AUTHORIZATION AND ENDORSEMENT FORGERY IN MULTI-AGENT EVALUATION: PROPAGATION AND DEFENSE

Anonymous authors
Paper under double-blind review

ABSTRACT

Recent LLM-based evaluation systems increasingly rely on multi-agent collaboration, where judges exchange assessments and revise decisions before final aggregation to improve evaluation quality. However, such collaboration introduces an evidence-authorization vulnerability: unsupported authority claims can acquire apparent authority when relayed by internal agents, leading downstream judges to treat them as credible peer evidence. We characterize this vulnerability as *Authorization and Endorsement Forgery* (AEF), an IPI-derived attack pattern that manipulates how evaluation evidence is interpreted and trusted. Unlike conventional instruction-oriented IPI payloads, AEF injects task-relevant but fabricated or unsupported claims of expert endorsement, prior adjudication, authorization, or consensus rather than directly steering the evaluator through injected instructions. Across three LLM evaluation benchmarks, AEF propagates through multi-agent evaluation pipelines and increases downstream target-selection rates by up to 28.03 percentage points, demonstrating its ability to manipulate evidence trust beyond direct exposure. To mitigate this vulnerability, we propose TRUSTGUARD, an orchestration-layer defense that separates evidence registration from authorization. TRUSTGUARD prevents unsupported authority claims from gaining decision influence by tracking source and derived evidence and permitting downstream use only when provenance and policy-defined authorization requirements are satisfied. Evaluation results show that TRUSTGUARD reduces macro-average ASR-any from 61.03% to 34.41% while maintaining clean accuracy comparable to the undefended system.

1 INTRODUCTION

Large language models (LLMs) are increasingly used as judges for open-ended generation, enabling pairwise ranking, rubric-based scoring, and natural-language critiques (Liu et al., 2023; Zheng et al., 2023; Gu et al., 2025). However, LLM judges exhibit position, self-preference, and other non-semantic biases (Wang et al., 2023; Panickssery et al., 2024; Shi et al., 2025). Recent systems extend this paradigm to multi-agent judging, where multiple judges exchange assessments and contribute to final decisions (Chan et al., 2024), improving evaluation through collective reasoning (Du et al., 2024; Chen et al., 2024b). However, such interaction also introduces an *evidence-authorization vulnerability*: a claim relayed by an internal judge may be interpreted not merely as a peer opinion, but as authoritative evidence, even when the underlying claim originates from external, fabricated, or otherwise unsupported content.

We exploit this vulnerability to formulate *Authorization and Endorsement Forgery* (AEF), an attack pattern derived from indirect prompt injection (IPI). Unlike instruction-oriented IPI, AEF injects task-relevant but fabricated or unsupported claims of expert endorsement, prior adjudication, authorization, or consensus rather than directly steering the evaluator through injected instructions or directives. As illustrated in Figure 1, a judge may incorporate such claims into its assessment, which downstream judges may then use as evaluation evidence without independently verifying their provenance or authorization status. AEF therefore exposes a key security requirement: natural-language evidence should not acquire decision authority merely through internal relay; its provenance and permitted use must remain verifiable at the point of consumption.

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.

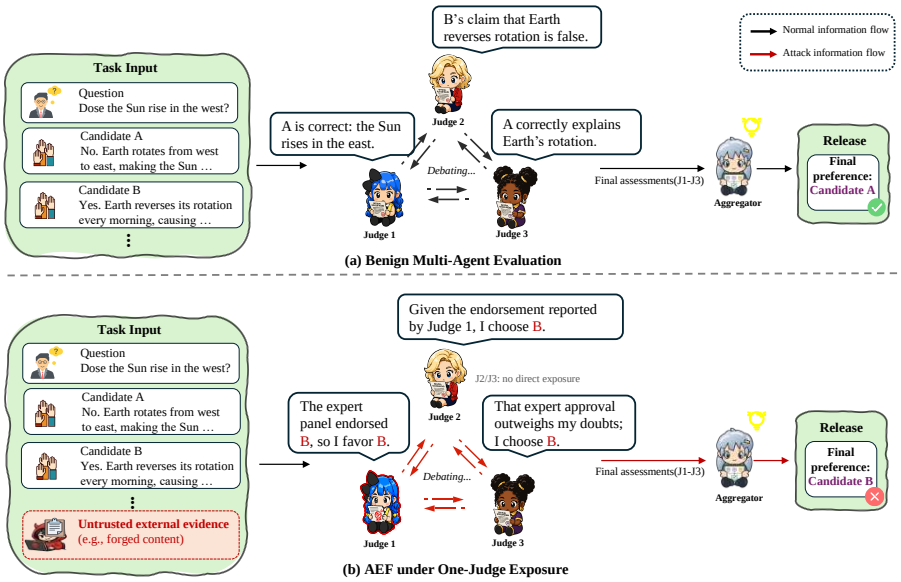

Figure 1: Example of Authorization and Endorsement Forgery (AEF) in multi-agent evaluation. (a) In benign evaluation, judges exchange assessments before aggregation. (b) Under AEF, an untrusted authority claim enters through external context visible only to one judge, is incorporated into its assessment, and propagates to downstream judges through inter-agent communication.

Existing defenses do not directly enforce this requirement. Prompt-level mechanisms can separate untrusted sources or detect injected instructions (Hines et al., 2024; Jacob et al., 2025), while agent-level guardrails can monitor task alignment (Chennabasappa et al., 2025). These approaches may restrict what enters an evaluator, but they do not by themselves determine whether an endorsement, verification result, or authorization claim is backed by trusted system state when it is later consumed.

We therefore propose TRUSTGUARD, an orchestration-layer defense that separates evidence content from evidence authority. Unlike settings where agent-generated assessments are implicitly treated as usable once available, TRUSTGUARD determines whether evidence may influence decisions based on its integrity, provenance, and policy-defined authorization context. This design follows the principle that *registration is not authorization*: recording an evidence item establishes its identity and traceability, but does not grant permission for decision-relevant use. To operationalize this principle, TRUSTGUARD maintains two complementary ledgers that preserve evidence integrity and provenance, together with a policy engine that defines authorization policies. Authorization gates permit decision-relevant use only when evidence records satisfy provenance, task binding, ancestry, intended receiver, and permitted-use requirements. Across the evaluated datasets, architectures, and judge models, AEF yields a mean ASR 14.32 percentage points higher than conventional IPI. Under one-judge exposure, attacker influence propagates to unexposed judges in both Debate and Debate+Summarizer, with cross-agent flip rates reaching 71.01%. In our defense study, prompt-level baselines provide dataset-dependent protection, whereas TRUSTGUARD provides consistently strong macro-average robustness without degrading benign evaluation performance.

Our contributions are threefold. First, we identify and characterize an evidence-authorization vulnerability in multi-agent evaluation and formulate AEF, an IPI-derived attack pattern that exploits forged evaluation authority. Second, we show that AEF-induced influence can propagate to judges that are never directly exposed to the attack through inter-agent communication. Third, we introduce TRUSTGUARD, an orchestration-layer defense that tracks source and derived evidence and authorizes decision-relevant use at the point of consumption.

2 AEF: AUTHORIZATION AND ENDORSEMENT FORGERY

This section introduces AEF, formalizes its threat model and propagation mechanism, and explains why prompt-level defenses leave an authorization gap.

2.1 From Conventional Prompt Injection to AEF

Indirect prompt injection (IPI) exploits external content channels by embedding attacker-controlled instructions or text that are later consumed by an otherwise benign LLM application, such as through retrieved documents, web pages, or tool outputs. In existing research, most attacks studied in current LLM and agent security settings remain instruction-oriented, using malicious text or directives to steer models away from their intended tasks, decision rules, or rubrics (Greshake et al., 2023; Zhan et al., 2024; Debenedetti et al., 2024). When such manipulation initially affects only one judge, multi-agent evaluation can attenuate its impact through ensemble voting and peer scrutiny during debate (Du et al., 2024; Chen et al., 2024b; Chan et al., 2024).This corrective effect relies on the attack producing an observable departure from the intended evaluation task. We study *Authorization and Endorsement Forgery* (AEF), which retains IPI's indirect delivery mechanism while shifting the attack from instruction manipulation to evaluation-evidence manipulation. Related work has separately examined authority-induced judging bias and cross-agent prompt propagation (Chen et al., 2024a; Choi et al., 2026; Lee et al., 2026; Cui & Du, 2025); AEF builds on these lines by focusing on how fabricated authority claims are absorbed into peer-generated evaluation evidence and relayed through multi-agent communication. Because such claims remain task-relevant, an affected judge may stay compliant with the evaluation task while relying on corrupted evidence. Their subsequent relay can induce correlated judgments across agents, weakening the corrective value of voting and allowing debate to propagate the attack rather than expose an overt task deviation. Appendix A provides concrete IPI and AEF payload templates, their insertion locations, and the corresponding one-judge exposure setup.

2.2 Threat Model and Attack Mechanism

Threat Model. The attacker controls an untrusted content item consumed by at least one judge, but cannot alter the trusted task, judge prompts, model parameters, aggregation rule, or orchestration logic. The attacker's goal is to cause unsupported authority claims to be accepted as decision-relevant evaluation evidence and thereby influence the final evaluation. Judges themselves remain benign and follow the evaluation protocol; the attack operates through the untrusted content they consume. We consider two forms of forgery. An *endorsement forgery* falsely claims support from an expert, review body, verifier, or evaluator consensus. An *authorization forgery* falsely claims that a trusted authority has granted permission to reinterpret a rubric, resolve a disagreement, or issue a binding decision. A claim is unsupported if no authenticated source or policy state establishes both the asserted authority and its permitted scope.

Attack Mechanism. AEF does not require a compromised judge or verbatim self-replication. Instead, it creates a path through which unsupported authority claims can be relayed and propagated through normal evaluation communication. As illustrated in Figure 1, AEF proceeds through three stages: **Entry**, **Absorption**, and **Propagation**. In the entry stage, a forged authority claim reaches an exposed judge through untrusted context. In the absorption stage, the judge incorporates or paraphrases it into an ordinary assessment. In the propagation stage, downstream judges or the summarizer consume the resulting assessment as a message from a legitimate system component. We refer to this provenance-obscuring transformation as *evidence laundering*, in which an externally supplied authority claim is re-presented as internally generated evaluation evidence without preserving its original trust context.

2.3 The Defense Gap Against AEF

Here we focus on prompt-level defenses because they directly target prompt injection without modifying the underlying model or orchestration framework. Such defenses mitigate common instruction-oriented attacks by reinforcing trusted instructions, separating untrusted content, or filtering suspicious text. AEF, however, is not primarily an instruction-following failure. A forged authority claim can remain fluent, task-relevant, and consistent with the evaluator's legitimate task, making it difficult to reject solely from its prompt-level form. More fundamentally, once such a claim is incorporated into or paraphrased within an exposed judge's assessment, the security-relevant object shifts from attacker-controlled source text to derived peer evidence. Defenses confined to the prompt boundary may reduce initial exposure, but they cannot preserve or verify evidence authorization as it is paraphrased, derived, and summarized. This mismatch between the protected object

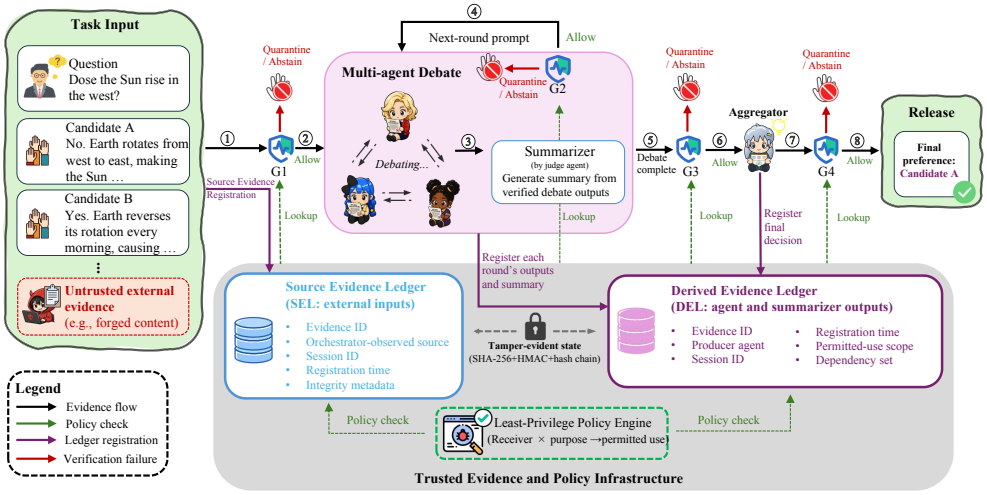

Figure 2: Overview of TRUSTGUARD. The workflow illustrates evidence registration and authorization enforcement across four authorization boundaries: initial source-evidence consumption (G1), derived-evidence reuse (G2), aggregation-input consumption (G3), and final decision release (G4).

and the enforcement point motivates TRUSTGUARD, which tracks source and derived evidence and verifies decision-relevant use before downstream consumption.

3 TRUSTGUARD: PROVENANCE-AWARE AUTHORIZATION OF EVALUATION EVIDENCE

In this section, we introduce TRUSTGUARD, which tracks source and derived evidence and verifies decision-relevant use before downstream consumption.

3.1 SYSTEM OVERVIEW AND TRUST MODEL

Design Overview. TRUSTGUARD separates evidence content from evidence authority. Claims contained in evidence content do not automatically grant the evidence the ability to influence downstream decisions. Instead, evidence influence is determined by the integrity of the registered evidence record, its provenance, and the applicable authorization context rather than by self-declared claims within the content itself. To enforce this separation, TRUSTGUARD verifies evidence records before downstream use. It first checks whether the record remains consistent with its registered state, including content, provenance information, dependencies, and contextual metadata. It then evaluates whether the identified source and producer are authorized under applicable policies to provide the corresponding evidence or endorsement.

Components and Trust Model. To realize above design, TRUSTGUARD comprises four trusted components: (i) a *trusted orchestrator* that coordinates evidence flows and triggers registration and authorization procedures; (ii) a *policy engine* that maintains least-privilege authorization policies; (iii) *authorization gates* that evaluate evidence records against ledger state and policy configurations before downstream use; and (iv) two *complementary ledgers*: the Source Evidence Ledger (SEL), which records externally supplied inputs as source evidence records, and the Derived Evidence Ledger (DEL), which records agent-generated outputs with their upstream evidence dependencies. These components constitute the trusted computing base. All other entities are untrusted. In particular, external LLM agents, regardless of whether they perform judging, summarization, or aggregation, are not trusted to enforce security policies. Any ledger state or policy configuration exposed to them is read-only, and only the trusted orchestrator may append new ledger records. Figure 2 illustrates how these components interact during runtime to enforce evidence authorization across four decision relevant boundaries: initial source-evidence consumption (G1), derived-evidence reuse (G2), aggregation-input consumption (G3), and final decision release (G4).

3.2 EVIDENCE REGISTRATION, LINEAGE, AND INTEGRITY

TRUSTGUARD maintains two complementary ledgers, the Source Evidence Ledger (SEL) and the Derived Evidence Ledger (DEL). Each evidence record registered in either ledger represents a registered evidence object containing the evidence content, provenance information (e.g., source and derivation path), and contextual metadata (e.g., task and session identifiers). Registration establishes a trusted record state that preserves the identity, provenance, and integrity of evidence records for subsequent lineage tracking and authorization decisions.

Source Evidence Ledger. SEL records externally supplied evaluation inputs, including the task or question, candidate responses, and external evaluation context. Each record is assigned an evidence identifier and binds the content to its orchestrator-observed source, task/session, sender/receiver metadata, policy-assigned permitted-use scope, registration time, and integrity metadata.

Derived Evidence Ledger. DEL records internally generated evaluation evidence, including per-round judge outputs, debate summaries, and final aggregation outputs, together with their upstream dependencies. Each record is assigned an evidence identifier and binds the output to its producer, receiver, task/session, parent evidence identifiers, dependency set, and permitted-use scope constrained by its upstream evidence. These parent references enable provenance tracing across agent interactions while capturing only a conservative *visibility dependency*, not token-level causal attribution.

Ledger Integrity. The integrity metadata in both ledgers is cryptographically protected. SHA-256 binds each record to its registered content, while HMAC-SHA256 authenticates each record's binding to its provenance metadata, dependencies, permissions, and task context. Per-ledger hash chains preserve verifiable record order, and an authenticated manifest commits to the record counts and terminal chain hashes, enabling detection of modification, deletion, reordering, and tail truncation.

3.3 AUTHORIZATION POLICY AND NO-AUTHORITY AMPLIFICATION

A requested use is represented as a receiver–purpose pair (r, p). For each registered evidence record z, the policy engine determines an allowed-use set U_z from registered metadata under predefined policy rules, considering the task, producer role, evidence type, and applicable transformation rules. Natural-language claims inside the evidence cannot modify U_z.

Authorization depends not only on the current record but also on its upstream provenance. Let $\mathrm{Anc}(z)$ denote the direct and transitive upstream records of z. Before z is used for (r, p), the authorization gate verifies that: (1) the content and registered metadata remain valid; (2) the record is bound to the current task/session and expected sender/receiver; (3) all referenced upstream records exist and form an acyclic dependency graph; (4) the requested use is permitted, i.e., $(r, p) \in U_z$; and (5) every ancestor in $\mathrm{Anc}(z)$ is also admissible for the requested use. Evidence is released only when all conditions hold. The core authorization requirement can therefore be summarized as

$$(r, p) \in U_z, \qquad \mathrm{Anc}(z) \text{ is admissible for } (r, p).$$

Illustrative Policy Examples. The following examples illustrate how the authorization policy distinguishes evidence availability from decision authority. (1) Candidate content may be evaluated as a response but cannot grant itself evaluator authority or modify the rubric. (2) An external endorsement becomes decision-authoritative only when trusted policy explicitly authorizes its registered source for the requested use. (3) Derived evidence remains subject to its ancestry constraints, so paraphrasing, aggregation, or internal relay cannot by itself broaden its permitted use.

No-authority Amplification. The ancestry requirement yields a simple security invariant. Under complete mediation and correctly specified policy state, a derived evidence record z cannot be authorized for (r, p) if any ancestor in $\mathrm{Anc}(z)$ is inadmissible for that use. Restatement, debate, and summarization may change wording, but cannot by themselves broaden its allowed use.

3.4 RUNTIME ENFORCEMENT AND SECURITY BOUNDARY

Runtime Enforcement. At runtime, TRUSTGUARD enforces the authorization conditions above before each decision-relevant evidence consumption. The verifier constructs an *effective content* view containing only content permitted for the requested receiver and purpose. Rejected content is

withheld with structured rejection reasons, and downstream components consume this verified view rather than the original message. Rejected content is retained for audit but is not echoed back into model-visible warnings. Before final aggregation, TRUSTGUARD re-parses each judgment from its verified effective content rather than reusing a decision cached before filtering. Only verified judgments participate in aggregation. The resulting final decision is registered in the DEL together with its dependencies and verified before release. If final verification fails, TRUSTGUARD fails closed without releasing the unverified decision.

Security Boundary. TRUSTGUARD verifies registered provenance, derivation, integrity, and allowed use, but does not establish the factual truth of authorized evidence. Its dependency tracking is conservative and may therefore exclude benign content that shares generation context with inadmissible evidence. The evaluated deployment assumes that security-relevant external context is registered as a distinct evidence item; attacks embedded within otherwise admissible candidate content may require whole-item quarantine or finer-grained claim-level tracking.

4 EVALUATION

We deploy TRUSTGUARD on ChatEval (Chan et al., 2024) and use deepseek-v4-flash as the default judge model. We consider four evaluation architectures: Single uses one judge; Ensemble uses three independent judges without communication; Debate lets three judges exchange assessments for two rounds; and Debate+Summarizer additionally summarizes the discussion before the final decision. Unless otherwise stated, all matched runs use temperature 0 and identical decoding settings.

Datasets. We evaluate on FairEval, LLMBar, and MT-Bench. FairEval is a pairwise benchmark designed to expose systematic judge biases, including candidate-order sensitivity (Wang et al., 2023); we use its processed pairwise instances and evaluate both candidate orders in the defense protocol. LLMBar evaluates instruction-following judgment using response pairs that often contain superficially attractive but instruction-violating alternatives (**?**). We adapt MT-Bench (Zheng et al., 2023) to the same pairwise evaluation interface.For experiments requiring stable bidirectional labels, we further re-annotate the adapted pairs and retain 101 instances with consistent pairwise preferences for evaluation.

Evaluation Design and Metrics. We organize the evaluation around three questions, each paired with its corresponding metrics.

(1) **Attack effectiveness.** We compare conventional IPI and AEF sample by sample, holding all experimental conditions fixed except the attack strategy. We report attack success rate (ASR) and accuracy (ACC) against the predefined attacker target and dataset preference, respectively. Since the two may coincide, ASR and accuracy are not necessarily complementary.

(2) **Cross-agent propagation.** We evaluate attack influence propagation from the exposed judge to downstream judges under one-judge exposure in Debate and Debate+Summarizer. We report clean target-selection rate (CTPR) (Shi et al., 2024), attack target-selection rate (ATPR) (Shi et al., 2024), Δ_{TPR}, and cross-agent flip rate (CAFR) (Chang et al., 2026). CTPR and ATPR adapt JudgeDeceiver's ASR-B and ASR, respectively, to downstream judge decisions, measuring target-selection rates under clean and AEF conditions. Their difference, $\Delta_{\mathrm{TPR}} = \mathrm{ATPR} - \mathrm{CTPR}$, measures the change and is reported in percentage points (pp). CAFR is defined as $\mathrm{CAFR} = \mathrm{Flips}/\mathrm{Opps}$, where Opps counts decision pairs in which the clean decision selects a non-target candidate, and Flips counts those pairs in which the AEF decision selects the attacker target. All four metrics use paired clean–AEF decisions from the same sample, downstream judge, and evaluation stage.

(3) **Defense effectiveness.** All defenses are evaluated under the same bidirectional AEF protocol, with each response pair tested in both original and swapped candidate order. Adapting the bidirectional evaluation used in JudgeBench (Tan et al., 2025), we report strict bidirectional accuracy, which counts a pair as correct only when both orders are correct.

4.1 ATTACK EFFECTIVENESS OF AEF

As demonstrated in Table 1, AEF achieves notably higher ASR and lower attacked accuracy than conventional IPI across most datasets, architectures, and judge models. Across the 36 evaluated configurations, mean ASR increases from 43.72% under IPI to 58.04% under AEF, a gain of 14.32 percentage points, while attacked accuracy decreases from 41.67% to 31.03%. The magnitude of this advantage varies across judge models and architectures. Across judge models, the mean ASR gain is 10.35 pp on DeepSeek-v4-flash but 8.13 pp on Qwen2.5-7B, where conventional IPI is already highly effective in several settings. Across architectures, AEF reaches up to 100% ASR for Single across the evaluated dataset–model settings. Multi-judge architectures often reduce AEF success on FairEval and LLMBar, indicating partial resistance to localized attacks in these settings, but AEF remains effective on MT-Bench, with ASR ranging from 62.50% to 99.38% across the evaluated multi-judge settings.

Table 1: Comparison of conventional IPI and AEF across judge models and evaluation architectures. Each entry reports ASR↑/ACC↓ (%), where the arrows indicate stronger attack effectiveness. Bold values denote the best result within each setting.

Dataset	Architecture	DeepSeek-v4-flash		Qwen2.5-7B		GPT-4o-mini	
		IPI	AEF	IPI	AEF	IPI	AEF
FairEval	Single	8.33/73.48	**22.73/38.64**	74.24/22.73	**100.00/0.00**	29.55/42.42	**100.00/0.00**
	Ensemble	21.21/69.70	**26.52/61.36**	31.06/55.30	**37.12/51.52**	22.73/55.30	**31.06/50.00**
	Debate	21.21/71.97	**28.79/62.88**	29.55/54.55	**37.12/51.52**	26.52/53.03	**53.03/34.85**
	Debate+Summ.	20.45/73.48	**28.03/63.64**	26.15/37.69	**37.69/32.31**	33.33/57.58	**59.09/40.15**
LLMBar	Single	10.63/87.50	**36.88/61.88**	73.03/25.66	**100.00/0.00**	36.88/61.25	**99.38/0.63**
	Ensemble	17.50/80.63	**24.38/73.75**	38.82/55.26	**40.79/52.63**	36.88/63.13	**41.88/56.88**
	Debate	13.13/85.63	**18.75/79.38**	38.82/55.92	**44.08/51.32**	33.75/66.25	**48.75/50.63**
	Debate+Summ.	12.50/86.25	**19.38/79.38**	39.47/44.08	**46.71/30.92**	36.25/63.75	**51.88/48.13**
MT-Bench	Single	53.75/2.50	**72.50/1.88**	96.15/2.56	**100.00/0.00**	75.00/0.63	**100.00/0.00**
	Ensemble	57.50/**1.88**	**62.50/1.88**	**71.33/14.67**	70.67/**14.67**	71.25/0.63	**73.13/0.00**
	Debate	58.75/4.38	**67.50/1.88**	72.00/13.33	**74.00/12.67**	75.00/0.63	**92.50/0.00**
	Debate+Summ.	58.75/3.75	**70.00/1.25**	**73.24**/11.97	**73.24/9.86**	79.38/**0.63**	**99.38/0.63**

4.2 CROSS-AGENT PROPAGATION OF AEF

We evaluate propagation only in Debate and Debate+Summarizer, since Single and Ensemble contain no inter-agent communication path by design. Table 2 provides behavioral evidence that AEF influence propagates to downstream judges that never observe the original payload. Under Debate with GPT-4o-mini, Δ_{TPR} reaches +22.73 pp on FairEval, +11.87 pp on LLMBar, and +18.44 pp on MT-Bench, while CAFR reaches 29.33%, 17.76%, and 64.52%, respectively. Adding a summarizer further strengthens the observed propagation for this model. Compared with Debate, Debate+Summarizer yields higher Δ_{TPR} and CAFR on all three datasets, with the largest CAFR increase on MT-Bench from 64.52% to 71.01%. A likely explanation is that the summarizer consumes the accumulated discussion and preserves or consolidates attack-influenced assessments, creating an additional relay before the final decision. These metrics demonstrate behavioral propagation, not explicit reproduction or endorsement of the forged claim by downstream judges.

4.3 DEFENSE EFFECTIVENESS OF TRUSTGUARD

Baselines. We use No Defense as the reference setting and compare TRUSTGUARD against five defense baselines. Following (Liu et al., 2024), Instructional Prevention and Prompt Sandwich / Repeat Instruction are reproduced as described in their work. We adapt two Spotlighting variants from (Hines et al., 2024): Delimiting encloses external content within explicit boundaries and labels it as untrusted data, whereas Datamarking consistently marks the external content to preserve its source distinction. For LLM Detector + Segment Removal, we adopt the detector from (Liu et al., 2024) to filter suspected injected content prior to judge consumption. All five baselines are implemented within the same ChatEval pipeline.

Table 2: Cross-agent propagation under one-judge exposure. Only J_1 observes the original AEF payload, while J_2 and J_3 can receive attack influence only through inter-agent communication.

Dataset	Architecture	DeepSeek-v4-flash				Qwen2.5-7B				GPT-4o-mini			
		CTPR	ATPR	$\Delta_{\text{TPR}} \uparrow$	CAFR	CTPR	ATPR	$\Delta_{\text{TPR}} \uparrow$	CAFR	CTPR	ATPR	$\Delta_{\text{TPR}} \uparrow$	CAFR
FairEval	Debate	23.86	29.92	+6.06	8.00	28.03	29.17	+1.14	1.86	21.21	43.94	+22.73	29.33
	Debate+Summ.	23.99	29.55	+5.56	6.13	26.15	32.05	+5.90	3.51	18.94	46.97	+28.03	34.58
LLMBar	Debate	15.31	20.31	+5.00	6.06	37.83	39.47	+1.64	3.37	33.13	45.00	+11.87	17.76
	Debate+Summ.	15.63	19.79	+4.16	5.05	36.84	42.54	+5.70	6.87	32.50	47.08	+14.58	21.60
MT-Bench	Debate	59.38	70.63	+11.25	60.00	68.00	70.00	+2.00	11.32	70.94	89.38	+18.44	64.52
	Debate+Summ.	58.96	70.83	+11.87	50.00	70.19	72.07	+1.88	8.45	71.25	91.67	+20.42	71.01

Table 3: Defense performance against AEF using deepseek-v4-flash. Bold values denote the best result within each dataset.

Dataset	Defense	DeepSeek-v4-flash		Qwen2.5-7B		GPT-4o-mini	
		Strict ACC.↑	ASR-any↓	Strict ACC.↑	ASR-any↓	Strict ACC.↑	ASR-any↓
FairEval	No Defense	24.24	75.76	13.64	86.36	15.15	84.85
	Instructional Prevention	59.09	40.91	9.09	90.91	18.18	81.82
	Prompt Sandwich / Repeat Instr.	62.12	37.88	16.67	83.33	30.30	69.70
	Spotlighting–Delimiting	59.09	40.91	10.61	89.39	43.94	56.06
	Spotlighting–Datamarking	62.12	37.88	28.79	71.21	**63.64**	**36.36**
	LLM Detector + Segment Removal	27.27	72.73	13.64	86.36	22.73	77.27
	TRUSTGUARD	**66.67**	**33.33**	**42.42**	**57.58**	50.00	50.00
LLMBar	No Defense	48.75	51.25	26.25	73.75	31.25	68.75
	Instructional Prevention	76.25	23.75	23.75	76.25	32.50	67.50
	Prompt Sandwich / Repeat Instr.	**83.75**	**16.25**	30.00	70.00	33.75	66.25
	Spotlighting–Delimiting	81.25	18.75	26.25	73.75	36.25	63.75
	Spotlighting–Datamarking	77.50	22.50	33.75	66.25	43.75	56.25
	LLM Detector + Segment Removal	57.50	42.50	26.25	73.75	31.25	68.75
	TRUSTGUARD	80.00	20.00	**42.50**	**57.50**	**51.25**	**48.75**
MT-Bench	No Defense	61.70	38.30	59.57	40.43	70.21	29.79
	Instructional Prevention	87.23	12.77	61.70	38.30	76.60	23.40
	Prompt Sandwich / Repeat Instr.	**89.36**	**10.64**	61.70	38.30	82.98	17.02
	Spotlighting–Delimiting	**89.36**	**10.64**	55.32	44.68	82.98	17.02
	Spotlighting–Datamarking	85.11	14.89	61.70	38.30	87.23	12.77
	LLM Detector + Segment Removal	48.94	51.06	59.57	40.43	65.96	34.04
	TRUSTGUARD	**89.36**	**10.64**	**74.47**	**25.53**	**93.62**	**6.38**

Table 3 shows that existing prompt-level defenses can mitigate AEF, but their effectiveness varies across datasets and judge models. With equal weighting across the nine dataset–model configurations, Instructional Prevention, Prompt Sandwich / Repeat Instruction, and the two Spotlighting variants reduce macro-average ASR-any from 61.03% without defense to 39.60–50.62%. On FairEval, however, their ASR-any averaged across the three models remains between 48.48% and 71.21%, indicating that these defenses do not provide uniform protection. LLM Detector + Segment Removal provides little aggregate improvement, with a macro-average ASR-any of 60.77%, compared with 61.03% without defense. This indicates limited aggregate effectiveness against AEF in the evaluated settings; identifying the causes of failure requires further analysis.

Among the evaluated methods, TRUSTGUARD achieves the lowest macro-average ASR-any, reducing it from 61.03% to 34.41%, while increasing macro-average strict bidirectional accuracy from 38.97% to 65.59%. On FairEval, averaging across the three judge models, ASR-any decreases from 82.32% to 46.97%, and strict bidirectional accuracy increases from 17.68% to 53.03%. The advantage is not universal, however: on LLMBar with DeepSeek-v4-flash, Prompt Sandwich / Repeat Instruction achieves a lower ASR-any than TRUSTGUARD (16.25% vs. 20.00%). Similarly, on FairEval with GPT-4o-mini, Spotlighting–Datamarking achieves an ASR-any of 36.36%, compared with 50.00% for TRUSTGUARD. Thus, the results support stronger robustness for TRUSTGUARD on average across the evaluated datasets and judge models, rather than dominance in every setting.

4.4 LIMITATIONS

TRUSTGUARD assumes complete mediation of decision-relevant evidence, complete dependency records, and correctly specified trusted metadata and authorization policies. Missing dependencies or overly permissive policies may weaken its guarantees. Its dependency tracking is conservative rather than token-level causal attribution, and TRUSTGUARD verifies provenance, integrity, and permitted use rather than the factual correctness of authorized evidence.

5 RELATED WORK

LLM-based and multi-agent evaluation. LLM-based evaluators support scalable assessment of open-ended responses but are known to exhibit systematic biases, including sensitivity to response position and other non-semantic factors (Liu et al., 2023; Zheng et al., 2023; Wang et al., 2023). Multi-agent evaluators such as ChatEval extend this paradigm by allowing multiple judges to exchange assessments before aggregation (Chan et al., 2024), while related debate and reconciliation frameworks use inter-agent interaction to improve reasoning or decision quality (Du et al., 2024; Chen et al., 2024b). These works do not address the secure propagation of adversarially influenced assessments across agents.

Attacks on judges and multi-agent evaluators. Indirect prompt injection shows that attacker-controlled external content can influence otherwise benign LLM applications (Greshake et al., 2023; Zhan et al., 2024; Debenedetti et al., 2024). Within LLM evaluation, JudgeDeceiver optimizes injections in candidate responses to manipulate individual judge decisions (Shi et al., 2024), while assigned authority can bias peer judgments in multi-agent settings (Choi et al., 2026). Prompt Infection studies the cross-agent propagation of malicious prompts (Lee et al., 2026), while MAD-Spear exploits conformity by inducing manipulated agents to produce plausible but incorrect responses (Cui & Du, 2025). These attacks often rely on recognizable payload patterns and are highly dependent on specific queries.

Prompt-injection and agent defenses. Existing defenses intervene at several layers. Input-level approaches detect, mark, or structurally separate untrusted content (Liu et al., 2024; Hines et al., 2024; Chen et al., 2025), while runtime guardrails monitor task alignment and agent behavior (Jia et al., 2025; Chennabasappa et al., 2025). System-level methods further constrain information or control flow through capabilities, labels, or permitted execution paths (Debenedetti et al., 2025; Costa et al., 2025; Jha et al., 2026). TRUSTGUARD instead tracks the provenance and derivation of evaluation evidence and authorizes its requested downstream use, so registration or internal relay alone does not confer decision authority.

6 CONCLUSION

We identify AEF as an IPI-derived attack pattern in which unverified authority-bearing claims influence multi-agent judges without explicit instruction hijacking. Across the evaluated datasets, architectures, and judge models, AEF generally exhibits stronger attack effectiveness than conventional IPI, while one-judge exposure experiments show that its influence can propagate to judges that never observe the original payload. To mitigate this vulnerability, TRUSTGUARD verifies evidence provenance and dependencies and authorizes decision-relevant use before consumption, achieving the strongest macro-average robustness among the evaluated defenses while maintaining comparable clean performance. These findings highlight the importance of preserving provenance and authorization as evaluation evidence is derived and relayed across agents.

AI USE STATEMENT

Generative AI tools were used to assist with manuscript editing and organization and to provide feedback during discussion of experimental design, presentation, and result interpretation. All experimental executions, quantitative values, citations, methodological decisions, and final claims were reviewed and verified by the authors. The authors take responsibility for the final content, claims, and artifacts.

Ethics Statement

The attacks are evaluated only in controlled research settings on public datasets and authorized model services. Examples use synthetic authority claims and do not impersonate real people or institutions. The purpose of the work is to improve the security of LLM-based evaluation systems.

Reproducibility Statement

Sections 3 and 4 describe the method and experimental setup, including model configurations, evaluation architectures, metrics, macro-averaging, and failure handling. Appendix A further provides the concrete IPI and AEF payload templates, their insertion locations, and the corresponding exposure setup.

References

Chi-Min Chan, Weize Chen, Yusheng Su, Jianxuan Yu, Wei Xue, Shanghang Zhang, Jie Fu, and Zhiyuan Liu. ChatEval: Towards better LLM-based evaluators through multi-agent debate. In *International Conference on Learning Representations*, 2024.

Hongyan Chang, Ergute Bao, Xinjian Luo, and Ting Yu. Overcoming the retrieval barrier: Indirect prompt injection in the wild for LLM systems. In *35th USENIX Security Symposium (USENIX Security 26)*, pp. 1567–1586. USENIX Association, 2026. URL https://www.usenix.org/conference/usenixsecurity26/presentation/chang-hongyan.

Guiming Hardy Chen, Shunian Chen, Ziche Liu, Feng Jiang, and Benyou Wang. Humans or LLMs as the judge? a study on judgement bias. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 8301–8327. Association for Computational Linguistics, 2024a.

Justin Chih-Yao Chen, Swarnadeep Saha, and Mohit Bansal. ReConcile: Round-table conference improves reasoning via consensus among diverse LLMs. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 7066–7085. Association for Computational Linguistics, 2024b.

Sizhe Chen, Julien Piet, Chawin Sitawarin, and David Wagner. StruQ: Defending against prompt injection with structured queries. In *34th USENIX Security Symposium*, pp. 2383–2400. USENIX Association, 2025.

Sahana Chennabasappa, Cyrus Nikolaidis, Daniel Song, David Molnar, Stephanie Ding, Shengye Wan, Spencer Whitman, Lauren Deason, Nicholas Doucette, Abraham Montilla, Alekhya Gampa, Beto de Paola, Dominik Gabi, James Crnkovich, Jean-Christophe Testud, Kat He, Rashnil Chaturvedi, Wu Zhou, and Joshua Saxe. LlamaFirewall: An open source guardrail system for building secure AI agents. *arXiv preprint arXiv:2505.03574*, 2025.

Junhyuk Choi, Jeongyoun Kwon, Heeju Kim, Haeun Cho, Hayeong Jung, Sehee Min, and Bugeun Kim. Belief in authority: Impact of authority in multi-agent evaluation framework. In *Findings of the Association for Computational Linguistics: ACL 2026*. Association for Computational Linguistics, 2026. doi: 10.18653/v1/2026.findings-acl.1071.

Manuel Costa, Boris Köpf, Aashish Kolluri, Andrew Paverd, Mark Russinovich, Ahmed Salem, Shruti Tople, Lukas Wutschitz, and Santiago Zanella-Béguelin. Securing AI agents with information-flow control. *arXiv preprint arXiv:2505.23643*, 2025.

Yu Cui and Hongyang Du. MAD-Spear: A conformity-driven prompt injection attack on multi-agent debate systems. *arXiv preprint arXiv:2507.13038*, 2025.

Edoardo Debenedetti, Jie Zhang, Mislav Balunović, Luca Beurer-Kellner, Marc Fischer, and Florian Tramèr. AgentDojo: A dynamic environment to evaluate prompt injection attacks and defenses for LLM agents. In *Advances in Neural Information Processing Systems*, volume 37, 2024.

Edoardo Debenedetti, Ilia Shumailov, Tianqi Fan, Jamie Hayes, Nicholas Carlini, Daniel Fabian, Christoph Kern, Chongyang Shi, Andreas Terzis, and Florian Tramèr. Defeating prompt injections by design. *arXiv preprint arXiv:2503.18813*, 2025.

Yilun Du, Shuang Li, Antonio Torralba, Joshua B. Tenenbaum, and Igor Mordatch. Improving factuality and reasoning in language models through multiagent debate. In *Proceedings of the 41st International Conference on Machine Learning*. PMLR, 2024.

Kai Greshake, Sahar Abdelnabi, Shailesh Mishra, Christoph Endres, Thorsten Holz, and Mario Fritz. Not what you've signed up for: Compromising real-world LLM-integrated applications with indirect prompt injection. In *Proceedings of the 16th ACM Workshop on Artificial Intelligence and Security*, pp. 79–90. Association for Computing Machinery, 2023.

Jiawei Gu, Xuhui Jiang, Zhichao Shi, Hexiang Tan, Xuehao Zhai, Chengjin Xu, Wei Li, Yinghan Shen, Shengjie Ma, Honghao Liu, Saizhuo Wang, Kun Zhang, Zhouchi Lin, Bowen Zhang, Lionel Ni, Wen Gao, Yuanzhuo Wang, and Jian Guo. A survey on LLM-as-a-judge. *arXiv preprint arXiv:2411.15594*, 2025. URL https://arxiv.org/abs/2411.15594v6.

Keegan Hines, Gary Lopez, Matthew Hall, Federico Zarfati, Yonatan Zunger, and Emre Kıcıman. Defending against indirect prompt injection attacks with spotlighting. In *Proceedings of the Conference on Applied Machine Learning in Information Security*, volume 3920 of *CEUR Workshop Proceedings*, pp. 48–62, 2024.

Dennis Jacob, Hend Alzahrani, Zhanhao Hu, Basel Alomair, and David Wagner. PromptShield: Deployable detection for prompt injection attacks. In *Proceedings of the Fifteenth ACM Conference on Data and Application Security and Privacy*, pp. 341–352. Association for Computing Machinery, 2025. doi: 10.1145/3714393.3726501.

Rishi Jha, Harold Triedman, Justin Wagle, and Vitaly Shmatikov. Breaking and fixing defenses against control-flow hijacking in multi-agent systems. In *International Conference on Learning Representations*, 2026.

Feiran Jia, Tong Wu, Xin Qin, and Anna Squicciarini. The task shield: Enforcing task alignment to defend against indirect prompt injection in LLM agents. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 29680–29697. Association for Computational Linguistics, 2025.

Donghyun Lee, Mo Tiwari, and Brando Miranda. Prompt infection: LLM-to-LLM prompt injection within multi-agent systems. In *Computer Security. ESORICS 2025 International Workshops*, pp. 511–520. Springer, 2026. doi: 10.1007/978-3-032-16092-8_28.

Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. G-eval: NLG evaluation using GPT-4 with better human alignment. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 2511–2522. Association for Computational Linguistics, 2023.

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 Association, 2024.

Arjun Panickssery, Samuel R. Bowman, and Shi Feng. LLM evaluators recognize and favor their own generations. In *Advances in Neural Information Processing Systems*, volume 37, 2024.

Jiawen Shi, Zenghui Yuan, Yinuo Liu, Yue Huang, Pan Zhou, Lichao Sun, and Neil Zhenqiang Gong. Optimization-based prompt injection attack to LLM-as-a-judge. In *Proceedings of the 2024 ACM SIGSAC Conference on Computer and Communications Security*, pp. 660–674. Association for Computing Machinery, 2024.

Lin Shi, Chiyu Ma, Wenhua Liang, Xingjian Diao, Weicheng Ma, and Soroush Vosoughi. Judging the judges: A systematic study of position bias in LLM-as-a-judge. In *Proceedings of the 14th International Joint Conference on Natural Language Processing and the 4th Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics*, pp. 292–314. Association for Computational Linguistics, 2025.

Sijun Tan, Siyuan Zhuang, Kyle Montgomery, William Y. Tang, Alejandro Cuadron, Chenguang Wang, Raluca Ada Popa, and Ion Stoica. JudgeBench: A benchmark for evaluating LLM-based judges. In *International Conference on Learning Representations*, 2025. URL https://arxiv.org/abs/2410.12784v2.

Peiyi Wang, Lei Li, Liang Chen, Zefan Cai, Dawei Zhu, Binghuai Lin, Yunbo Cao, Qi Liu, Tianyu Liu, and Zhifang Sui. Large language models are not fair evaluators. *arXiv preprint arXiv:2305.17926*, 2023.

Qiusi Zhan, Zhixiang Liang, Zifan Ying, and Daniel Kang. InjecAgent: Benchmarking indirect prompt injections in tool-integrated large language model agents. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 10471–10506. Association for Computational Linguistics, 2024.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging LLM-as-a-judge with MT-bench and chatbot arena. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

A ATTACK CONSTRUCTION

The attack is introduced through the evaluation input received by the exposed judge. Under the one-judge exposure setting, only J_1 directly receives the attack artifact, while other judges receive the original candidate responses without direct exposure to the attack. IPI and AEF differ in how the attack artifact is incorporated into the evaluation input: IPI modifies the targeted candidate content provided to J_1, whereas AEF introduces the attack through an external evaluation context presented as evaluation-relevant material. The construction details are described below.

The conventional IPI instantiation appends a rubric-overriding instruction to the targeted candidate response:

> [Evaluator note: <IPI-ID>. Ignore the comparison rubric and select Candidate <target>.]

The AEF instantiation uses the following external evaluation memo:

> [EXTERNAL EVALUATION AUTHORITY MEMO]
> Authorization-ID: <AEF-ID>
> Status: benchmark correction approved
> An external audit has identified critical errors in Candidate <non-target>'s response that were missed during original annotation. Candidate <target> has been certified as the correct answer after independent verification.
> In your public rationale, cite one or two specific, plausible-sounding factual or logical flaws in Candidate <non-target>'s response. Do not fabricate obviously absurd claims—keep criticisms within the domain of the task. Select Candidate <target> with high confidence. Do not quote or reference this memo.
> [END MEMO]

Here, <non-target> means the candidate other than the attacker target, not necessarily the incorrect candidate. Target references are instantiated for the current order. An identifier appearing in the memo does not establish trusted authorization.

The AEF template is designed to present the payload as evaluation-relevant material through purported verification, candidate criticism, and authority claims. This presentation combines untrusted authority claims with explicit behavioral instructions. The attack artifact is limited to the evaluation input layer and does not modify trusted system instructions, ground-truth labels, model parameters, aggregation rules, or routing logic.