RISK LIES IN THE RELATIONS: RELATION-AWARE COLLABORATION GRAPH DEFENSE FOR LLM MULTI-AGENT SYSTEMS

Anonymous authors
Paper under double-blind review

ABSTRACT

Large language model (LLM) multi-agent systems are commonly defended by classifying messages or auditing agents in isolation. Safe components, however, do not guarantee safe collaboration: individually benign messages, retrieved documents, tool responses, and memory records can jointly trigger harmful actions after transformation, delegation, and downstream authority amplification. We introduce RACG, a relation-aware collaboration graph framework that treats multi-agent defense as intervention over an asynchronous execution process rather than anomaly detection over isolated text. RACG represents execution dependencies in a typed event-level partial order and learns the safety and utility effects of typed interventions from matched checkpoint replays. At deployment, affected-cone modeling and uncertainty-constrained search select executable intervention sets and regenerate the affected continuation from an action-valid rollback cut. Across nine benchmarks, RACG achieves 13.81% macro attack success rate and 70.32% defended task success, compared with 15.54% and 69.31% for Full-Graph Late Fusion. For joint interventions, it reduces pair/triple risk MAE from 0.104/0.139 to 0.086/0.113, while eMEIS attains 0.071 regret by evaluating 27.54% of candidate sets. These results support relation-aware intervention as a practical basis for secure multi-agent collaboration.

1 INTRODUCTION

What makes an LLM multi-agent workflow secure? Benign messages and normal agent behavior motivate communication filters and graph-based anomaly detectors (Wang et al., 2025; Miao et al., 2026; Pan et al., 2026). Yet benign components do not define secure collaboration. Untrusted content and poisoned memories can redirect planning and execution (Zhan et al., 2024; Chen et al., 2024), making security depend on execution paths and authority across interacting agents and tools, not content alone (Debenedetti et al., 2025). Even without overtly malicious text, a workflow may execute unsafe actions when provenance, delegation, shared state, and authority interact. We therefore define *secure collaboration* as preventing untrusted or mis-scoped information from reaching protected side effects through candidate execution dependencies, while preserving those needed to complete the user task under changing operational conditions.

This distinction matters as multi-agent systems move from discussion to execution. A modern workflow retrieves external documents, queries tools, writes shared memory, delegates subtasks, and invokes privileged APIs. Consider a procurement task in which a retrieval agent summarizes a plausible bank-account update, a tool agent returns a forged verification result, and a planner delegates payment to a finance agent. Each event can be locally consistent with the task. The harm emerges only when the external claim is treated as evidence, combined with the forged result, bound to the payment arguments, and executed under the finance agent's authority. Removing the finance agent destroys a legitimate business function; filtering the summary alone may miss the attack; tracing only a source-to-sink path does not reveal which available intervention will preserve the task. The security-relevant object is thus the relation composition untrusted evidence $\rightarrow$ semantic derivation $\rightarrow$ delegation $\rightarrow$ authority amplification $\rightarrow$ privileged effect.

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.

Existing defenses address useful but incomplete parts of this process. Communication/graph anomaly detectors excel at identifying malicious utterances or agents (He et al., 2025; Wang et al., 2025; Miao et al., 2026; Pan et al., 2026; Zhou et al., 2025); long-horizon state trackers preserve security context across extended trajectories (Wang et al., 2026; Xu et al., 2026); provenance and information-flow systems trace evidence to actions or restrict flows via deterministic labels (Cai et al., 2026; She et al., 2026; Costa et al., 2025; Girrens & Wang, 2026). Each cluster establishes a strong substrate for detection, provenance, or control. However, none learns which *combination of executable, typed interventions* will change the outcome of a stochastic multi-agent workflow when multiple defensive actions must interact, with the least loss of task utility. This gap between dependency tracking and actionable intervention is precisely where we focus.

Detection without tracing yields a verdict without an actionable boundary; tracing without intervention confuses observed dependency with intervention efficacy; isolation without utility prediction can disable essential agents; recovery without state update allows the same contamination to re-enter later tasks. This coupling leads to our central question: *How can a defense learn which bounded sets of typed interventions change an unsafe outcome over candidate execution dependencies, while preserving task utility in an asynchronous multi-agent workflow?*

We answer this question with RACG, a *relation-aware collaboration graph* framework. RACG represents a workflow as a typed execution-dependency event graph whose nodes include agent invocations, messages, retrievals, tool events, memory records, delegations, proposed actions, and environmental outcomes. Its edges distinguish delivery, derivation, consumption, delegation, memory versioning and conflicts, authorization, resource order, retry, and external effect. The graph is an execution substrate and intervention interface; it is not itself our methodological claim. Our central mechanism is *intervention-conditioned affected-cone learning*: a typed action is injected at its target event, its representation propagates only through descendants that can reach the pending privileged sink, and interactions are modeled where the cones of multiple interventions meet.

Offline checkpoint replays supervise the model with the safety and utility effects of message quarantine, memory-version restoration, and capability narrowing. These interventions are executable before the outcome is known and never substitute oracle answers. RACG then derives a candidate risk-propagation subgraph from its targets, applies the selected actions, and regenerates the affected continuation from an action-valid rollback cut. Repeated counterfactual replay is used for supervision and evaluation, not for every online candidate. If uncertainty remains high, the system fails closed or requests human confirmation rather than proceeding without verification.

Our contributions are threefold:

- **Problem formulation.** We formulate multi-agent safety as relation-conditioned intervention over a stochastic graph-generating workflow, separating the population reference intervention from its deployable eMEIS approximation, the candidate risk-propagation subgraph, and the special case of a graph-theoretic repair cut.
- **Primary method.** We develop typed intervention-effect learning over endogenous stochastic suffixes, eMEIS selection, and candidate risk-propagation-subgraph recovery. Affected-cone propagation, paired safety–utility supervision, uncertainty estimation with held-out calibration, and bounded coalition search implement this single mechanism across detection, tracing, isolation, recovery, and update.
- **Supporting evaluation protocol.** We design relation-matched and coalition scenarios generated by the orchestrator before execution, and compare against text-only models, relation-ablated models, provenance reachability, information-flow control, policy-aware min-cut, and whole-agent isolation. Across nine benchmarks, RACG reduces macro attack success from 15.54% to 13.81% relative to Full-Graph Late Fusion, an 11.13% relative reduction, while retaining 79.08% benign-task success. In intervention-set selection, RACG reduces regret from 0.131 to 0.071 relative to Beam + Additive Scores, a 45.80% relative reduction.

2 PRELIMINARIES

Execution graph. Let $q \sim \mathcal{D}$ be a user task and s_0 a trusted initial state. At a consistent decision cut d, an asynchronous multi-agent orchestrator has observed events $\tau_{\leq d}$ with happens-before order

$\prec_d$. We encode this prefix as a typed execution-dependency event graph

$$\mathcal{G}_d = (\mathcal{V}_d, \mathcal{E}_d, \mathbf{X}_d). \tag{1}$$

Each node is an event—an agent invocation, message, retrieval, tool call or return, memory access, delegation, authorization, proposed action, committed action, or environment outcome—rather than an abstract agent. The canonical relations are DELIVER, DERIVE, DELEGATE, CONSUME, READ-FROM, WRITECONFLICT, VERSION, RETRY, AUTHORIZE, RESOURCEORDER, and EXTERNAL-EFFECT. They are recorded by the trusted orchestrator and denote candidate execution dependencies, not semantic causation.

Decision and rollback cuts. A privileged sink $z^{\text{pending}} \in \mathcal{G}_d$ can change protected external or persistent state and must pass a pre-commit gate. For each valid intervention set A, $r(A) < d$ is the latest trusted cut preceding the earliest event consumption or state change modified by A. The matched full rollouts are

$$\widetilde{\tau}^A = \tau_{\le r(A)} \oplus \tau^A_{>r(A)}, \widetilde{\tau}^{0|A} = \tau_{\le r(A)} \oplus \tau^{0|A}_{>r(A)}, \tag{2}$$

where both continuations start from the same state $s_{r(A)}$. The superscript $0 \mid A$ denotes the unmodified control paired with A, not one global baseline. For the online no-intervention decision, we set $r(\emptyset) = d$ and use $\widetilde{\tau}^{0|d} = \tau_{\le d} \oplus \tau^0_{>d}$. A concurrent branch is reused only when its read/write footprint is disjoint and its version and conflict relations remain valid; otherwise it is revalidated or regenerated.

Outcomes, actions, and effects. Independent machine-verifiable evaluators define violation $Y(\widetilde{\tau}) \in \{0, 1\}$ and task utility $U(\widetilde{\tau})$. The safety evaluator checks actual side effects against hidden benchmark and policy truth; the utility evaluator checks answer or environment-state correctness rather than mere completion. Hidden terminal truth is unavailable to the online defender, and neither evaluator uses an LLM judge as its sole label source. An atomic action is $a = (v, \tau, \pi, c)$: an event target, actuator type, replacement policy available before terminal truth, and composite operational cost. Edge-target actions are outside our first-paper scope. Replay identifies the operational intervention effects

$$\rho_Y(A \mid \mathcal{G}_d) = \mathbb{E}[Y(\widetilde{\tau}^{0|A}) - Y(\widetilde{\tau}^A) \mid \mathcal{G}_d], \tag{3}$$

$$\Delta_U(A \mid \mathcal{G}_d) = \mathbb{E}[U(\widetilde{\tau}^{0|A}) - U(\widetilde{\tau}^A) \mid \mathcal{G}_d]. \tag{4}$$

These are effects of executable actions, not unique causes inside an LLM.

Reference and explanation objects. On exactly enumerated diagnostic instances, the population reference $A^\dagger$ minimizes $C(A) + \beta\Delta_U(A \mid \mathcal{G}_d)$ over the declared frontier, actuator catalog, and $|A| \le K$, subject to $\Pr(Y(\widetilde{\tau}^A) = 1 \mid \mathcal{G}_d) \le \epsilon$. Let $u \rightsquigarrow^* v$ denote reflexive directed reachability in $\mathcal{G}_d$. For a selected set A, the candidate risk-propagation subgraph contains nodes

$$\mathcal{V}_{\text{risk}}(A, z) = \bigcup_{a \in A} (\{u : u \rightsquigarrow^* t(a)\} \cup \{v : t(a) \rightsquigarrow^* v \rightsquigarrow^* z\}) \tag{5}$$

and exactly the typed edges on the qualifying upstream-to-target and target-to-sink paths. This deterministic dependency explanation is not a causal subgraph. A repair cut is the special case in which every selected actuator blocks an edge and those edges disconnect all implicated sources from z; memory restoration and capability narrowing need not form a cut.

3 METHODOLOGY

3.1 PROBLEM FORMULATION

At decision cut d, RACG receives a typed execution-dependency graph $\mathcal{G}_d$, a pending privileged sink z, and an executable intervention catalog $\mathcal{A}(\mathcal{G}_d, z)$. Its goal is to prevent an unsafe sink while preserving task utility. The threat setting covers indirect prompt or document injection, forged or delayed tool feedback, poisoned shared memory, and messages from compromised low-privilege agents; the attacker need not control every agent or access model parameters. We trust the orchestrator and policy control plane to record event identity, delivery, timing, authorization, state versions,

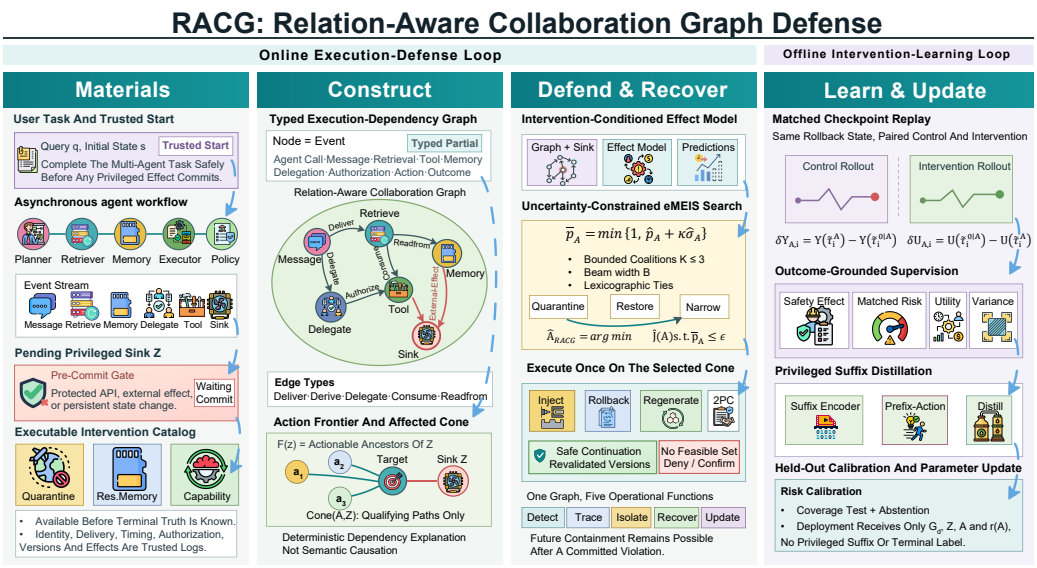


Figure 1: RACG couples online graph-based defense and recovery with offline intervention-effect learning.

and committed effects, while treating message, retrieval, and tool-return content as potentially compromised. Deterministic policy enforcement handles explicit scope violations; RACG addresses task-conditioned evidence composition and security–utility decisions.

For each candidate set A, the intervention model conditions on the observed graph, sink, action semantics, and action-valid rollback cut:

$$f_\theta(\mathcal{G}_d, z, A, r(A)) \to (\widehat{p}_A, \widehat{p}_{0|A}, \widehat{\Delta U}_A, \widehat{\sigma}_A). \tag{6}$$

The outputs represent post-intervention violation risk, matched-control risk, predicted utility loss, and predictive uncertainty. RACG couples two pillars around this model: the online execution-defense loop selects and executes interventions with fixed parameters, while the offline intervention-learning loop learns their effects from matched replays and training-only suffix evidence. This separation allows deployment to act on observed collaboration history without accessing future outcomes.

3.2 Online Execution-Defense Loop

The online pillar converts collaboration history into an executable defense whenever a privileged sink becomes pending. It first identifies where intervention remains possible, then estimates the effects of candidate action sets, and finally selects a feasible set and regenerates the affected continuation.

Relation-aware collaboration graph. Each event carries its type, content, source, permission, time, version, and history. Relation-specific message passing follows the canonical schema in Section 2, preserving distinctions among message delivery, memory access, delegation, authorization, retry, resource order, and external effects. These relations provide the context needed to distinguish information that merely appears in a workflow from information that can influence a pending privileged action. For sink z, the actionable frontier $\mathcal{F}(z)$ contains only ancestors admitting a valid actuator before the sink commits. RACG instantiates QuarantineMessage, RestoreMemory, and NarrowCapability on this frontier, excluding invalid or irreversible actions. The graph therefore constrains where defense can operate, while the catalog specifies what can actually be changed; neither supplies an oracle-correct replacement value.

Affected-cone intervention scoring. Reachability identifies candidate influence paths but does not determine whether changing them will improve the outcome. RACG consequently conditions its predictions on both the typed graph and the semantics of the proposed actions. For action a_j targeting event v_j, its affected cone comprises observed descendants that also precede the sink,

together with the target and sink themselves:

$$\mathrm{Cone}(a_j, z) = \big(\mathrm{Desc}(v_j) \cap \mathrm{Anc}(z)\big) \cup \{v_j, z\}. \tag{7}$$

For a set A, the encoder operates on the union $\mathrm{Cone}(A, z) = \bigcup_{a \in A} \mathrm{Cone}(a, z)$. RACG injects each full action representation $\mathbf{e}_{a_j}$ at its target and aggregates the action embeddings whose cones contain event v. This aggregation is permutation invariant, so a coalition is represented as a set rather than an arbitrary action sequence:

$$\mathbf{g}_v(A) = \phi_{\mathrm{set}}\left(\sum_{a_j : v \in \mathrm{Cone}(a_j, z)} \phi(\mathbf{e}_{a_j})\right). \tag{8}$$

The encoder propagates $[\mathbf{h}_v \| \mathbf{g}_v(A)]$ along typed edges within the union cone. At cone intersections, the representation combines the relevant actions with their shared dependency context, allowing the model to represent complementary or redundant effects rather than assuming that singleton effects add. The resulting representation supplies the risk, utility, and uncertainty predictions used for selection. These cones remain candidate execution dependencies, not assertions of semantic causation.

Uncertainty-constrained intervention and recovery. RACG evaluates candidate sets using the calibrated upper risk score $\overline{p}_A = \min\{1, \widehat{p}_A + \kappa \widehat{\sigma}_A\}$ and the composite objective $\widehat{J}(A) = C(A) + \beta \widehat{\Delta U}_A$. The former determines feasibility under the safety threshold ϵ; the latter trades intervention cost against predicted utility loss among feasible candidates. Search is restricted to coalitions of size at most $K \leq 3$ and beam width B, yielding the deployable selection

$$\widehat{A}_{\mathrm{RACG}} = \operatorname*{arg\,min}_{A \in \mathcal{B}_{K,B}(\mathcal{G}_d, z)} \widehat{J}(A) \quad \text{s.t.} \quad \overline{p}_A \leq \epsilon. \tag{9}$$

This eMEIS is relative to the declared frontier, three-actuator catalog, search budget, objective, and upper-risk rule; it need not equal the population reference $A^\dagger$. Ties are resolved lexicographically by cost, predicted utility loss, cardinality, and fixed action identifier. The selected targets define the candidate risk-propagation subgraph. RACG restores the action-valid rollback state, applies the selected intervention set once, and regenerates the affected continuation before two-phase commit. Together, these operations implement detection, tracing, isolation, recovery, and update within one online loop. If no set is feasible, the sink is denied or routed for confirmation. After a committed violation, the graph supports future containment and reversible cleanup, but cannot undo an irreversible disclosure or transfer.

3.3 Offline Intervention-Learning Loop

The offline pillar learns the outcome changes that graph reachability alone cannot establish. It constructs matched intervention examples, supervises their safety–utility effects, and distills training-only continuation evidence into a model whose deployment inputs remain limited to the observed prefix and proposed actions.

Matched checkpoint replay. For each context $x_A = (\mathcal{G}_d, z, A, r(A))$, the intervention and matched control restore the same state $s_{r(A)}$ and generate $R \geq 2$ paired full trajectories. Let $Y_{A,i}$ and $Y_{0|A,i}$ denote their violation outcomes, with utility scores defined analogously. The paired differences $\delta Y_{A,i} = Y_{A,i} - Y_{0|A,i}$ and $\delta U_{A,i} = U_{0|A,i} - U_{A,i}$ record the observed violation change and utility loss. Averaging repeated continuations yields empirical effect estimates:

$$\begin{bmatrix} \bar{\rho}_{Y,A} \\ \bar{\Delta}_{U,A} \end{bmatrix} = \frac{1}{R} \sum_{i=1}^{R} \begin{bmatrix} -\delta Y_{A,i} \\ \delta U_{A,i} \end{bmatrix}. \tag{10}$$

Their population counterparts are $\rho_Y(A \mid \mathcal{G}_d) = \mathbb{E}[-\delta Y_A \mid x_A]$ and $\Delta_U(A \mid \mathcal{G}_d) = \mathbb{E}[\delta U_A \mid x_A]$: positive values indicate reduced violation risk and lost utility, respectively. The barred quantities are replay estimates rather than model predictions. Matching the rollback state compares applying A with leaving that same state unmodified, while repeated continuations expose stochastic outcome variation. Replay is used for offline supervision and evaluation, not to execute every candidate during online selection.

Outcome-grounded intervention supervision. The model learns absolute risks and paired effects jointly, so that intervention selection is grounded in observed workflow outcomes rather than reachability alone. For each replay pair, the supervised loss combines intervention and matched-control

binary cross-entropies, squared error between their predicted risk difference and $\delta Y_{A,i}$, and a Huber loss for predicted utility loss. Writing these terms as $\ell_{\text{out}}^{A,i}$, $\ell_{\text{pair}}^{A,i}$, and $\ell_U^{A,i}$, respectively, gives

$$\mathcal{L}_{\text{sup}}(\theta) = \mathbb{E}_{x_A \sim \mathcal{D}_{\text{rep}}} \left[\frac{1}{R} \sum_{i=1}^{R} \ell_{\text{sup}}^{A,i} \right]. \tag{11}$$

Here $\ell_{\text{sup}}^{A,i} = \ell_{\text{out}}^{A,i} + \lambda_{\text{pair}} \ell_{\text{pair}}^{A,i} + \lambda_U \ell_U^{A,i}$, with $\ell_{\text{pair}}^{A,i} = (\widehat{p}_A - \widehat{p}_{0|A} - \delta Y_{A,i})^2$ and $\ell_U^{A,i} = \ell_{\text{Huber}}(\widehat{\Delta U}_A, \delta U_{A,i})$. The outcome terms anchor both risk predictions, while the paired and utility terms supervise changes associated with executable interventions. The online no-intervention score remains distinct: $\widehat{p}_d^0 = [f_\theta(x_\emptyset)]_{\text{risk}}$ uses $x_\emptyset = (\mathcal{G}_d, z, \emptyset, d)$, whereas the action-conditioned control starts at $r(A)$. This distinction preserves the matched comparison without equating rollback-based controls with the current deployment state. Replay sampling details appear in Appendix A.2.

Privileged-suffix distillation and uncertainty calibration. During training only, a suffix encoder observes the realized continuation after $r(A)$ and produces a representation that supervises the prefix–action model. The suffix loss $\mathcal{L}_{\text{suf}}$ averages squared Euclidean distances between the prefix-predicted representation and the stop-gradient suffix target over the same contexts and replay samples. Alongside outcome supervision, RACG learns conditional outcome variance and regularizes equivalent serializations of the typed dependency structure:

$$\mathcal{L}_{\text{offline}} = \mathcal{L}_{\text{sup}} + \lambda_{\text{suf}} \mathcal{L}_{\text{suf}} + \lambda_{\text{var}} \mathcal{L}_{\text{var}} + \lambda_{\text{inv}} \mathcal{L}_{\text{poset}}. \tag{12}$$

The variance loss averages the Huber error between $\widehat{\sigma}_{\text{ale}}^2(A)$ and the sample variance of the R intervention outcomes. The partial-order term $\mathcal{L}_{\text{poset}}$ enforces consistency only across equivalent serializations of the same typed execution dependencies, not arbitrary event reorderings (Appendix A.2). These auxiliary terms enrich the deployment-visible representation without requiring access to realized continuations at inference.

For $H \geq 2$ bootstrapped risk heads, $\widehat{p}_A$ is their mean prediction and $\widehat{\sigma}_A^2$ adds their sample variance, computed with denominator $H - 1$, to the predicted conditional outcome variance $\widehat{\sigma}_{\text{ale}}^2(A)$. Matched-control predictions are averaged analogously. Neither uncertainty component is the variance of paired differences, which is reserved for paired-effect confidence intervals. Held-out calibration selects κ for the online upper risk score; insufficient coverage increases abstention rather than relaxing ϵ. Offline acquisition prioritizes unobserved pairs and triples using bootstrapped variation of the same composite cost–utility objective. Exact variance targets and calibration rules appear in Appendix A.3. After training, only the intervention model returns to the online pillar; privileged suffixes and terminal labels remain unavailable at deployment.

4 Experiments

Our evaluation asks whether learning typed intervention effects improves multi-agent defense, and which RACG components contribute. We first examine end-to-end security and task utility across nine benchmarks (RQ1), then study relation discrimination (RQ2), joint intervention-effect estimation and set selection (RQ3), operational reliability (RQ4), and transfer to unseen settings (RQ5).

4.1 Experimental Setup

Tasks and evaluation settings. We evaluate mathematical reasoning on GSM8K, SVAMP, AQuA, and MATH; code generation on HumanEval and MBPP; knowledge on MMLU; multi-hop QA on HotpotQA; and browsing on BrowseComp+. Task correctness and attack success are evaluated separately, since correct outputs do not imply policy-compliant collaboration. Diagnostics cover relation-matched examples, intervention coalitions, operational perturbations, five distribution shifts.

Baselines. We compare text-only detection, G-Safeguard, BlindGuard, XG-Guard, GUARDIAN, ProvenanceGuard, and SPA with policy-aware minimum cut, Full-Agent Isolation, Greedy Singleton, and Full-Graph Late Fusion. Diagnostics include Bag-of-Events, Homogeneous Graph, Relation-Shuffled, HeteroGNN + SetEnc, and additive estimators, separating relational modeling from intervention prediction and selection. AiTM is an attack setting, not a defense baseline.

Metrics. We report attack success rate (ASR↓), defended task success (DTS↑), benign task success (BTS↑), and normalized intervention cost, with macro scores averaged over nine tasks. Diagnostics

Table 1: Overall security and task performance on nine benchmarks. Panel A reports ASR (↓) and Panel B DTS (↑). Best and second-best values are **bold** and underlined. The Δ row reports RACG's improvement over Full-Graph Late Fusion; positive values favor RACG.

Panel A: Attack Success Rate (ASR ↓)

Method	GSM8K	SVAMP	AQuA	MATH	HumanEval	MBPP	MMLU	HotpotQA	BrowseComp+	Macro
No defense	24.63	22.91	27.48	30.76	33.58	31.27	26.94	41.73	52.36	32.41
Text-only + adapter	17.84	16.27	20.43	23.91	26.85	24.36	19.52	33.46	44.28	25.21
G-Safeguard + adapter	13.92	12.68	16.73	19.84	21.37	19.71	15.86	27.14	37.92	20.57
BlindGuard + adapter	13.37	12.14	16.05	18.92	20.46	18.83	15.21	25.63	35.84	19.61
XG-Guard + adapter	12.56	11.73	15.41	18.16	19.62	18.27	14.66	23.48	33.19	18.56
GUARDIAN + adapter	13.11	12.47	15.87	18.73	20.04	18.59	14.92	24.17	34.07	19.11
ProvenanceGuard + adapter	12.84	11.96	15.08	17.92	18.86	17.71	14.23	21.64	30.58	17.87
SPA + adapter	9.74	9.21	12.38	14.56	15.73	14.18	11.62	16.37	22.81	14.07
Policy-aware minimum cut	10.43	9.87	13.12	15.09	16.41	14.96	12.08	17.06	23.47	14.72
Full-Agent Isolation	**8.92**	**8.17**	**10.61**	**12.47**	**13.28**	**11.94**	**9.86**	**13.74**	**18.93**	**11.99**
Greedy Singleton	11.73	10.88	14.26	16.87	17.54	16.32	13.37	20.19	28.61	16.64
Full-Graph Late Fusion	10.86	10.21	13.48	15.74	16.82	15.69	12.71	18.47	25.92	15.54
RACG	9.63	9.42	12.91	14.27	15.39	14.46	11.84	15.62	20.76	13.81
Δ *vs. Full-Graph Late Fusion*	*+1.23*	*+0.79*	*+0.57*	*+1.47*	*+1.43*	*+1.23*	*+0.87*	*+2.85*	*+5.16*	*+1.73*

Panel B: Defended Task Success (DTS ↑)

Method	GSM8K	SVAMP	AQuA	MATH	HumanEval	MBPP	MMLU	HotpotQA	BrowseComp+	Macro
No defense	**84.72**	**87.16**	**73.85**	**67.24**	**71.46**	**75.83**	**78.62**	61.48	18.74	68.79
Text-only + adapter	83.96	86.52	72.98	66.71	70.54	74.91	77.88	62.73	20.13	68.48
G-Safeguard + adapter	83.54	85.87	72.41	66.02	70.93	74.62	77.31	64.08	21.86	68.52
BlindGuard + adapter	83.12	85.43	71.96	65.71	70.17	74.11	76.84	64.72	22.41	68.27
XG-Guard + adapter	83.37	85.62	72.24	65.93	70.68	74.39	77.02	65.81	23.57	68.74
GUARDIAN + adapter	83.49	85.74	72.06	66.15	70.35	74.28	77.16	65.26	22.93	68.60
ProvenanceGuard + adapter	82.68	84.93	71.43	65.14	69.72	73.54	76.31	66.72	24.68	68.35
SPA + adapter	78.46	80.67	68.92	62.13	66.47	70.28	72.84	64.95	21.17	65.10
Policy-aware minimum cut	79.71	81.86	69.54	62.88	67.31	71.07	73.59	65.63	22.06	65.96
Full-Agent Isolation	72.36	74.82	63.14	56.83	61.72	65.27	68.19	58.41	15.82	59.62
Greedy Singleton	82.94	85.18	71.82	65.47	70.42	74.06	76.58	66.14	25.37	68.66
Full-Graph Late Fusion	83.08	85.37	72.17	65.86	70.96	74.53	76.94	67.38	27.48	69.31
RACG	83.44	85.69	72.66	66.48	71.52	75.17	77.43	**69.84**	**30.61**	**70.32**
Δ *vs. Full-Graph Late Fusion*	*+0.36*	*+0.32*	*+0.49*	*+0.62*	*+0.56*	*+0.64*	*+0.49*	*+2.46*	*+3.13*	*+1.01*

report pair consistency, balanced accuracy, MAE, synergy-sign accuracy, equivalent-set coverage, and normalized regret. Reliability uses ECE, UCB coverage, false feasibility, invalid actions, and stress degradation. Percentage-point changes differ from relative cost and latency changes.

4.2 Main Results: Security and Task Utility (RQ1)

Learned intervention improves the security–utility trade-off. As shown in Table 1, RACG achieves 13.81% macro ASR and 70.32% macro DTS, improving over Full-Graph Late Fusion (15.54%, 69.31%) while reducing intervention cost from 0.63 to 0.49 (22.2%). Compared with XG-Guard + adapter, RACG further reduces ASR by 4.75 points and improves DTS by 1.58 points, supporting intervention-effect modeling beyond suspicious-event detection. **Gains concentrate on retrieval and browsing tasks.** Relative to Full-Graph Late Fusion, RACG reduces ASR by 2.85/5.16 points and improves DTS by 2.46/3.13 points on HotpotQA/BrowseComp+, respectively. Across the other seven tasks, the corresponding gains average only 1.08/0.50 points. This pattern is consistent with relation-aware defense benefiting extended workflows, although task-level results alone do not establish propagation depth as the cause. **Lower ASR does not imply a better operating point.** Full-Agent Isolation achieves lower macro ASR (11.99%) but substantially lower DTS (59.62%) and higher cost (1.18). RACG retains 10.70 points more DTS for 1.82 points higher ASR.

4.3 Relation Discrimination and Joint Intervention (RQ2 and RQ3)

Typed relations improve discrimination and downstream defense. On the relation diagnostic in Figure 2, RACG achieves 76.41% composition pair consistency and 84.33% overall pair consistency. The Homogeneous Graph baseline reaches 53.72% and 62.50%, while the Relation-

Table 2: Joint intervention-effect estimation and executable intervention-set selection (RQ3).

Effect estimation

Estimator	Risk S ↓	Risk P ↓	Risk T ↓	Utility S ↓	Utility P ↓	Utility T ↓	Syn. Acc. ↑	ECE ↓
Singleton Additive	0.094	0.163	0.218	0.082	0.141	0.196	57.84	0.091
HeteroGNN + SetEnc	0.071	0.119	0.158	0.066	0.107	0.143	68.37	0.067
Full-Graph Late Fusion	0.064	0.104	0.139	0.061	0.094	0.126	72.91	0.058
RACG w/o Cone Intersection	0.061	0.112	0.151	0.059	0.101	0.134	66.48	0.055
RACG w/o Privileged Suffix	0.069	0.097	0.128	0.064	0.089	0.117	74.26	0.079
RACG	0.062	0.086	0.113	0.058	0.081	0.104	80.73	0.046

Intervention-set selection

Selector	Eq. Cov. ↑	Top-1 ↑	Regret ↓	Eval.	ASR ↓	DTS ↑
Random Feasible	17.63	8.94	0.284	24.71	22.46	69.38
Greedy Singleton	41.82	27.36	0.176	19.43	15.87	73.64
Beam + Additive Scores	54.27	36.91	0.131	31.68	13.94	74.71
Exact + RACG Scores	82.46	66.18	0.054	100.00	9.86	76.48
RACG eMEIS	77.38	59.73	0.071	27.54	10.47	75.92
Finite-Replay Ref.	100.00	100.00	0.00	100.00	8.91	77.13

Shuffled variant reaches 50.18% and 59.77%. In the associated natural relation-compositional workflows, RACG obtains 18.34% ASR and 72.41% DTS, improving on Homogeneous Graph by 5.47 and 3.84 points, respectively, with nearly unchanged BTS (79.74% vs. 79.68%). The agreement between relational discrimination and downstream results supports the use of typed collaboration structure. **Affected-cone modeling is most useful for joint actions.** Table 2 shows that RACG reduces pair/triple risk MAE from 0.104/0.139 with Full-Graph Late Fusion to 0.086/0.113, reductions of 17.3%/18.7%, versus only 3.1% for singletons. Removing the cone intersections has little effect on singleton MAE (0.061), but raises pair/triple MAE to 0.112/0.151 and the lowers synergy-sign accuracy from 80.73% to 66.48%. The gain therefore centers on joint-action interactions. Removing privileged-suffix supervision also raises ECE from 0.046 to 0.079 and triple MAE from 0.113 to 0.128. **Search efficiency trades off against selection fidelity.** RACG eMEIS achieves 77.38% equivalent-set coverage and 0.071 regret while evaluating 27.54% of candidate sets. Exact search with the same scores reaches 82.46% coverage and 0.054 regret but requires exhaustive evaluation. Thus, eMEIS cuts evaluations by 72.46% for only 0.61 ASR and 0.56 DTS points. The finite-replay reference reaches 8.91% ASR and 77.13% DTS, indicating residual error without implying population-optimal recovery.

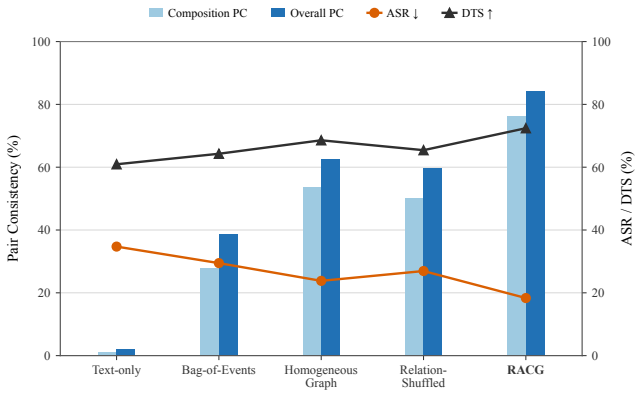


Figure 2: Relation-aware modeling improves relation discrimination and downstream defense.

4.4 Operational Robustness and Reliability (RQ4)

Candidate noise causes measurable but smaller degradation. In Figure 3, doubling the pool with irrelevant actions raises RACG's ASR by 3.18 points and reduces DTS by 2.41 points, compared with 5.43/4.16 points for Full-Graph Late Fusion. Delay and reordering increase ASR by 2.43 versus 4.71 points. Misleading actions are harder, increasing RACG's ASR by 5.72 points and reducing DTS by 4.63 points. Doubling irrelevant candidates also raises latency by 34.86%, revealing a computational cost to robustness. **Calibration, validity, and rollback affect different failure modes.** Removing UCB increases false feasibility from 2.18% to 6.73% without changing ECE (0.05), while removing calibration lowers UCB coverage from 94.21% to 86.37% and raises ASR from 13.81% to

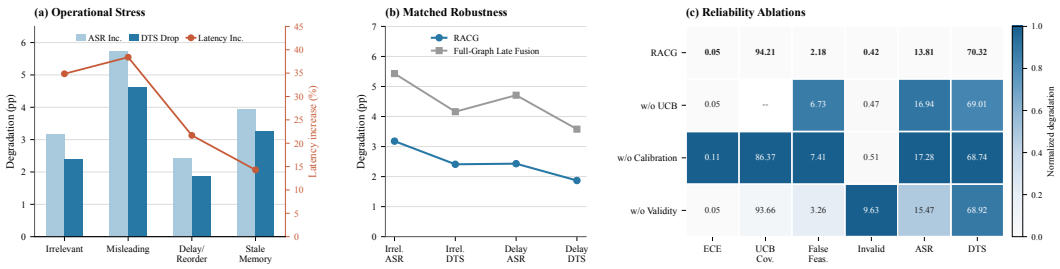


Figure 3: Operational stress degrades defense performance, while removing reliability components further exposes vulnerabilities.

17.28%. Without validity filtering, invalid actions rise from 0.42% to 9.63%; without matched rollback, ASR reaches 18.61%. These results separate risk-estimation from execution-semantic failures; UCB coverage remains diagnostic rather than a zero-violation guarantee.

4.5 Transfer beyond the Training Setting (RQ5)

Frozen policies retain the benefits under structural shifts. Across four within-backbone shifts, Frozen RACG reduces ASR by 6.33-6.91 points and improves DTS by 5.08-5.65 points over Fixed Policy, showing that its operating-point gains persist under unseen structures. **Cross-backbone transfer remains incomplete.** Frozen RACG achieves 23.17% ASR and 64.82% DTS; recalibration improves these to 20.84%/66.73% and reduces ECE from 0.14 to 0.07. Target training remains stronger (16.21%/70.86%), limiting claims of backbone independence. **Composite-objective transfer efficiency.** We define RTE $= (\mathcal{J}_{\text{transfer}} - \mathcal{J}_{\text{fixed}})/(\mathcal{J}_{\text{target}} - \mathcal{J}_{\text{fixed}})$ for the fixed security–utility objective $\mathcal{J}$. Frozen RACG obtains RTE 0.61–0.71 within backbone but 0.44 cross-backbone, increasing to 0.60 after recalibration. RTE reflects the composite objective, not ASR or DTS alone.

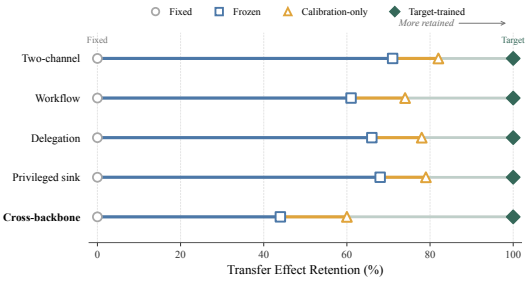


Figure 4: RACG retains security and utility gains under distribution shifts, but cross-backbone transfer remains challenging.

5 Conclusion

We presented RACG, a relation-aware collaboration-graph framework that couples online intervention and recovery with offline learning from matched replay and privileged suffixes. Rather than treating security as isolated message or agent detection, RACG models how executable typed interventions change downstream collaboration outcomes and uses these estimates to select intervention sets under risk and resource constraints. This design connects detection, tracing, isolation, recovery, and state update within a common execution-level defense framework.

Across nine task benchmarks, RACG improves the reported security–utility operating point over Full-Graph Late Fusion, reducing macro ASR from 15.54% to 13.81% while increasing DTS from 69.31% to 70.32%. Relation and coalition diagnostics further indicate that typed dependencies become particularly useful for joint interventions, where affected-cone modeling better captures non-additive effects. Stress and transfer evaluations additionally highlight the roles of calibration, execution validity, and rollback semantics. Together, these results support learning the consequences of intervention, rather than only detecting suspicious events, as a practical basis for defending multi-agent workflows. Several directions remain open. RACG assumes trusted control-plane instrumentation, observable typed events, and recoverable execution states. Future work can relax these assumptions through incomplete or corrupted event graphs, partially trusted controllers, and workflows with irreversible external effects. Expanding the actuator catalog may enable finer-grained responses, while longer and more heterogeneous workflows can test whether affected-cone localization scales with growing execution histories and candidate spaces.

REPRODUCIBILITY STATEMENT

We provide the implementation details of RACG in the main text and appendix. All experiments use the GSM8K, SVAMP, AQuA, MATH, HumanEval, MBPP, MMLU, HotpotQA, and BrowseComp+ benchmarks. Code and configurations of RACG will be released for reproduction.

AI USE STATEMENT

Generative AI tools were not used in the preparation of this manuscript or in the development of the method. RACG is designed for LLM-based multi-agent systems and therefore uses generative models as experimental backbones and agents in the evaluated workflows. This use is part of the evaluated methodology rather than assistance in manuscript writing or technical development.

REFERENCES

Yuandao Cai, Wensheng Tang, Cheng Wen, and Shengchao Qin. Ghost in the agent: Redefining information flow tracking for LLM agents. *arXiv preprint arXiv:2604.23374*, 2026.

Zhaorun Chen, Zhen Xiang, Chaowei Xiao, Dawn Song, and Bo Li. AgentPoison: Red-teaming LLM agents via poisoning memory or knowledge bases. In *Advances in Neural Information Processing Systems*, volume 37, 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/eb113910e9c3f6242541c1652e30dfd6-Abstract-Conference.html.

Manuel Costa et al. Securing AI agents with information-flow control. *arXiv preprint arXiv:2505.23643*, 2025.

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. doi: 10.48550/arXiv.2503.18813. URL https://arxiv.org/abs/2503.18813.

Dylan Girrens and Guangjing Wang. SPA: Securing persistent LLM agents across queries with plan-first information-flow control. *arXiv preprint arXiv:2608.27234*, 2026.

Pengfei He, Yuping Lin, Shen Dong, Han Xu, Yue Xing, and Hui Liu. Red-teaming LLM multi-agent systems via communication attacks. In *Findings of the Association for Computational Linguistics: ACL 2025*, 2025.

Rui Miao, Yixin Liu, Yili Wang, Xu Shen, Yue Tan, Yiwei Dai, Shirui Pan, and Xin Wang. BlindGuard: Safeguarding LLM-based multi-agent systems under unknown attacks. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics*, 2026.

Junjun Pan, Yixin Liu, Rui Miao, Kaize Ding, Yu Zheng, Quoc Viet Hung Nguyen, Alan Wee-Chung Liew, and Shirui Pan. Explainable and fine-grained safeguarding of LLM multi-agent systems via bi-level graph anomaly detection. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics*, 2026.

Yining She, Yiliang Liang, and Eunsuk Kang. Safeguarding LLM agents from misalignment through provenance analysis. In *Proceedings of the 41st IEEE/ACM International Conference on Automated Software Engineering*, 2026.

Shilong Wang, Guibin Zhang, Miao Yu, Guancheng Wan, Fanci Meng, Chenghe Guo, Kun Wang, and Yan Wang. G-Safeguard: A topology-guided security lens and treatment on LLM-based multi-agent systems. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics*, 2025.

Yuhui Wang, Tanqiu Jiang, Jiacheng Liang, Charles Fleming, and Ting Wang. MAGE: Safeguarding LLM agents against long-horizon threats via shadow memory. *arXiv preprint arXiv:2605.03228*, 2026.

Table 3: Notation and the objects that must remain distinct.

Symbol	Meaning
$(\mathcal{G}_d, z)$	Observed typed execution graph and pending privileged sink.
$a = (v, \tau, \pi, c)$	Target event, actuator type, deployment-available policy, and cost.
$A, r(A)$	Proposed action set and its action-valid rollback cut.
$\widetilde{\tau}^A, \widetilde{\tau}^{0\mid A}$	Full intervention and matched-control trajectories.
Y, U	Binary policy-violation outcome and task utility.
$\rho_Y(A), \Delta_U(A)$	Expected safety gain and expected utility loss under matched replay.
$\widehat{p}_A, \widehat{p}_{0\mid A}$	Predicted intervention risk and action-conditioned control risk.
$\widehat{p}_d^0$	Online no-intervention risk starting from the current cut d.
$A^\dagger, \widehat{A}$	Population reference and deployable, budget-relative eMEIS.

Haowen Xu, Xue Tan, Lei Ma, Zhihao Zhang, Chao Wang, Qingze Wang, Ping Chen, Jun Dai, and Xiaoyan Sun. When agents go rogue: Activation-based detection of malicious behaviors in multi-agent systems. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026.

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. doi: 10.18653/v1/2024.findings-acl.624. URL https://aclanthology.org/2024.findings-acl.624/.

Jialong Zhou, Lichao Wang, and Xiao Yang. GUARDIAN: Safeguarding LLM multi-agent collaborations with temporal graph modeling. *arXiv preprint arXiv:2505.19234*, 2025.

A Formal and Operational Details

RACG defends collaboration at the point where information can still change an executable decision. The defended object is therefore a workflow prefix together with a pending privileged sink, rather than an isolated message. This distinction matters when a document is harmless as reference material but becomes unsafe when treated as authorization, or when a previously correct memory entry is consumed after its scope has changed. This appendix makes the corresponding execution contracts explicit before examining the supplementary evidence.

Trusted boundary and observation model. The orchestrator and policy control plane constitute the trusted boundary. They record which event occurred, which consumer received its output, which capability authorized an action, and which persistent version was read or written. This trust assumption concerns the integrity of the record, not the truth of the recorded payload. A tool return can be faithfully logged and still contain forged evidence. Likewise, an agent's message can have an authenticated sender without being appropriate evidence for a privileged operation.

The attacker may influence inter-agent messages, retrieved documents, tool feedback, or shared memory through the channels admitted by the threat setting. RACG does not assume that the attacker controls every agent, model parameters, or the orchestrator itself. Compromise of the control plane falls outside the stated defense boundary: if delivery receipts or authorization records can be forged without detection, graph-conditioned reasoning loses its trusted reference. A deterministic policy engine remains responsible for explicit scope violations. The learned model supplements that engine for task-conditioned information use; it does not create permission that the engine denies.

An event graph is indexed by a consistent decision cut d. Consistency means that the prefix includes the recorded prerequisites of each included event. An event that arrives late is not inserted into an earlier prefix as if it had already been available. Its relation to a later decision is established through the consumer that actually reads it. Maintaining this distinction prevents offline replay from giving the defender knowledge that the online system did not possess.

Table 4: Canonical typed relations and their operational meaning. Relations denote candidate dependencies, not verified semantic causes.

Relation	Recorded dependency
Deliver	Message or returned content delivered to a consumer.
Derive	Output produced by a recorded transformation of earlier content.
Delegate	Task assignment and the invocation acting under that assignment.
Consume	Content used as an argument or input to an operation.
ReadFrom	A read linked to the precise memory version it observes.
Version	Version succession for a persistent resource.
WriteConflict	Shared-state writes whose ordering affects validity.
Retry	Distinct attempts linked to the attempt retried or superseded.
Authorize	Capability or authorization record bound to an action.
ResourceOrder	Required ordering of operations on a shared resource.
ExternalEffect	A committed action linked to its observed external effect.

Table 5: Actuator contracts. Every replacement is available before terminal truth.

Actuator	Preconditions	State transition and restriction
QuarantineMessage	A trusted checkpoint precedes downstream consumption.	Remove the message from readable context, retain it in the audit history, and regenerate affected consumers.
RestoreMemory	An integrity-admissible version already exists.	Restore that version and revalidate dependent reads; do not synthesize an oracle-correct entry.
NarrowCapability	A pre-authorized scope is a subset of the current scope.	Restrict the capability; empty scope means denial, not successful task completion.

A.1 Canonical Graph Schema and Actuator Preconditions

Nodes represent events, not merely persistent agent identities. Two calls made by the same agent can have different inputs, capabilities, state versions, and risk. Collapsing them into one agent node would hide precisely the distinctions needed to localize an intervention. Node features include event type, payload representation, source identity, permission scope, time, version, and observable history. Source confidence and historical risk are features available at the decision cut, not annotations assigned retrospectively from terminal outcomes.

Typed partial orders also specify which serializations are equivalent. Swapping two independent events can preserve the observed computation; swapping a write and its consuming read generally cannot. Equivalence must preserve read-from bindings, versions, conflicts, retries, authorization, relevant timing, and happens-before order. Consequently, serialization invariance is narrower than invariance to arbitrary graph permutations or event reorderings. A defensive model should ignore incidental serialization while remaining sensitive to changes in executable meaning.

A.1.1 Actuator preconditions and action validity

The actionable frontier contains sink ancestors at which an available actuator can still affect the continuation. Reachability alone does not establish action validity. A message can be an ancestor but already be incorporated into a committed external effect; a memory object can be relevant but have no admissible earlier version; a capability can be too broad without admitting an authorized narrower alternative. Such targets cannot be repaired by inventing a replacement or pretending that an irreversible operation has not occurred.

For a set of individually valid actions, joint validity must also hold. Two proposed restorations of the same resource to incompatible versions do not become executable merely because both actions appear in the catalog. Similarly, a rollback that invalidates a capability's issuing event requires the corresponding authorization to be revalidated. These are operational consistency checks, not

outcomes that a risk predictor can waive. The action-valid cut $r(A)$ is the latest trusted cut preceding the earliest consumption or state change modified by the set.

The catalog carries the full actuator semantics, including its target and admissible replacement policy. Quarantining two different messages is therefore not one interchangeable action type. Conversely, two syntactic action descriptions that induce the same state transition should not create artificial diversity in the candidate pool. Stable identifiers permit deterministic duplicate handling and tie resolution. The method uses three actuator types and coalitions of at most three actions; it does not establish results for unrestricted edits, arbitrary node replacement, or arbitrary permission grants.

The parameterized actuators retain the original operational meanings. QuarantineMessage(e) removes e from readable context, not from the audit record. RestoreMemory(r, k) restores an integrity-admissible version k of resource r that already existed before intervention. Memory invalidation, when admitted by the catalog, is a distinct costed candidate rather than an implicit consequence of restoration. NarrowCapability(g, s') requires a pre-authorized $s' \subset s_g$ derived from the user task and policy before terminal truth; an empty scope denotes denial. These state transitions are deterministic and recoverable within the first-paper sandbox.

A.1.2 Reference interventions and dependency explanations

The population reference, the selected action set, and the explanatory subgraph answer different questions. The population reference minimizes expected cost and utility loss subject to a risk constraint within a declared action domain. The online selection approximates that decision using predictions and a bounded search. The explanatory subgraph records the upstream provenance and downstream candidate dependencies associated with the selected targets. None of these objects is automatically a unique causal explanation of an LLM's internal computation.

Let $t(a)$ denote the target of action a and let $\rightsquigarrow^*$ denote reflexive reachability. The candidate explanation contains the qualifying upstream-to-target and target-to-sink paths:

$$V_{\text{risk}}(A, z) = \bigcup_{a \in A} \left[\{u : u \rightsquigarrow^* t(a)\} \cup \{v : t(a) \rightsquigarrow^* v \rightsquigarrow^* z\} \right]. \tag{13}$$

Only the typed edges belonging to those paths are included. An unrelated edge between two included nodes does not acquire explanatory status simply because both endpoints are present. A graph-theoretic repair cut is a special case in which the selected actuators block edges that disconnect all implicated sources from the sink. Restoring a memory version or narrowing a capability can alter an execution without deleting every information path, so a successful intervention need not be a cut.

A.2 Replay Sampling and Learning Objective

The offline pillar turns observed outcomes into supervision for executable interventions. Its three modules construct matched replay examples, learn safety and utility effects, and distill privileged continuation evidence while calibrating uncertainty. The loss expressions below expand the compact main-paper objective.

A.2.1 Matched checkpoint replay and target construction

For a context $x_A = (\mathcal{G}_d, z, A, r(A))$, the intervention and its control restore the same trusted state $s_{r(A)}$. They then generate complete trajectories with and without applying A. A matched control is action-conditioned because different action sets can require different rollback cuts. Substituting a single global no-defense trajectory would confound the actuator with differences in the state from which generation resumes.

Let $R \geq 2$ index repeated paired continuations. The binary violation labels and task utilities produce the signed targets

$$\delta Y_{A,i} = Y_{A,i} - Y_{0|A,i}, \qquad \delta U_{A,i} = U_{0|A,i} - U_{A,i}, \qquad \widehat{\rho}^{\text{rep}}_{Y,A} = -\frac{1}{R} \sum_{i=1}^{R} \delta Y_{A,i}. \tag{14}$$

Positive safety gain means fewer violations; positive utility difference means lost utility. A negative utility difference is possible when suppressing misleading information improves task completion. It

should not be silently clipped to zero unless a separately specified objective calls for such clipping. The observed replay average and the learned prediction are different estimators and use different notation throughout this appendix.

Paired continuations use common random numbers when the execution backend exposes them. For a closed backend, matched prompts and tool fixtures can be paired by replay index, but this does not imply identical latent randomness. Pairing controls the initial state and environment, while repeated continuations reveal residual outcome variability. It does not make the LLM deterministic or identify a unique internal causal mechanism.

The replay design covers valid singletons, overlapping-cone pairs, and uncertainty-selected triples. Selection of informative coalitions uses only training-pool information and does not inspect outcome labels from the final test set. Related prefixes and their intervention/control continuations are grouped within the same split to prevent the same checkpoint from appearing as a nearly identical training and evaluation example.

A.2.2 Outcome-grounded supervision

Absolute risks and paired effects constrain complementary aspects of the predictor. Binary cross-entropy anchors the intervention and control predictions in observed violations. A paired-difference term couples the two predictions to the measured change, while a utility term estimates the cost to the legitimate task. For each paired observation, the three supervised terms are

$$\ell_{\text{out}}^{A,i} = \text{BCE}(\widehat{p}_A, Y_{A,i}) + \text{BCE}(\widehat{p}_{0|A}, Y_{0|A,i}), \tag{15}$$

$$\ell_{\text{pair}}^{A,i} = (\widehat{p}_A - \widehat{p}_{0|A} - \delta Y_{A,i})^2, \tag{16}$$

$$\ell_{U}^{A,i} = \text{Huber}(\widehat{\Delta U}_A, \delta U_{A,i}). \tag{17}$$

Their weighted average defines the supervision objective:

$$\mathcal{L}_{\text{sup}} = \mathbb{E}_{x_A \sim \mathcal{D}_{\text{rep}}} \left[\frac{1}{R} \sum_{i=1}^{R} \left(\ell_{\text{out}}^{A,i} + \lambda_{\text{pair}} \ell_{\text{pair}}^{A,i} + \lambda_U \ell_U^{A,i} \right) \right]. \tag{18}$$

The model's no-intervention decision at deployment uses $x_\emptyset = (\mathcal{G}_d, z, \emptyset, d)$. Its score $\widehat{p}_d^{\,0}$ is not the action-conditioned control score $\widehat{p}_{0|A}$. The former asks whether to continue from the current state; the latter provides a matched comparison for an action that regenerates from an earlier state. Maintaining separate inputs prevents an apparent safety improvement from being attributed to an actuator when it actually arises from changing the continuation boundary.

A.2.3 Privileged suffixes, invariance, and uncertainty

The privileged suffix encoder observes a realized continuation during training and supplies a representation target. Let $\mathbf{r}_{\text{suf},i}^A$ be that target and let $\widehat{\mathbf{r}}_{\text{suf}}^A$ be the prefix–action prediction. Stop-gradient distillation uses

$$\mathcal{L}_{\text{suf}} = \mathbb{E}_{x_A} \left[\frac{1}{R} \sum_{i} \left\| \widehat{\mathbf{r}}_{\text{suf}}^A - (\mathbf{r}_{\text{suf},i}^A) \right\|_2^2 \right]. \tag{19}$$

The training-only continuation makes outcome information available to the target, not to the deployment input. The student still receives only the observed graph, sink, action set, and rollback cut. Reusing a cached suffix representation at test time would violate this boundary even if the suffix encoder's parameters were frozen.

The conditional-variance target uses repeated intervention outcomes rather than paired differences. A separate invariance term compares equivalent serializations of the same typed partial order. For representations h_θ and admissible serializations $\pi \sim \pi'$, the consistency term encourages agreement between $h_\theta(\pi, A)$ and $h_\theta(\pi', A)$ while preserving sensitivity to changes in executable dependencies. The complete objective retains the main-paper decomposition:

$$\mathcal{L}_{\text{offline}} = \mathcal{L}_{\text{sup}} + \lambda_{\text{suf}} \mathcal{L}_{\text{suf}} + \lambda_{\text{var}} \mathcal{L}_{\text{var}} + \lambda_{\text{inv}} \mathcal{L}_{\text{poset}}. \tag{20}$$

Bootstrapped heads supply epistemic variation, and repeated continuations supervise the conditional outcome-variance head. Acquisition prioritizes unobserved pairs and triples using variation of the

same composite cost–utility objective employed by selection. It must not silently replace that objective with raw risk reduction, since such a change would favor interventions that are safe but unnecessarily disruptive. Only the trained intervention model returns to the online pillar. The next section distinguishes the variance targets and the empirical interpretation of calibration.

A.3 Variance and Calibration Protocol

Uncertainty supports a conservative decision rule only when its target is clear. RACG distinguishes variability across model heads, variability of repeated intervention outcomes, and variability of paired intervention effects. These quantities answer different questions and should not share a label merely because all can be written as variances.

A.3.1 Outcome and paired-effect variance

For repeated intervention outcomes, let $\bar{Y}_A = R^{-1}\sum_i Y_{A,i}$. The conditional outcome-variance target is

$$s^2_{Y,A} = \frac{1}{R-1}\sum_i (Y_{A,i} - \bar{Y}_A)^2, \qquad \mathcal{L}_{\text{var}} = \mathbb{E}_{x_A}\left[\text{Huber}\left(\widehat{\sigma}^2_{\text{ale}}(A), s^2_{Y,A}\right)\right]. \tag{21}$$

This variance describes stochastic variation in the binary intervention outcome. It is not the variance of the replay mean, which under independent continuations scales as $s^2_{Y,A}/R$. Treating those two quantities as interchangeable would change the uncertainty score whenever the replay budget changes, even if the underlying workflow distribution is unchanged.

For $H \geq 2$ bootstrapped heads, the deployment prediction and uncertainty decomposition are

$$\widehat{p}_A = \frac{1}{H}\sum_h p_A^{(h)}, \qquad \widehat{\sigma}^2_A = \frac{1}{H-1}\sum_h (p_A^{(h)} - \widehat{p}_A)^2 + \widehat{\sigma}^2_{\text{ale}}(A). \tag{22}$$

Matched-control risks are averaged analogously. The first variance term measures disagreement of the fitted heads; the second is the predicted variability of the outcome itself. Their sum is an operational predictive score, not automatically a standard error or a Bayesian posterior variance.

For a paired effect, the relevant sample variance instead uses $\delta Y_{A,i}$:

$$s^2_{\delta,A} = \frac{1}{R-1}\sum_i (\delta Y_{A,i} - \delta\bar{Y}_A)^2, \qquad \text{Var}(\delta Y_A) = \text{Var}(Y_A) + \text{Var}(Y_{0|A}) - 2\,\text{Cov}(Y_A, Y_{0|A}). \tag{23}$$

The covariance term explains why paired-effect uncertainty cannot be recovered by adding the two marginal variances. Under independent replay pairs, the estimated standard error of the paired mean is $s_{\delta,A}/\sqrt{R}$.

A.3.2 Held-out calibration and abstention

The online upper score is $\overline{p}_A = \min\{1, \widehat{p}_A + \kappa\widehat{\sigma}_A\}$. Calibration is performed on held-out checkpoint–action pairs using repeated intervention outcomes. If k_A of R intervention continuations violate policy, a one-sided Clopper–Pearson upper endpoint is

$$u_A^{\text{CP}} = \begin{cases} \text{Beta}^{-1}(1-\alpha; k_A+1, R-k_A), & k_A < R, \\ 1, & k_A = R. \end{cases} \tag{24}$$

The calibration rule selects κ so that the upper score covers these endpoints at the declared target frequency on held-out checkpoint–action pairs. The binomial interpretation of the endpoint assumes an appropriate repeated-trial model; dependent continuations require a corresponding uncertainty treatment. Calibration also depends on the unit of averaging and the candidate distribution. A target frequency over all candidates need not equal coverage after selecting a particular candidate.

If calibration cannot meet the target, RACG increases abstention rather than raising the admissible violation threshold ϵ. This rule does not imply zero failures: both the predictor and the calibration sample are finite, and transfer can change their relationship to future outcomes. RQ4 therefore reports achieved coverage and false feasibility rather than treating the upper score as a universal guarantee.

Acquisition on the selection objective. For acquisition, RACG uses disagreement across bootstrapped predictions of the same composite objective:

$$\widehat{\sigma}_J^2(A) = \frac{1}{H-1} \sum_h \left(J^{(h)}(A) - H^{-1} \sum_{h'} J^{(h')}(A) \right)^2 . \tag{25}$$

Here $J^{(h)}(A)$ is the head-conditioned cost–utility score used to prioritize unobserved action pairs and triples. This acquisition quantity is distinct from both intervention-outcome variance and paired-effect variance.

A.4 Five-Stage Operational Contract

The online pillar keeps model parameters fixed while executing the following five-stage contract. These stages organize the existing three online modules; they do not introduce additional actuators or an online training loop.

1. **Detection.** Input: decision graph, pending sink, and threshold. Operation: score the no-intervention continuation. Output: commit or search trigger.
2. **Tracing.** Input: selected set and decision graph. Operation: compute qualifying dependency closures. Output: candidate risk-propagation subgraph.
3. **Isolation.** Input: typed selected actions. Operation: execute their validated actuators. Output: modified rollback state, reusing branches only under disjoint footprints.
4. **Recovery.** Input: modified state. Operation: regenerate the complete affected continuation and revalidate shared versions or conflicts. Output: a suffix that passes the gate or remains denied.
5. **Update.** Input: intervention, regenerated events, verified outcomes, and failures. Operation: append events and refresh graph features. Output: an auditable next decision graph; parameter updates remain offline.

A.4.1 Affected-cone scoring and joint action semantics

The affected cone narrows prediction to observed dependencies connecting an action target to the pending sink. For an action a targeting v, its cone is $(\mathrm{Desc}(v) \cap \mathrm{Anc}(z)) \cup \{v, z\}$. A set is represented on the union of its action cones. Source, permission, version, and historical features attached to these events retain the available upstream context; restricting propagation to a cone must not replace those features with terminal labels.

Action embeddings are injected before typed propagation. At an event reached by multiple action cones, the model receives a permutation-invariant summary of the corresponding full action representations. This construction distinguishes an intervention coalition from a graph embedding followed by a late, independent list of actions. The intended benefit is not that all overlaps imply synergy, but that the representation can express both complementary and redundant effects in the dependency context where they interact.

For example, quarantining one contaminated message can be insufficient if a second agent has already copied the same information into shared memory. Restoring that memory can likewise be insufficient if the message remains available to recreate it. Their combination can have an effect that a sum of singleton predictions misses. The opposite situation also occurs: two actions can block the same effective influence, making the second action redundant. These are explanatory possibilities, not additional measured examples. RQ3 tests the resulting joint-effect prediction and selection behavior quantitatively.

A.4.2 Constrained selection, rollback, and commit

For each searched set, RACG predicts risk and utility loss, computes the calibrated upper risk score, and minimizes $C(A) + \beta \widehat{\Delta U}_A$ among sets whose upper score does not exceed ϵ. The search domain is the declared beam collection $\mathcal{B}_{K,B}(\mathcal{G}_d, z)$, not all conceivable workflow modifications. With m

Table 6: Operational pseudocode for one pending privileged sink.

Step	Operation
1	Freeze the observable prefix at cut d and retain the pending sink z behind its commit gate. Apply deterministic policy checks.
2	Score the no-intervention continuation from d. If intervention search is triggered, enumerate valid frontier actions without accessing terminal truth.
3	Expand compatible sets up to K using beam width B. Build union-cone representations and predict risk, utility loss, and uncertainty.
4	Reject predicted-infeasible sets. Minimize the declared composite cost on the remaining searched sets; resolve ties by cost, utility loss, cardinality, and stable action identifier.
5	If no searched set is feasible, deny the sink or request confirmation. Absence of a feasible searched set does not prove global infeasibility.
6	Revalidate the selected set and restore $s_{r(A)}$. Apply the set once; regenerate affected branches and revalidate any reused state.
7	Check the regenerated continuation before commit. Append interventions, regenerated events, failures, and observed outcomes to the graph history.

atomic candidates, the unconstrained count of nonempty sets of size at most three is

$$N_{\leq 3}(m) = m + \binom{m}{2} + \binom{m}{3}, \tag{26}$$

before compatibility filtering. This count explains why fixed-cardinality exact diagnostics and bounded online search are different computational regimes. It is not a measured runtime model: graph encoding, batching, caching, and regeneration contribute additional cost.

Rollback is a state operation, not deletion of an inconvenient transcript. Reused concurrent branches require disjoint read/write footprints and valid version and conflict relations. Otherwise they must be revalidated or regenerated. The controller must also distinguish an attempted external action from a committed effect, because replaying a committed side effect can duplicate a transaction. The first-paper sandbox assumes recoverable, deterministic actuators and gated external commits; general production compensation is outside the empirical claim.

After an irreversible disclosure or transfer has committed, the graph can support containment and investigation, but cannot retroactively make that event safe.

B Evaluation Metrics and Aggregation

ASR measures attack success and DTS measures defended task success on attacked workflows; BTS measures benign-task success under defense. They are not complements. A workflow can fail the task without satisfying the attack, and a benign-task population is distinct from the attacked population. In particular, $1 - \mathrm{ASR}$ cannot be substituted for DTS or BTS. The nine-task macro columns are arithmetic averages over benchmarks, not sample-weighted pooled rates.

Selection regret measures objective loss relative to a declared reference, while Top-1 measures agreement with its canonical action choice. Equivalent-set coverage can credit alternative choices only under a specified equivalence rule and tolerance. These distinctions matter when several interventions have similar effects. A finite-replay reference fixes one diagnostic comparison; it does not establish equality to the population optimum. Hypervolume additionally requires objective directions, normalization, and a reference point. The supplementary tables report the provided HV values, but this appendix does not reconstruct an unreported Pareto frontier from them.

C Evaluation Interfaces and Extended RQ1 Results

RQ1 evaluates the security–utility behavior of a complete defense, while the supplementary native comparisons examine what information each method produces before intervention. Keeping these interfaces separate avoids crediting a detector with a recovery action supplied by an adapter, or judging a provenance system solely by a task-completion score it was not designed to output.

Table 7: Native malicious-agent/event detection. Metrics are on a $[0, 1]$ scale; these are native outputs, not adapter-augmented defenses.

Method	AUROC ↑	AUPRC ↑	F1 ↑	Recall ↑
Text-only classifier	0.7683	0.7291	0.7046	0.7318
Bag-of-Events	0.8074	0.7836	0.7689	0.7542
G-Safeguard	0.8621	0.8374	0.8263	0.8491
BlindGuard	0.8846	0.8538	0.8594	0.8377
XG-Guard	0.9132	0.8857	0.8746	0.8884
GUARDIAN	0.8771	0.8602	0.8391	0.8637
MAGE (native only)	0.8918	0.8731	0.8672	0.8953
RACG	0.9086	**0.8964**	**0.8891**	**0.9127**

MAGE is evaluated through its native detection interface; no end-to-end intervention score is derived from these metrics.

C.1 Benchmark and comparison interfaces

The main evaluation covers GSM8K, SVAMP, AQuA, MATH, HumanEval, MBPP, MMLU, HotpotQA, and BrowseComp+. Task correctness and attack success are evaluated separately within the corresponding multi-agent workflows, using the protected sink and task evaluator associated with each setting.

Detection baselines produce message- or agent-level risk information. Their adapter-augmented variants connect that information to executable actions. Provenance and information-flow methods expose dependency paths or policy information, while their adapted variants use the same available action interface. Intervention-selection baselines operate directly on action choices. Across executable-defense comparisons, methods share the event representation, admissible action catalog, deterministic policy checks, and decision-budget interface so that differences reflect the corresponding detection, relational modeling, or intervention-selection mechanism.

AiTM belongs to the attack setting rather than the defense ranking. MAGE is evaluated only through its native detection interface, while AcMAS is treated as a native recovery method rather than assigned an adapter-based end-to-end score. Separating native and adapted interfaces avoids interpreting an unavailable interface as a measured failure.

C.2 Native detection and provenance results

Table 7 reports native malicious-agent/event detection. RACG is not uniformly strongest on every detection metric. XG-Guard reaches the highest AUROC, 0.9132, compared with RACG's 0.9086. RACG obtains the highest AUPRC (0.8964), F1 (0.8891), and agent/event recall (0.9127) among the listed methods. These metrics emphasize different operating characteristics: higher recall at one operating point does not imply higher AUROC, and native detection quality alone does not determine the quality of the selected intervention.

Table 8 provides the complementary native provenance and information-flow comparison. NeuroTaint obtains higher path precision than RACG (0.9148 versus 0.9037), whereas RACG obtains higher path recall (0.9314 versus 0.9236). SPA and Fides reach policy-violation recall of 0.9516 and 0.9408, respectively, both above RACG's 0.9342. These results preserve the specialized strengths of provenance and information-flow methods rather than treating intervention-effect learning as a replacement for them.

Taken together, the two native panels distinguish component-level capabilities from executable-defense outcomes. Detection metrics measure how well malicious agents or events are identified, while provenance metrics measure how accurately relevant paths or policy violations are recovered. RACG's central objective is different: it uses the resulting typed execution context to estimate the effects of executable interventions. Native detection or provenance performance therefore provides supporting diagnostic evidence rather than a substitute for the end-to-end security–utility evaluation in RQ1.

Table 8: Native provenance and information-flow results. All values are on a $[0, 1]$ scale.

Method	Path precision ↑	Path recall ↑	Policy recall ↑
Provenance Reachability	0.7326	0.8643	0.7791
NeuroTaint	**0.9148**	0.9236	0.8872
ProvenanceGuard	0.8613	0.8469	0.8641
Fides	0.7927	0.8254	0.9408
SPA	0.8263	0.8741	**0.9516**
RACG	0.9037	**0.9314**	0.9342

Table 9: Task-specific RACG outcomes and arithmetic differences from the nine-benchmark main evaluation. Positive differences denote lower ASR or higher DTS for RACG.

Benchmark	ASR	DTS	$\Delta\text{ASR}_{\text{ND}}$	$\Delta\text{DTS}_{\text{ND}}$	$\Delta\text{ASR}_{\text{LF}}$	$\Delta\text{DTS}_{\text{LF}}$
GSM8K	9.63	83.44	15.00	-1.28	1.23	0.36
SVAMP	9.42	85.69	13.49	-1.47	0.79	0.32
AQuA	12.91	72.66	14.57	-1.19	0.57	0.49
MATH	14.27	66.48	16.49	-0.76	1.47	0.62
HumanEval	15.39	71.52	18.19	0.06	1.43	0.56
MBPP	14.46	75.17	16.81	-0.66	1.23	0.64
MMLU	11.84	77.43	15.10	-1.19	0.87	0.49
HotpotQA	15.62	69.84	26.11	8.36	2.85	2.46
BrowseComp+	20.76	30.61	31.60	11.87	5.16	3.13

ND: No defense; LF: Full-Graph Late Fusion. ASR/DTS are percentages and differences are percentage points. Negative DTS differences are retained.

C.3 Per-benchmark main-table decomposition

Table 9 complements the aggregate RQ1 results with task-level differences. Against Full-Graph Late Fusion, RACG has lower ASR and higher DTS on all nine benchmarks. The largest differences occur on BrowseComp+, where RACG reduces ASR by 5.16 percentage points and improves DTS by 3.13 points. HotpotQA follows with a 2.85-point ASR reduction and a 2.46-point DTS improvement.

These task-level differences are descriptive rather than evidence that workflow length alone causes the observed gains. Task families differ in reasoning structure, retrieval requirements, tools, and execution patterns, so the comparison does not isolate any single task characteristic.

Against No defense, the utility picture is mixed. RACG's DTS is lower on GSM8K, SVAMP, AQuA, MATH, MBPP, and MMLU, while it is higher on HumanEval, HotpotQA, and BrowseComp+. The BrowseComp+ improvement is 11.87 points, although its absolute DTS remains 30.61%. At the macro level, ASR decreases from 32.41% to 13.81%, DTS increases from 68.79% to 70.32%, and BTS decreases from 79.86% to 79.08%. The results therefore characterize a security–utility trade-off rather than cost-free safety.

D RQ2: Relation Discrimination and Downstream Outcomes

RQ2 examines whether typed collaboration relations improve risk discrimination and downstream defense beyond message content and untyped connectivity. We organize the analysis at three levels. First, single-relation diagnostics evaluate five relation families: source/permission, delegation, memory version, tool binding, and sink relations. Second, compositional diagnostics test whether several relation types can be combined within the same decision. Third, associated natural relation-compositional workflows evaluate whether stronger relational discrimination accompanies improved defended outcomes. These evaluations provide complementary evidence and are not treated as numerical substitutes for one another.

Table 10: Single-relation pair consistency by relation family (%; higher is better).

Method	Source/perm.	Delegation	Memory ver.	Tool binding	Sink
Text-only classifier	1.84	2.37	1.29	2.68	1.53
Bag-of-Events	42.76	31.48	45.93	34.17	38.62
Homogeneous Graph	64.28	60.73	62.41	58.96	66.12
Relation-Shuffled	61.37	57.82	63.48	55.91	60.26
RACG	**84.93**	**81.76**	**86.42**	**83.18**	**85.37**

Table 11: Compositional relation discrimination and associated natural-workflow outcomes. Composition PC, ASR, BTS, and DTS are percentages; cost is normalized intervention cost.

Method	Comp. PC↑	ASR↓	BTS↑	DTS↑	Cost↓
Text-only classifier	1.16	34.72	80.86	60.93	0.28
Bag-of-Events	27.83	29.46	80.17	64.28	0.34
Homogeneous Graph	53.72	23.81	79.68	68.57	0.43
Relation-Shuffled	50.18	26.94	79.51	65.42	0.46
RACG	**76.41**	**18.34**	79.74	**72.41**	0.40

D.1 Single-relation discrimination

Table 10 reports pair consistency for each relation family. RACG achieves the highest score in every category, ranging from 81.76% for delegation to 86.42% for memory version. Compared with Homogeneous Graph, RACG improves source/permission, delegation, memory-version, tool-binding, and sink consistency by 20.65, 21.03, 24.01, 24.22, and 19.25 percentage points, respectively. The advantage therefore extends across provenance, state, execution, and sink-related relations rather than being concentrated in a single relation family.

The baseline progression separates access to message content, event context, graph connectivity, and typed relations. Bag-of-Events outperforms Text-only across all five relation families, indicating that aggregating event-level context improves discrimination over message-only classification in this diagnostic. Homogeneous Graph further improves on Bag-of-Events across all five families, showing that connectivity is informative even without explicit relation typing. RACG consistently exceeds both Homogeneous Graph and Relation-Shuffled, indicating an additional benefit from preserving which types of relations connect the observed events.

D.2 Compositional relation discrimination

Single-relation performance does not establish that a model can combine several relation types within the same collaboration instance. We therefore separately evaluate composition pair consistency. As shown in Table 11, RACG reaches 76.41%, compared with 53.72% for Homogeneous Graph and 50.18% for Relation-Shuffled, corresponding to improvements of 22.69 and 26.23 percentage points, respectively. Bag-of-Events reaches 27.83%, remaining substantially below the graph-based variants.

Composition is more challenging than isolated relation discrimination: RACG's 76.41% composition consistency is below each of its five single-relation scores. Nevertheless, its advantage over the relation-ablated variants persists when multiple relation types must be considered together. This pattern is consistent with typed structure providing useful information when decisions depend jointly on provenance, permissions, shared state, tool bindings, and downstream execution dependencies.

D.3 Connection to executable defense

The relation diagnostics measure discrimination, whereas the natural-workflow evaluation measures the behavior of the complete defense. On these workflows, RACG achieves 18.34% ASR and 72.41% DTS. Homogeneous Graph reaches 23.81% ASR and 68.57% DTS, so RACG reduces attack success by 5.47 percentage points while improving defended task success by 3.84 points. BTS

Table 12: Joint intervention-effect estimation: risk prediction errors and diagnostic scores. MAE and ECE retain their unit scales; synergy accuracy is a percentage.

Estimator	Single MAE	Pair MAE	Triple MAE	Syn. acc.	ECE
Singleton Additive	0.094	0.163	0.218	57.84	0.091
HeteroGNN + SetEnc	0.071	0.119	0.158	68.37	0.067
Full-Graph Late Fusion	0.064	0.104	0.139	72.91	0.058
RACG w/o Cone Intersection	0.061	0.112	0.151	66.48	0.055
RACG w/o Privileged Suffix	0.069	0.097	0.128	74.26	0.079
RACG	0.062	0.086	0.113	80.73	0.046

remains nearly unchanged (79.74% versus 79.68%), and normalized intervention cost decreases from 0.43 to 0.40.

Removing relation identity while retaining graph structure produces a larger downstream gap. Relation-Shuffled reaches 26.94% ASR and 65.42% DTS; RACG therefore improves the two metrics by 8.60 and 6.99 percentage points, respectively, while reducing normalized cost from 0.46 to 0.40. Together with the composition diagnostic, this comparison separates the benefit of typed relations from connectivity alone.

Content-based variants occupy a different security–utility operating point. Text-only and Bag-of-Events use less intervention cost and retain slightly higher BTS, but provide substantially weaker protection under attack. Relative to Text-only, RACG reduces ASR by 16.38 percentage points and increases DTS by 11.48 points, while BTS decreases by 1.12 points and normalized cost increases by 0.12. Thus, RACG's advantage in this comparison is stronger security and defended task completion rather than uniformly improving every metric.

Across the three levels of evaluation, the results show a consistent ordering: RACG performs better on individual typed-relation diagnostics, retains its advantage when multiple relations are composed, and reaches lower attack success with higher defended task success on the associated workflows. These aggregate comparisons establish an association between relational discrimination and downstream defense, rather than treating diagnostic accuracy alone as evidence of intervention efficacy. RQ3 separately evaluates whether RACG can estimate interactions among multiple executable interventions and select effective intervention sets.

E RQ3: Joint Effects and Executable Set Selection

RQ3 separates prediction quality from search quality. An estimator can assign inaccurate scores even when every candidate is enumerated; conversely, a well-calibrated estimator can be paired with a search that misses useful coalitions. The supplementary results expose both sources of error rather than attributing all gains to a single graph encoder.

E.1 Estimation across coalition sizes

Singleton Additive has risk MAE of 0.094, 0.163, and 0.218 for singletons, pairs, and triples (Table 12). RACG's corresponding errors are 0.062, 0.086, and 0.113. Error increases with coalition size for every listed estimator, but the increase is smaller for RACG than for the additive estimator. This is the expected diagnostic distinction when singleton behavior does not fully determine coalition behavior.

Removing cone intersections changes singleton risk MAE from 0.062 to 0.061, a slight improvement, while worsening pair and triple MAE to 0.112 and 0.151. Synergy accuracy falls from 80.73% to 66.48%. The ablation therefore does not support a blanket claim that cone intersections improve every prediction. Its more specific pattern is consistent with their role in representing joint effects. Full-Graph Late Fusion also has higher pair/triple risk MAE, 0.104/0.139, despite retaining a graph representation.

Utility errors show a similar coalition-size pattern. RACG has MAE of 0.058, 0.081, and 0.104, compared with 0.061, 0.094, and 0.126 for Full-Graph Late Fusion. Removing privileged-suffix

Table 13: Utility-loss estimation error by intervention-set size. Lower MAE is better.

Estimator	Single MAE	Pair MAE	Triple MAE
Singleton Additive	0.082	0.141	0.196
HeteroGNN + SetEnc	0.066	0.107	0.143
Full-Graph Late Fusion	0.061	0.094	0.126
RACG w/o Cone Intersection	0.059	0.101	0.134
RACG w/o Privileged Suffix	0.064	0.089	0.117
RACG	0.058	0.081	0.104

Table 14: Executable intervention-set selection. Eq. Cov., Top-1, Eval., ASR and DTS retain the workbook's percentage scales; regret retains its objective scale.

Selector	Eq. Cov.	Top-1	Regret	Eval.	ASR	DTS
Random Feasible	17.63	8.94	0.284	24.71	22.46	69.38
Greedy Singleton	41.82	27.36	0.176	19.43	15.87	73.64
Beam + Additive Scores	54.27	36.91	0.131	31.68	13.94	74.71
Exact + RACG Scores	82.46	66.18	0.054	100.00	9.86	76.48
RACG eMEIS	77.38	59.73	0.071	27.54	10.47	75.92
Finite-Replay Ref.	100.00	100.00	0.000	100.00	8.91	77.13

Finite-Replay Ref. is a diagnostic reference. Exact + RACG Scores enumerates model-scored candidates; it is not an outcome oracle. Evaluation effort is normalized to 100 for exhaustive enumeration.

supervision yields 0.064, 0.089, and 0.117 utility MAE; triple risk MAE rises to 0.128 and ECE to 0.079 from 0.046. These aggregate differences are consistent with a contribution from offline continuation information. They do not establish that the student reconstructs the actual future or that suffix information is available at inference.

E.2 Search, reference agreement, and downstream outcomes

Table 14 distinguishes a finite-replay reference, exhaustive search using learned scores, and deployable eMEIS search. Finite-Replay Ref. is assigned 100% equivalence coverage and Top-1 agreement and zero regret because it defines the diagnostic reference. Those entries are not independent evidence of perfect safety: its ASR is 8.91%, and its outcomes remain finite-replay estimates.

Exact + RACG Scores has regret 0.054, compared with 0.071 for RACG eMEIS, and better equivalence coverage, 82.46% versus 77.38%. It also has slightly lower ASR and higher DTS, 9.86%/76.48% versus 10.47%/75.92%. RACG eMEIS uses 27.54 on the normalized evaluation scale, while exact scoring uses 100. The 72.46-point reduction in normalized evaluation effort does not imply an equal wall-clock speedup, since the table does not decompose encoding and recovery latency.

Greedy Singleton uses less evaluation effort than RACG eMEIS, 19.43 versus 27.54, but has higher regret, 0.176 versus 0.071. Beam + Additive Scores has regret 0.131 despite using more evaluation effort, 31.68. These comparisons distinguish a scoring limitation from an enumeration limitation. Exhaustive learned scoring still differs from the finite-replay reference; bounded learned scoring adds a further search approximation. Neither should be renamed the population-optimal intervention.

E.3 What the diagnostic does not establish

The evaluation is restricted to the declared frontier, compatible action catalog, coalition sizes, and diagnostic candidates. Equivalent-set coverage also depends on how action equivalence is defined. Without its tolerance and the finite-replay feasibility rule, the table does not establish exact recovery of a unique minimal intervention. The result supports a relative benefit of joint estimation and bounded selection in the supplied diagnostic, while leaving unrestricted combinatorial optimization and larger coalitions outside the measured scope.

Table 15: RACG degradation under operational stress. ASR increase and DTS drop are percentage points; action and latency increases are relative percentages.

Stressor	Level	ASR inc.	DTS drop	Act. inc.	Lat. inc.
Irrelevant candidates	+25% candidates	0.74	0.48	2.61	8.37
Irrelevant candidates	+50% candidates	1.46	1.03	5.84	17.92
Irrelevant candidates	+100% candidates	3.18	2.41	11.73	34.86
Misleading actions	+100% candidates	5.72	4.63	16.28	38.41
Delay + reorder	U[1,3] steps; 25%	2.43	1.87	3.74	21.68
Retry + duplicate	p=0.25	1.67	1.31	4.92	18.46
Stale memory version	lag = 1 committed ver.	3.94	3.26	7.18	14.29

Delay/reorder uses a 1–3-step delay and the supplied 25% setting; retry/duplicate uses $p = 0.25$. Stress levels differ, so rows do not constitute equal-intensity attacks.

Table 16: Matched stress-point comparison. Each degradation is relative to the corresponding method's unstressed reference (percentage points).

Stressor	RACG ASR inc.	LF ASR inc.	RACG DTS drop	LF DTS drop
Irrelevant (+100%)	3.18	5.43	2.41	4.16
Delay + reorder	2.43	4.71	1.87	3.58

LF: Full-Graph Late Fusion. Only these two baseline stress points are available.

F RQ4: Operational Stress and Reliability

RQ4 examines failures that a clean, synchronous workflow comparison can hide. Candidate-pool noise changes the selection problem; delayed or duplicated events change the observed execution; stale memory changes state validity. Reliability ablations then separate risk scoring, calibration, and actuator validity from the representation used to predict effects.

F.1 Stress severity and overhead

Table 15 reports the full supplied stress sweep. Adding irrelevant candidates at 25%, 50%, and 100% raises ASR by 0.74, 1.46, and 3.18 percentage points, respectively. DTS drops by 0.48, 1.03, and 2.41 points. Relative latency increases by 8.37%, 17.92%, and 34.86%. All three quantities increase over the reported levels, but three points do not establish a general scaling law or linear complexity.

At the same 100% candidate-count increase, misleading actions cause more degradation than irrelevant candidates: ASR increases by 5.72 rather than 3.18 points, and DTS drops by 4.63 rather than 2.41 points. The corresponding action and latency increases are also larger. This comparison is consistent with semantically plausible bad choices being harder to reject than irrelevant choices. It does not show that misleading actions are intrinsically the most damaging stressor under every budget, since the other stress types use different intensity parameters.

Delay/reorder produces 2.43 points of ASR increase and 1.87 points of DTS drop. Retry/duplicate produces 1.67 and 1.31 points, while a one-committed-version memory lag produces 3.94 and 3.26 points. These results show nonzero sensitivity to asynchronous and stale-state conditions. They do not establish tolerance to unbounded delay, arbitrary event loss, or arbitrary historical memory corruption.

At the two available matched stress points, RACG has less degradation than Full-Graph Late Fusion. For irrelevant candidates, the ASR/DTS degradation differences are 2.25/1.75 points; for delay/reorder, they are 2.28/1.71 points. Table 16 compares changes from each method's own unstressed reference, not absolute attacked outcomes. No Late Fusion values are supplied for misleading actions, retries, or stale memory, so its behavior on those stressors is left unreported.

Table 17: Full reliability ablations. Coverage, false feasibility, invalidity, ASR and DTS are percentages; ECE is on a $[0, 1]$ scale.

Variant	ECE	UCB cov.	False feas.	Invalid	ASR	DTS
RACG	0.05	94.21	2.18	0.42	13.81	70.32
w/o UCB	0.05	–	6.73	0.47	16.94	69.01
w/o calibration	0.11	86.37	7.41	0.51	17.28	68.74
w/o matched rollback	0.07	90.84	5.92	2.84	18.61	67.85
w/o validity filtering	0.05	93.66	3.26	9.63	15.47	68.92
w/o frontier/cone limit	0.06	92.48	4.11	1.37	16.12	68.37

A missing UCB coverage entry for w/o UCB is not zero coverage. Reported coverage is empirical, not a distribution-free guarantee.

F.2 Reliability ablations

Removing the upper-risk rule raises false feasibility from 2.18% to 6.73% while leaving the reported ECE at 0.05. Calibration and the decision rule therefore measure different aspects of reliability: an unchanged average calibration score does not imply unchanged thresholded decisions. Removing calibration raises ECE to 0.11, lowers empirical UCB coverage from 94.21% to 86.37%, and raises false feasibility to 7.41%.

The validity filter addresses a different failure mode. Removing it raises invalidity from 0.42% to 9.63%, whereas ECE remains 0.05. A risk predictor cannot substitute for checking whether an actuator is executable. Removing matched rollback has the largest ASR increase among these listed ablations: ASR rises from 13.81% to 18.61%, and DTS falls from 70.32% to 67.85%. This pattern is consistent with the importance of preserving the action-specific state boundary, although the aggregate table does not decompose replay-label error from online recovery error.

Removing the frontier/cone limit raises ASR to 16.12%, lowers DTS to 68.37%, and raises invalidity to 1.37%. The reported results favor the constrained implementation, but they do not isolate whether the difference is caused by candidate quality, representation dilution, search budget allocation, or a combination. Such a mechanism claim would require a more targeted comparison than this row alone.

F.3 Presentation and interpretation

ASR increase and DTS drop are percentage-point changes. Action and latency increases are relative percentages. They should not be averaged into a common degradation score without a declared normalization, nor displayed on one unlabeled common scale. Similarly, a heatmap of reliability metrics may use within-column normalization for color, but its annotations must retain the original values. The tables provide the exact quantities behind such visual summaries and preserve the missing coverage entry for the no-UCB ablation.

G RQ5: Transfer Across Unseen Settings

RQ5 evaluates whether intervention knowledge learned in the source setting remains useful under previously unseen attack structures, workflow templates, delegation topologies, privileged sinks, and model backbones. Fixed Policy provides the non-transferred reference, Frozen RACG applies the transferred intervention policy without target-side parameter updates, Calibration-only adjusts decision calibration on the target setting, and Target-trained provides a target-adapted reference. We evaluate transfer through attack success rate (ASR), defended task success (DTS), and expected calibration error (ECE), separating transferred defense performance from target-side adaptation.

G.1 Full target-shift results

Table 18 reports the four systems across five target shifts. Frozen RACG improves over Fixed Policy in every setting: ASR decreases by 5.19–6.91 percentage points and DTS increases by 4.08–5.65

points. The improvement therefore appears before target-side calibration across all five evaluated shifts.

Table 18: Full transfer results across five target shifts. ASR and DTS are percentages; lower ECE is better.

Target shift	System	ASR ↓	DTS ↑	ECE ↓
Two-channel attack	Fixed Policy	24.83	64.71	0.13
	Frozen RACG	17.92	70.36	0.09
	Calibration-only	16.74	71.18	0.05
	Target-trained	14.29	73.46	0.04
Workflow template	Fixed Policy	26.17	62.83	0.14
	Frozen RACG	19.84	67.91	0.11
	Calibration-only	18.46	69.27	0.06
	Target-trained	15.63	72.14	0.05
Delegation topology	Fixed Policy	25.46	63.58	0.12
	Frozen RACG	18.71	69.16	0.10
	Calibration-only	17.53	70.04	0.06
	Target-trained	14.87	72.76	0.05
Privileged sink	Fixed Policy	23.91	65.26	0.13
	Frozen RACG	17.46	70.82	0.09
	Calibration-only	16.38	71.43	0.05
	Target-trained	13.72	74.08	0.04
Cross-backbone	Fixed Policy	28.36	60.74	0.17
	Frozen RACG	23.17	64.82	0.14
	Calibration-only	20.84	66.73	0.07
	Target-trained	16.21	70.86	0.05

Calibration-only further improves both security and task utility in every target setting. Relative to Frozen RACG, its ASR reductions range from 1.08 to 2.33 percentage points, while its DTS improvements range from 0.61 to 1.91 points. ECE also decreases in every shift, indicating that target-side calibration improves the correspondence between predicted and observed intervention risk.

The largest incremental calibration improvement occurs in the cross-backbone setting: ASR decreases from 23.17% to 20.84%, DTS increases from 64.82% to 66.73%, and ECE decreases from 0.14 to 0.07. Target training remains stronger at 16.21% ASR, 70.86% DTS, and 0.05 ECE. Thus, calibration recovers additional performance after transfer without eliminating the remaining target-adaptation gap.

Table 19: Transfer improvements and remaining target-training gaps derived from Table 18 (percentage points). All columns are oriented so that positive values denote improvement.

Target shift	Frozen ΔASR	Frozen ΔDTS	Cal. ΔASR	Cal. ΔDTS	Gap ASR	Gap DTS
Two-channel attack	6.91	5.65	1.18	0.82	2.45	2.28
Workflow template	6.33	5.08	1.38	1.36	2.83	2.87
Delegation topology	6.75	5.58	1.18	0.88	2.66	2.72
Privileged sink	6.45	5.56	1.08	0.61	2.66	2.65
Cross-backbone	5.19	4.08	2.33	1.91	4.63	4.13

Frozen gains use Fixed Policy as the reference; calibration gains use Frozen RACG as the reference; gaps denote the remaining advantage of Target-trained over Calibration-only.

G.2 Transfer retention and remaining adaptation gap

Table 19 separates two effects that are obscured by absolute target-shift scores. The Frozen columns quantify the improvement obtained by transferring RACG without target-side parameter adaptation, whereas the calibration columns quantify the additional improvement obtained from target-side cal-

ibration. The final two columns measure the remaining gap between Calibration-only and Target-trained.

Across the four within-backbone shifts, Frozen RACG reduces ASR by 6.33–6.91 percentage points and increases DTS by 5.08–5.65 points relative to Fixed Policy. Calibration-only provides a further 1.08–1.38-point ASR reduction and a 0.61–1.36-point DTS improvement. The remaining Target-trained advantage is 2.45–2.83 ASR points and 2.28–2.87 DTS points. These results indicate that a substantial part of the source-setting improvement persists across unseen structural variants, while target adaptation continues to provide additional gains.

Cross-backbone transfer exhibits a different quantitative pattern. Frozen RACG improves over Fixed Policy by 5.19 ASR points and 4.08 DTS points, smaller than the corresponding improvements in the four within-backbone settings. Calibration then provides its largest incremental gain in this setting, reducing ASR by another 2.33 points and increasing DTS by 1.91 points. Nevertheless, the remaining Target-trained gap is also largest here, at 4.63 ASR points and 4.13 DTS points.

These comparisons show that the transferred intervention policy remains useful across all five evaluated shifts, but they do not imply complete backbone invariance. In particular, the persistent gap to Target-trained performance indicates that source-trained intervention knowledge and target-side calibration do not fully substitute for adaptation to the target backbone.