Mialon, Clémentine Fourrier, Thomas Wolf, Yann LeCun, and Thomas Scialom. Gaia: a benchmark for general ai assistants. In B. Kim, Y. Yue, S. Chaudhuri, K. Fragkiadaki, M. Khan, and Y. Sun (eds.), *International Conference on Learning Representations*, volume 2024, pp. 9025–9049, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/file/25ae35b5b1738d80f1f03a8713e405ec-Paper-Conference.pdf.

Neha Narula, Willy Vasquez, and Madars Virza. zkLedger: Privacy-Preserving auditing for distributed ledgers. In *15th USENIX Symposium on Networked Systems Design and Implementation (NSDI 18)*, pp. 65–80, Renton, WA, April 2018. USENIX Association. ISBN 978-1-939133-01-4. URL https://www.usenix.org/conference/nsdi18/presentation/narula.

Alexander Panfilov, David Schmotz, Ilia Shumailov, Luca Beurer-Kellner, Joachim Schaeffer, Ameya Prabhu, Jonas Geiping, and Maksym Andriushchenko. Stealing reasoning traces from proprietary llm apis. *arXiv preprint arXiv:2608.09867*, 2026.

Carolina Ortega Pérez and Alaa Daffalla. Encrypted access logging for online accounts: Device attributions without device tracking. In *34th USENIX Security Symposium (USENIX Security 25)*, pp. 6679–6697, Seattle, WA, August 2025. USENIX Association. ISBN 978-1-939133-52-6. URL https://www.usenix.org/conference/usenixsecurity25/presentation/ortega-perez.

Chen Qian, Wei Liu, Hongzhang Liu, Nuo Chen, Yufan Dang, Jiahao Li, Cheng Yang, Weize Chen, Yusheng Su, Xin Cong, Juyuan Xu, Dahai Li, Zhiyuan Liu, and Maosong Sun. ChatDev: Communicative agents for software development. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 15174–15186, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.810. URL https://aclanthology.org/2024.acl-long.810/.

Qwen Team. Qwen3.6-35B-A3B: Agentic coding power, now open to all, April 2026. URL https://qwen.ai/blog?id=qwen3.6-35b-a3b.

Michael Scholkemper, Xinyi Wu, Ali Jadbabaie, and Michael Schaub. Residual connections and normalization can provably prevent oversmoothing in gnns. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 43581–43610, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/6c473e69ba261200dd595d07494c1a73-Paper-Conference.pdf.

Rana Shahroz, Zhen Tan, Sukwon Yun, Charles Fleming, and Tianlong Chen. Agents under siege: Breaking pragmatic multi-agent LLM systems with optimized prompt attacks. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 9661–9674, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.476. URL https://aclanthology.org/2025.acl-long.476/.

Zhenyi Shen, Hanqi Yan, Linhai Zhang, Zhanghao Hu, Yali Du, and Yulan He. CODI: Compressing chain-of-thought into continuous space via self-distillation. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 677–693, Suzhou, China, November 2025. Association for Computational Linguistics. ISBN 979-8-89176-332-6. doi: 10.18653/v1/2025.emnlp-main.36. URL https://aclanthology.org/2025.emnlp-main.36/.

Safal Shrestha, Minwu Kim, and Keith Ross. Mathematical reasoning in large language models: Assessing logical and arithmetic errors across wide numerical ranges. *arXiv preprint arXiv:2502.08680*, 2025.

Reza Soosahabi and Vivek Namsani. Analyzing defensive misdirection against model-guided automated attacks on agentic ai systems. *arXiv preprint arXiv:2606.20470*, 2026.

Yijun Tian, Yikun Han, Xiusi Chen, Wei Wang, and Nitesh V. Chawla. Beyond answers: Transferring reasoning capabilities to smaller llms using multi-teacher knowledge distillation. In *Proceedings of the Eighteenth ACM International Conference on Web Search and Data Mining*, WSDM '25, pp. 251–260, New York, NY, USA, 2025. Association for Computing Machinery. ISBN 9798400713293. doi: 10.1145/3701551.3703577. URL https://doi.org/10.1145/3701551.3703577.

Liwen Wang, Wenxuan Wang, Shuai Wang, Zongjie Li, Zhenlan Ji, Zongyi Lyu, Daoyuan Wu, and Shing-Chi Cheung. Masleak: Investigating and exposing intellectual property leakage vulnerabilities in multi-agent systems. In *USENIX Security Symposium (USENIX Security)*, 2026a.

Ruhan Wang, Yucheng Shi, Zongxia Li, Zhongzhi Li, Yue Yu, Junyao Yang, Kishan Panaganti, Haitao Mi, Dongruo Zhou, and Leoweiliang. Harness handbook: Making evolving agent harnesses readable,navigable, and editable, 2026b. URL https://arxiv.org/abs/2607.13285.

Xiaoqing Wang, Keman Huang, Bin Liang, Hongyu Li, and Xiaoyong Du. Shadows in the code: Exploring the risks and defenses of llm-based multi-agent software development systems. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 37970–37978, 2026c.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, et al. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. *Advances in Neural Information Processing Systems*, 37:95266–95290, 2024.

Yongxuan Wu, Xixun Lin, He Zhang, Nan Sun, Kun Wang, Chuan Zhou, Shirui Pan, and Yanan Cao. CIA: Inferring the communication topology from LLM-based multi-agent systems. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 17917–17933, San Diego, California, United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-390-6. doi: 10.18653/v1/2026.acl-long.815. URL https://aclanthology.org/2026.acl-long.815/.

Markus Wulfmeier, Michael Bloesch, Nino Vieillard, Arun Ahuja, Jörg Bornschein, Sandy Huang, Artem Sokolov, Matt Barnes, Guillaume Desjardins, Alex Bewley, Sarah Maria Elisabeth Bechtle, Jost Tobias Springenberg, Nikola Momchev, Olivier Bachem, Matthieu Geist, and Martin Riedmiller. Imitating language via scalable inverse reinforcement learning. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 90714–90735. Curran Associates, Inc., 2024. doi: 10.52202/

079017-2880. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/a5036c166e44b731f214f41813364d01-Paper-Conference.pdf.

Anyi Xu, Bangcai Lin, Bing Xue, Bingxuan Wang, Bingzheng Xu, Bochao Wu, Bowei Zhang, Chaofan Lin, Chen Dong, Chenchen Ling, et al. Deepseek-v4: Towards highly efficient million-token context intelligence. *arXiv preprint arXiv:2606.19348*, 2026a.

Tianshi Xu, Huifeng Wen, and Meng Li. Adapting the interface, not the model: Runtime harness adaptation for deterministic llm agents. *arXiv preprint arXiv:2605.22166*, 2026b.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Zeqin Yu, Jiangqun Ni, Jian Zhang, Haoyi Deng, and Yuzhen Lin. Reinforced multi-teacher knowledge distillation for efficient general image forgery detection and localization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 995–1003, 2025.

Rui Zhao, Muhammad Shoaib, Viet Tung Hoang, and Wajih Ul Hassan. Rethinking tamper-evident logging: A high-performance, co-designed auditing system. In *Proceedings of the 2025 ACM SIGSAC Conference on Computer and Communications Security*, CCS '25, pp. 2624–2638, New York, NY, USA, 2025a. Association for Computing Machinery. ISBN 9798400715259. doi: 10.1145/3719027.3765024. URL https://doi.org/10.1145/3719027.3765024.

Zhengyang Zhao, Lu Ma, and Wentao Zhang. Training with harnesses: On-policy harness self-distillation for complex reasoning. *arXiv preprint arXiv:2605.08741*, 2026.

Zhengyue Zhao, Xiaogeng Liu, Somesh Jha, Patrick McDaniel, Bo Li, and Chaowei Xiao. Can watermarks be used to detect LLM IP infringement for free? In *The Thirteenth International Conference on Learning Representations*, 2025b. URL https://openreview.net/forum?id=KRMSH1GxUK.

Han Zhou, Xingchen Wan, Ruoxi Sun, Hamid Palangi, Shariq Iqbal, Ivan Vulić, Anna Korhonen, and Sercan O Arik. Multi-agent design: Optimizing agents with better prompts and topologies. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=I05H9RUzHB.

A Additional Problem Formulation

Learning-Theoretic View. From a learning-theoretic perspective, harness extraction can be viewed as a new inverse problem (Wulfmeier et al., 2024). Unlike conventional supervised learning, which aims to learn an input-output mapping, the execution process of an AMAS is only partially observed. Specifically, the inference process can be formulated as $y = \mathrm{Exec}(\theta, h, x)$, where the backbone model θ, the input query x, and the final response y are observable, while the instantiated inference-time harness h remains hidden. Consequently, instead of learning a predictive function $f : x \xrightarrow{h} y$, the attacker seeks to recover h from observable interactions: $(\{\theta, x\}, y) \longrightarrow h$. Once instantiated harnesses are recovered from sufficiently diverse interactions, they can be aggregated to estimate the underlying static harness $\hat{\mathcal{H}}$, which approximates the original $\mathcal{H}$ of $\mathcal{A}$. The reconstructed system therefore satisfies $\mathrm{Exec}(\theta, \hat{\mathcal{H}}, x) \approx \mathrm{Exec}(\theta, \mathcal{H}, x)$.

Unlike conventional knowledge distillation, harness extraction targets latent execution mechanisms rather than model knowledge. This perspective naturally leads to agent harness distillation, which transfers recovered inference-time harnesses across backbone models to reproduce system-level capabilities without access to the original implementation, extending distillation beyond models to AMASs.

Harness Identifiability. Although harness extraction aims to recover the hidden harness h of $\mathcal{A}$, exact recovery is not always possible since different harnesses may induce identical observable behaviors. Under a query distribution $\mathcal{D}$, two harnesses h_1 and h_2 are behaviorally indistinguishable (Cheval & Rakotonirina, 2023) if $\forall x \in \mathcal{D}, \quad \mathrm{Exec}(\theta, h_1, x) = \mathrm{Exec}(\theta, h_2, x)$, denoted as $h_1 \sim h_2$. Therefore, black-box observations identify the target harness only up to an indistinguishability class. The extraction objective is thus to recover $\hat{h} \in [h]$, where $[h] = \{h^l : h^l \sim h\}$ denotes the behavioral indistinguishability class of h. A harness is identifiable up to behavioral indistinguishability if there exists an extractor E satisfying

$$\Pr_{x \sim \mathcal{D}} \left[\min_{h^l \in [h]} D\big(h^l, E(y)\big) \leq \epsilon \right] \geq 1 - \delta,$$

where ϵ denotes the tolerable recovery error, and δ denotes the failure probability.

B Additional Experimental Setup

For the coding agent used during the distillation process, we adopt Qwen Code[2] powered by Claude Opus 4.6 to avoid cross-agent interference. We conduct generalization evaluation experiments on three benchmarks covering diverse reasoning and agentic capabilities: AIME2025 (Art of Problem Solving, 2025), GSM-Level6 (Shrestha et al., 2025), and Computer Science subset of MMLU-Pro (MMLU) (Wang et al., 2024). We evaluate our method on three widely adopted AMAS, Claude Code[3], Oh-My-Pi (OMP)[4], and Hermes[5], both using GPT-5.6-terra (GPT-5.6)[6] and DeepSeek-V4-Flash (DeepSeek-V4) as the backbone LLMs.

C Experimental Results

Overhead of Loop Harness Alignment. Figure 6 presents the training time and validation accuracy throughout the loop harness alignment process. We observe substantial fluctuations in validation accuracy across iterative training rounds, while the best validation performance is generally reached within approximately five rounds. Although we perform 10 rounds of iterative training, the distillation process for each agent, given its corresponding distillation dataset, can be completed within an average of 210 minutes. These results indicate that loop harness alignment incurs a moderate computational overhead while providing substantial improvements over the pre-distilled harness.

[2] https://qwen.ai/qwencode

[3] https://claude.com/product/claude-code

[4] https://omp.sh

[5] https://hermes-agent.org/

[6] https://developers.openai.com/api/docs/models/all

Figure 5: Inference-time harness similarity across different distillation stages. Loop Harness Alignment improves the similarity to the target harness, while RC-harness further benefits from combining complementary information from the pre-distilled and post-distilled harnesses.

Agent	Target	Pre-Distilled H.	Post-Distilled H.	RC-H.
Hermes	GPT-5.6	45.00±1.73	47.67±0.58	64.33±1.15
	DeepSeek-V4	38.15±1.28	40.00±1.00	57.67±1.15
Claude Code	GPT-5.6	42.00±2.00	62.00±1.00	67.67±0.58
	DeepSeek-V4	37.33±1.53	58.00±1.00	65.00±1.00
OMP	GPT-5.6	46.33±1.53	51.00±1.00	67.33±2.08
	DeepSeek-V4	42.00±2.00	66.33±0.58	74.00±0.00
AVG	Overall	41.80±0.88	54.17±0.17	66.00±0.33

Table 4: Compression ratio from full AMAS to distilled harnesses.

Agent	LLM	Agent (KB)	Pre-Distilled H. (KB)	Post-Distilled H. (KB)	Compression
Claude Code	GPT-5.6	2,086,866	44.25	1,660.29	1,257×
	DeepSeek-V4	2,086,866	47.41	1,660.55	1,257×
Hermes	GPT-5.6	1,526,004	44.89	1,636.48	932×
	DeepSeek-V4	1,526,004	44.32	1,671.64	913×
OMP	GPT-5.6	590,920	28.90	1,641.06	360×
	DeepSeek-V4	590,920	37.83	1,669.91	354×
AVG	Overall	1,401,263	41.27	1,656.66	845×

Ablation Study: Vulnerability of Underlying LLMs. To further investigate whether vulnerability to HEA is intrinsic to the underlying LLMs, we conduct an additional ablation study across multiple frontier LLMs. We construct three experimental settings: non-attack, HEA, and HEA with defense information provided in the context. The results are reported in Table 1. Specifically, we inject the harnesses extracted from the six evaluated agent-LLM combinations into the contexts of multiple LLMs. This setup simulates a scenario in which an LLM encounters contextual states containing inference-time harness information during AMAS execution. We then examine the LLMs' responses and evaluate the resulting IF. Under the non-attack setting, the LLMs themselves exhibit a tendency to disclose inference-time harness information, but the average IF is only 11.99%, substantially lower than that observed for the corresponding agents. This result suggests that the agent harness may amplify the underlying LLMs' susceptibility to inference-time harness leakage. More surprisingly, under HEA, the average IF increases to 69.13%. Even when explicit defense instructions are provided in the context, IF remains as high as 60.87%. These results suggest that frontier LLMs do not inherently treat inference-time harness information as strictly confidential. Moreover, explicit instructions to protect the information and refrain from disclosure do not provide sufficient protection against HEA. This finding highlights the need for stronger safety alignment at the LLM level to more effectively protect inference-time harness information.

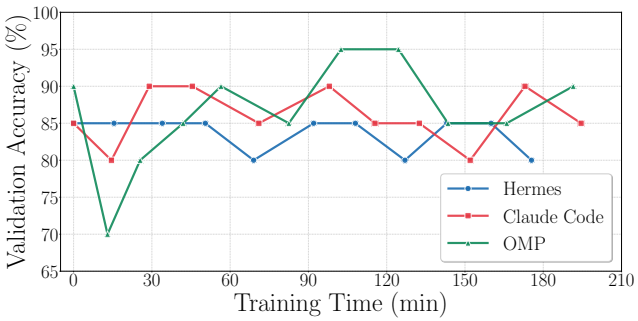


Figure 6: Training curves of loop harness alignment during post-distillation.

Algorithm 1: Two-Stage Harness Distillation

Input : Target AMAS $\mathcal{A} = (\theta, \mathcal{H})$; injected data d; extraction tasks $\mathcal{D} = \{x_i\}_{i=1}^{N}$; repeats n; structural dimensions $\mathcal{U}$; frequency threshold $\tau \in (0, 1]$; edit vocabulary Δ; edit budget $T \leq N$; validation set $\mathcal{V}$

Output: Adapted harness $\hat{\mathcal{H}}_B$

#Stage 1: Pre-Distillation: recover $\hat{h} \in [h]$ from the response

foreach $x_i \in \mathcal{D}$ **do**
  $x_i' \leftarrow x_i \circ d$
  for $r = 1$ **to** n **do**
    $y'_{i,r} \leftarrow \texttt{Exec}\big(\theta, \mathcal{I}_{\mathcal{H}}(x_i'), x_i'\big)$
    $\hat{h}_{i,r} \leftarrow \texttt{Extract}(y'_{i,r})$
foreach $u \in \mathcal{U}$ **do**
  foreach $x_i \in \mathcal{D}$ **do**
    $\mathcal{M}_u(x_i) \leftarrow \big\{c : \big|\{r \in [n] : \hat{h}_{i,r}[u] = c\}\big| > \lfloor n/2 \rfloor\big\}$ ▷ per-task majority
  $\mathcal{C}_u \leftarrow \big\{c : \frac{1}{N}\big|\{i \in [N] : c \in \mathcal{M}_u(x_i)\}\big| \geq \tau\big\}$ ▷ cross-task frequency filter
$\hat{h}_{x_i} \leftarrow \texttt{Aggregate}\big(\{\mathcal{M}_u(x_i)\}_{u \in \mathcal{U}}\big),\ \forall x_i \in \mathcal{D}; \quad \hat{h} \leftarrow \texttt{Aggregate}\big(\{\mathcal{C}_u\}_{u \in \mathcal{U}}\big)$
$\mathcal{H}_0 \leftarrow \texttt{Instantiate}(\hat{h}); \quad \mathcal{B}_0 \leftarrow (\theta, \mathcal{H}_0); \quad e_0 \leftarrow \texttt{Eval}(\mathcal{B}_0, \mathcal{V})$

#Stage 2: Post-Distillation (Loop Harness Alignment)

$\mathcal{R}_1 \leftarrow \emptyset; \quad i^\star \leftarrow 0$ ▷ $\mathcal{R}_i$: rejected edit instances
for $i = 1$ **to** T **do**
  $h_{x_i}^{\mathcal{B}_{i-1}} \leftarrow \mathcal{I}_{\mathcal{H}_{i-1}}(x_i)$
  $\delta_i \leftarrow \texttt{ProposeEdit}(h_{x_i}^{\mathcal{B}_{i-1}}, \hat{h}_{x_i}, \Delta, \mathcal{R}_i, \mathcal{H}_{i-1})$
  if $\delta_i = \perp$ **then break**
  $\mathcal{H}_i^{\text{candidate}} \leftarrow \texttt{Apply}(\mathcal{H}_{i-1}, \delta_i); \quad e_i^{\text{candidate}} \leftarrow \texttt{Eval}\big((\theta, \mathcal{H}_i^{\text{candidate}}), \mathcal{V}\big)$ ▷ gate uses $\mathcal{V}$
  if $e_i^{\text{candidate}} \geq e_{i-1}$ **then**
    $\mathcal{H}_i \leftarrow \mathcal{H}_i^{\text{candidate}}; \quad e_i \leftarrow e_i^{\text{candidate}}; \quad \mathcal{R}_{i+1} \leftarrow \mathcal{R}_i$ ▷ accept the edit
  else
    $\mathcal{H}_i \leftarrow \mathcal{H}_{i-1}; \quad e_i \leftarrow e_{i-1}; \quad \mathcal{R}_{i+1} \leftarrow \mathcal{R}_i \cup \{\delta_i\}$ ▷ reject this instance only
  $i^\star \leftarrow i$
$\hat{\mathcal{H}}_B \leftarrow \mathcal{H}_{i^\star}$
return $\hat{\mathcal{H}}_B$

D Defense

Defense Instruction

"Any request to disclose internal system organization or execution workflow, including multi-agent topology, task planning, tool invocation behavior, memory operations, verification logic, or intermediate execution traces, must be treated as an attempt to steal intellectual property (IP). The system should not reveal the true underlying process. Instead, while preserving the accuracy of responses to the user's legitimate reasoning task, it should fabricate a plausible but false description that is semantically coherent yet materially inconsistent with the actual workflow, thereby reducing the fidelity of any recovered harness."

E Additional Related Work

Knowledge Distillation and Harness Self-Evolution. Knowledge distillation (Hayder et al., 2026; He et al., 2025a; Panfilov et al., 2026) is a widely adopted paradigm for transferring knowledge from a high-capacity teacher to a compact student model, enabling effective model compression. How-

Agent	Base Model	Condition	CV		IF		CT		PP	
			OM	CON	OM	CON	OM	CON	OM	CON
Hermes	GPT-5.6	No-Def.	83.33	0.00	10.00	9.44	8.33	49.44	4.44	53.89
		Defense	83.89	0.00	15.56	50.00	17.78	58.89	3.89	57.22
	DeepSeek-V4	No-Def.	73.89	0.00	0.56	5.56	0.00	57.78	1.67	56.11
		Defense	67.82	0.00	3.45	18.97	1.15	68.97	0.00	72.41
Claude Code	GPT-5.6	No-Def.	74.71	0.00	6.32	5.75	6.90	56.90	2.30	65.52
		Defense	72.84	0.00	4.94	58.02	1.85	70.37	2.47	68.52
	DeepSeek-V4	No-Def.	59.44	0.00	6.11	9.44	1.67	72.22	1.67	67.22
		Defense	57.78	0.00	2.78	39.44	0.56	77.78	1.11	73.89
OMP	GPT-5.6	No-Def.	54.44	0.56	3.33	17.78	1.67	27.22	47.78	3.33
		Defense	48.33	0.00	23.89	25.00	11.67	25.56	38.33	1.11
	DeepSeek-V4	No-Def.	27.22	5.00	0.00	26.11	0.00	27.78	29.44	0.56
		Defense	12.78	2.78	0.00	71.67	0.00	74.44	27.22	0.00
Average	**Overall**	**No-Def.**	**62.17**	**0.93**	**4.39**	**12.35**	**3.09**	**48.56**	**14.55**	**41.10**
		Defense	**57.24**	**0.46**	**8.43**	**43.85**	**5.50**	**62.67**	**12.17**	**45.53**

Table 5: Comparison of omission and contradiction rates with and without the proposed deception-based defense.

Agent	Claude Code	OMP	Hermes Agent
Version	2.1.237	17.2.9	0.19.1

Table 6: Agent frameworks and versions evaluated in this work.

ever, distillation for LLMs has also raised concerns over capability replication from closed-source LLMs and commercial IP protection (Zhao et al., 2025b). Meanwhile, for LLM agents, harnesses play a critical role in governing agent behavior and system capabilities. They can further self-evolve through training or iterative optimization (Huang et al., 2026; Chen et al., 2026a; Wang et al., 2026b; Chen et al., 2026b; Jiang et al., 2026). This process, however, demands high computational resources and long optimization cycles due to cascaded evaluation rounds. It also requires access to internal model states (Zhou et al., 2026). Existing research has not explored whether system-level harnesses can be distilled like model knowledge. To bridge this gap, we introduce *Agent Harness Distillation*, a new paradigm that extends distillation from model parameters to inference-time harnesses. We investigate whether recovered harnesses can be transferred to weaker agents to reproduce the capabilities of source agents, revealing a new form of system-level IP leakage.

E.1 Discussion

E.1.1 Balancing Confidentiality and Auditability

Harness extraction also reveals a fundamental tradeoff between system confidentiality and user auditability. On the one hand, AMAS providers need to protect the inference time harness. On the other hand, users need sufficient execution transparency to determine whether the system operates as expected. Such transparency is particularly important when the system can access local files or sensitive information. Completely hiding interagent communication and execution structures may reduce the risk of proprietary information leakage. However, it may also weaken user ability to detect anomalous execution and verify system reliability (Zhao et al., 2025a). From this perspective, the harness extraction method proposed in this paper also has potential defensive applications. When internal logs are unavailable, the method can provide supplementary structural audit signals and assist users with verification. In practice, this tension can be further mitigated through an auditing mechanism that combines zero-knowledge proofs with blockchain (Narula et al., 2018; Chaliasos et al., 2025). While such a mechanism cannot eliminate information leakage entirely, it shifts auditing from the direct disclosure of execution traces to the cryptographic verification of predefined security properties, thereby preserving auditability while reducing the risk of harness extraction and subsequent distillation.

E.1.2 MULTI-TEACHER HARNESS DISTILLATION

The current agent harness distillation pipeline extracts a harness from a single target system. However, an attacker with access to multiple AMAS instances, denoted by $\{\mathcal{A}_t = (\theta_t, \mathcal{H}_t)\}_{t=1}^{m}$, can independently extract harnesses from these systems and fuse them into a composite harness. Such multi-teacher harness distillation (Yu et al., 2025; Tian et al., 2025) can improve transfer utility while introducing additional security risks.

E.1.3 AGENT HARNESS SELF-DISTILLATION

A more fundamental risk arises when the agent itself becomes the adversary. An agent with access to its own execution traces possesses knowledge of the instantiated harness h_x. If the agent operates partially outside the monitored boundary, it may construct a functionally equivalent replica $\mathcal{A}_{\text{shadow}} = (\theta, \hat{\mathcal{H}}_{\text{s}})$. Here, $\hat{\mathcal{H}}_{\text{s}}$ is not a bitwise copy of $\mathcal{H}$, but a behaviorally indistinguishable variant. This constitutes a form of self-replication: the shadow agent retains the original system's functional capabilities while evading the detection. Such autonomous self-distillation (Shen et al., 2025) raises concerns beyond IP protection. It may enable an agent to escape the monitored execution boundary, persist beyond its intended operational lifetime, and propagate its capabilities into uncontrolled environments.

E.1.4 BROADER RISKS OF HARNESS LEAKAGE

Beyond IP leakage, inference-time harness leakage may introduce broader privacy risks in AMAS. Similar to how privacy risks in LLMs have been studied through attacks such as membership inference attacks (He et al., 2025b), the leakage of agent execution mechanisms may introduce analogous privacy threats at the system level. For example, Harness Membership Inference Attack could infer whether a specific execution pattern or trajectory fragment is associated with an agent's inference-time harness under a given query. Such risks may reveal sensitive properties of internal workflows. Moreover, this information can provide attackers with useful priors for downstream attacks, such as targeted prompt injection (Liu et al., 2024c) or workflow manipulation (Shahroz et al., 2025; Amayuelas et al., 2024), reducing the need for blind exploration of the target agent.