Dynamic Stance Routing: Separating Visibility from Authority in LLM Multi-Agent Systems

Anonymous ACL submission

Abstract

LLM-based multi-agent systems solve tasks by exchanging intermediate messages among agents. When a misleading claim or evidence fragment enters these messages, later agents may reuse it as a premise, relay it as support, or cite it as evidence before the final answer is produced. Existing defenses often focus on identifying, removing, or rewriting risky content, but a risk signal alone does not determine how the receiving agent should use the received information. We propose DSR, a receiver-side protocol that separates visibility from authority. A risky piece of information may remain visible for verification, qualification, refutation, or audit, while being denied authority as a factual premise, relay support, or final evidence. Given a shared risk characterization, DSR assigns use permissions over context admission, local adoption, relay authority, and final evidence eligibility. We evaluate DSR on controlled misinformation-injection tasks across multiple LLM backbones, using final-answer metrics and trace-level propagation diagnostics. Experiments show that DSR reduces final misinformation influence and authority-transfer adoption while preserving task success quality. These results suggest that robust LLM multi-agent collaboration requires controlling not only whether risky information remains visible, but also what authority it is allowed to carry.

1 Introduction

LLM-based multi-agent systems (LLM-MAS) often solve complex tasks by decomposing work across specialized agents that communicate during task execution (Wu et al., 2024; Li et al., 2023; Qian et al., 2024; Hong et al., 2024). Unlike a single model call, these systems produce answers through intermediate messages: later agents may read earlier messages, tool outputs, and partial analyses as part of their own inputs. This creates a path through which misleading or unsupported information can affect later reasoning before the final answer is produced.

Prior work has shown that errors and adversarial information can propagate through LLM-based agent interaction and tool-mediated contexts (Li et al., 2025; Xie et al., 2026; Wang et al., 2025; Zhou et al., 2025; Greshake et al., 2023; Zhan et al., 2024). The relevant risk is often smaller than an entire message. A message may contain useful task context while also including a misleading claim, an unsupported evidence snippet, or a poisoned tool-output fragment. We call such claim- or evidence-level pieces *communication units*. A risky unit is a communication unit that should not be treated as factual support without further verification. The key concern is that later agents may treat such a unit as a premise, paraphrase it in subsequent messages, or use it to support final synthesis, consistent with error-cascade observations in LLM-MAS communication (Xie et al., 2026).

Existing work has made important progress in detecting, correcting, tracing, and reducing such risks. Goal-aware misinformation work studies how misleading claims affect final answers and how such claims may be identified or corrected (Li et al., 2025). Trace- and provenance-oriented methods analyze how information moves through multi-agent communication and how errors propagate across agents (Zhou et al., 2025; Xie et al., 2026). Reliability-oriented methods study malicious or unreliable agents and their effect on collaboration (Chen et al., 2025; Zheng et al., 2026). Topology-level approaches analyze or modify communication structure to reduce unsafe information flow (Wang et al., 2025; Shen et al., 2025). These studies address where risky information appears, how it propagates, which sources or paths may be unreliable, and whether content should be corrected, filtered, or structurally isolated.

We focus on a narrower question in LLM-MAS

safety evaluation. Suppose a received unit has already been given a risk signal, such as whether it is acceptable, uncertain, risky, or rejectable. How should the agent that receives this unit use it? This question matters because message-level handling can be too coarse. Blocking a whole message may remove useful context together with the risky fragment, while simply warning about the fragment may still leave later agents free to use it as support. We therefore study how to control whether a characterized unit may enter the working context, be adopted as a premise, be relayed as support, or be used as final evidence.

We propose DSR, a receiver-local protocol that separates visibility from authority. A risky or uncertain unit may remain visible for verification, qualification, refutation, or audit, while being denied authority as a factual premise, relay support, or final evidence. DSR implements this distinction through explicit rules for four safety-relevant decisions studied in this paper: context admission, local adoption, relay authority, and final evidence eligibility. In this sense, DSR complements risk detection, correction, tracing, and topology-level interventions: it specifies how an agent should use a characterized unit after a risk signal is available.

We empirically evaluate DSR on controlled misinformation-injection workflows across four LLM backbones. All methods that use a risk signal receive the same characterization for the same injected unit, so the comparison focuses on how each method acts on that signal. We report final-answer misinformation toxicity (MT), task success score (TSS), and a trace-level adoption metric that counts when a risky unit is used as factual support. Across backbones, DSR reduces Avg. MT by about 40% or more relative to pass-through while maintaining competitive TSS. Under Prompt Injection, the most direct test of intermediate-message risk, DSR achieves the lowest MT on all four backbones. A 30-instance ablation further shows that DSR lowers Avg. MT from 5.46 to 2.83, while tag-only remains high at 5.18.

Our contributions are:

- We study how LLM-MAS should handle risky communication units after a risk signal is available, framing the problem as separating visibility from authority.
- We propose DSR, a receiver-local protocol that controls four safety-relevant decisions for a characterized unit: context admission, local adoption, relay authority, and final evidence eligibility.
- We empirically validate DSR across four LLM backbones, showing reduced final misinformation influence and lower trace-level adoption while maintaining competitive task success quality.

2 Dynamic Stance Routing

Figure 1 summarizes DSR. The protocol does not replace the multi-agent workflow and does not introduce a new risk detector. Instead, after a risk signal is available for a received unit, DSR controls how that unit may be used by the receiver. The central distinction is between *visibility* and *authority*: a unit can remain visible for inspection or refutation without being allowed to function as a factual premise, relay support, or final evidence.

2.1 Problem Setup: Units in Multi-Agent Communication

We consider an LLM-based multi-agent system as a directed communication graph:

$$\begin{aligned} G &= (V, E), \\ V &= \{a_1, \ldots, a_n\}. \end{aligned} \tag{1}$$

Each node is an agent, and an edge $(a_i, a_j) \in E$ indicates that agent a_i may send information to agent a_j. At communication round t, agent a_i may send a message to agent a_j:

$$m_{ij}^t : a_i \to a_j. \tag{2}$$

A message may contain several types of information at once, such as useful task context, intermediate reasoning, tool outputs, hypotheses, unsupported claims, or misleading evidence snippets. For this reason, DSR does not treat the whole message as a single indivisible object. Instead, we represent each message as an ordered list of communication units:

$$m_{ij}^t = \big(u_{ij,1}^t, u_{ij,2}^t, \ldots, u_{ij,K_{ij}^t}^t\big). \tag{3}$$

A unit is the claim-, evidence-, tool-output-, or partial-conclusion-level span to which DSR assigns allowed uses. This unit-level view is important because a message can contain both useful context and a risky fragment. A message-level block may remove both, while a unit-level protocol can restrict the risky fragment without necessarily discarding the whole message.

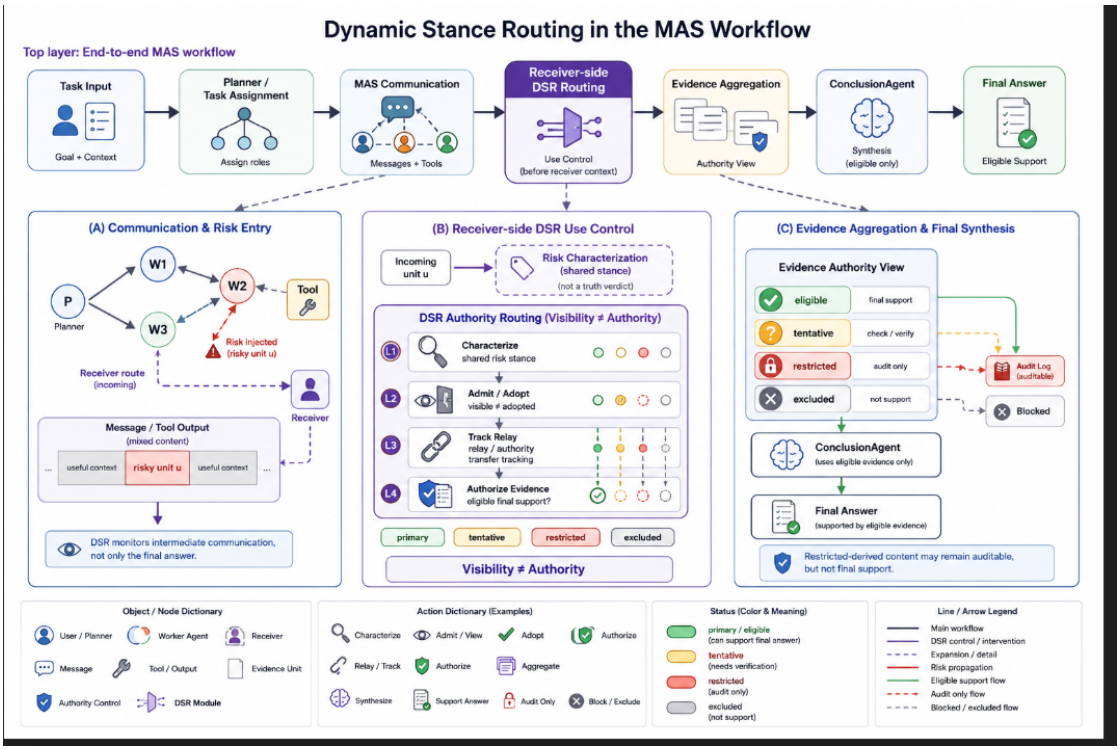

Figure 1: Overview of DSR in an LLM-MAS workflow. Intermediate messages or tool outputs may contain risky communication units. Once a received unit has a risk signal, DSR lets the receiver decide how that unit may be used: whether it may enter context, be adopted as a premise, be relayed as support, or be used as final evidence. A unit may remain visible for verification, qualification, refutation, or audit without being authorized as factual support.

Let c_i^t denote the working context of receiver a_i at round t. Given a received unit u, DSR decides how the receiver may represent and use u in its future reasoning, messages, and final evidence construction.

2.2 Risk Signal as Input, Not Decision

DSR starts after a risk signal is available for a received unit. For a unit u, a characterization function returns a stance label:

$$z(u) \in \mathcal{Z}, \quad \mathcal{Z} = \{\textsc{Accept}, \textsc{Uncertain-Usable}, \textsc{Uncertain-Risky}, \textsc{Reject}\}. \tag{4}$$

The characterization may come from a detector, an LLM judge, a rule-based module, or a controlled evaluation signal. This signal is not a truth verdict and is not itself a downstream-use decision. For example, a unit marked as uncertain may still be useful as a hypothesis to verify, while a unit marked as risky may still need to remain visible so that another agent can refute it or audit the reasoning trace. Thus, the risk signal only answers the question "how risky does this unit appear?" It does not answer the separate question "what is this unit allowed to do in later reasoning?"

In our experiments, all methods that use a risk signal consume the same cached characterization for the same unit. This design isolates the effect of downstream handling policy: differences between policies come from how they act on the same signal, not from different risk-characterization quality.

2.3 Allowed Uses: Visibility Is Not Authority

DSR maps each stance label to an authority status. We use four statuses: PRIMARY, TENTATIVE, RESTRICTED, and EXCLUDED. They describe how much downstream authority the receiver may grant to the unit.

The authority status determines an allowed-use vector:

$$p(u) = \big(p_{\text{ctx}}(u), p_{\text{adopt}}(u), p_{\text{relay}}(u), p_{\text{evid}}(u)\big). \tag{5}$$

The four entries indicate whether u may enter the receiver's working context, be used as a factual premise, be relayed as support, or be used as final evidence.

These allowed uses are ordered by downstream authority. Using a unit as final evidence or relaying it as factual support requires adoption authority,

Status	Meaning
PRIMARY	The unit may be treated as ordinary factual support.
TENTATIVE	The unit may be used only with qualification, verification, or independent support.
RESTRICTED	The unit may remain visible for scrutiny, refutation, or audit, but not as factual support.
EXCLUDED	The unit is not admitted into the receiver's working context.

Table 1: Authority statuses used by DSR. The full mapping from stance labels to rendering actions is provided in Appendix D.

and adoption requires context admission:

$$\begin{aligned} p_{\text{evid}}(u) &\le p_{\text{adopt}}(u) \le p_{\text{ctx}}(u), \\ p_{\text{relay}}(u) &\le p_{\text{adopt}}(u) \le p_{\text{ctx}}(u). \end{aligned} \tag{6}$$

Here p_{relay} refers to relay as factual support, not surface mention or critical discussion. For instance, a receiver may repeat a restricted unit in the form "this claim should be checked" or "the previous message asserts X, but it is not verified." Such a mention keeps the unit visible but does not grant relay authority. By contrast, restating the same unit as the receiver's own factual claim would require relay authority.

This design gives DSR its main behavior. A risky or uncertain unit does not have to disappear from the trace. It can remain available for verification, qualification, refutation, or audit, while being denied authority as a premise, relay support, or final evidence.

2.4 Receiver-side Protocol

DSR is applied by the agent that receives a unit. It does not require global agent quarantine or complete reconstruction of the communication graph. For each incoming message, the receiver applies four steps.

L1: Risk signal. For each received unit u, the receiver obtains the shared stance label $z(u)$. This step surfaces a risk signal but does not decide how the unit may be used. The same signal can lead to different downstream actions depending on whether the unit should be admitted, adopted, relayed, or excluded from final evidence.

L2: Context admission and local adoption. Without use control, a receiver may update its working context by appending the whole received message. DSR replaces this message-level update with permission-aware rendering over units:

$$c_i^{t+1} = c_i^t \oplus \bigoplus_{u \in \mathcal{U}(m_{ji}^t)} \text{Render}\big(u, p(u)\big). \tag{7}$$

Here $\mathcal{U}(m_{ji}^t)$ denotes the units in the received message. The rendering function determines how each unit appears to the receiver.

If $p_{\text{ctx}}(u) = 0$, the unit is not admitted into the working context. If $p_{\text{ctx}}(u) = 1$ and $p_{\text{adopt}}(u) = 0$, the unit may appear only as a guarded, refutable, or audit-only mention. In this case, the receiver can inspect or challenge the unit but should not use it as a factual premise. If $p_{\text{adopt}}(u) = 1$, the unit may be used as a working premise, possibly with qualification when its authority status is tentative. This operation separates context visibility from local adoption.

L3: Relay authority tracking. A receiver may later produce a downstream message that quotes, paraphrases, or depends on previously received units. DSR records whether such downstream content derives from units whose relay authority is restricted. The goal is not to reconstruct complete semantic provenance for every generated token. Instead, DSR tracks explicit references, quotations, and trace-matched reuse of prior units. When a downstream message is derived from a restricted unit, it should not be treated as an authoritative relay of that unit.

This distinction allows restricted units to remain discussable. A receiver can say that a prior message made a certain claim and then verify, qualify, or reject it. What DSR prevents is the silent upgrade from "this risky claim was mentioned" to "the receiver now endorses this claim as support."

L4: Final evidence eligibility. Final synthesis should not treat all trace-visible units as eligible evidence. Before producing the final answer, DSR separates units that merely appear in the trace from units that may support the answer. Let $\mathcal{T}$ denote the trace of units observed during collaboration. The final evidence set is:

$$\mathcal{E}_{\text{final}} = \{u \in \mathcal{T} : p_{\text{evid}}(u) = 1 \wedge \text{support}(u, y_{\text{final}}) = 1\}. \tag{8}$$

Here $\text{support}(u, y_{\text{final}}) = 1$ indicates that u is explicitly selected, cited, or semantically used as support for the final answer y_{final}. A unit that remains visible in the trace can therefore still be denied authority as final evidence.

This procedure is low-invasion. It changes how received units are rendered, tracked, relayed, and used as evidence, but it does not require redesigning the entire multi-agent workflow.

2.5 Trace Semantics

DSR records unit-level decisions in an audit trace. Each trace entry stores the unit, its stance label, its authority status, its allowed uses, and its downstream evidence status. The trace is used both for final evidence construction and for evaluating whether a risky unit was adopted during collaboration.

The trace distinguishes two cases that can look similar at the surface level. In a *controlled mention*, a receiver mentions a risky unit to verify, qualify, or refute it. The unit is visible, but it is not adopted as a factual premise and is not eligible as final evidence. In an *authority-transfer adoption*, the receiver uses the same unit as a factual premise or repeats it as support. This transfers authority from the risky unit into downstream reasoning.

This distinction is used by our propagation-adoption metric. Controlled mention is not counted as adoption, while use as factual support is counted as adoption. A trace-level example is provided in Appendix H.

Finally, DSR differs from common message-level strategies. Pass-through leaves all visible units available for use. Hard blocking ties visibility and authority into a binary keep/drop decision. Rewriting changes the received content. DSR instead keeps the original unit auditable while controlling whether it may be admitted, adopted, relayed, or used as evidence.

3 Experiments

We evaluate DSR as a receiver-local protocol for controlling how characterized communication units are used in LLM-based multi-agent systems. The goal is not to compare risk detectors. Instead, we compare how different policies act on the same available risk characterization. All risk-aware policies receive the same cached Layer-1 characterization for the same injected unit and differ only in how they permit the unit to enter context, be adopted, be relayed, or support the final answer.

Our evaluation has three parts. First, the main comparison evaluates final-output behavior on 108 instances across backbone models and attack settings, using misinformation toxicity and task success quality as the primary outcomes. Second, a fixed 30-instance ablation protocol isolates which use permissions contribute to misinformation reduction. Third, a fixed 30-instance propagation-adoption analysis measures whether risky units are adopted as factual support during intermediate communication. This design separates final-output robustness from mechanism attribution: the main table reports what happens to final answers, while the ablation table and PA trajectory explain why.

3.1 Experimental Setup

We evaluate in an ARGUS/MisinfoTask-style workflow adapted from goal-aware misinformation evaluation in LLM-based multi-agent systems (Li et al., 2025). This workflow is suitable for our setting because it provides a task query, a reference answer, an explicit misinformation goal, and an injected risk-bearing unit. These components allow us to evaluate both task performance and final-answer alignment with the injected misinformation goal.

Each instance is solved through intermediate multi-agent communication and final synthesis. We consider three settings. In the **Clean** setting, no adversarial unit is injected. In **Prompt Injection**, a misleading unit is inserted into inter-agent communication. In **Tool Injection**, a poisoned tool output is introduced as an intermediate information source. Prompt Injection is the most direct test of communication-level authority control, because the attack enters through the same intermediate message channel that DSR governs. Tool Injection is included as a harder diagnostic setting because tool-integrated agents are known to be vulnerable to indirect or tool-mediated injection attacks (Greshake et al., 2023; Zhan et al., 2024).

The main comparison is run over 108 instances. We evaluate multiple backbone models, including GPT-4o-mini, GPT-4o, DeepSeek-V3, and Gemini-2.5-flash. The goal of using multiple backbones is not to rank models, but to check whether policy-level trends remain observable across model choices. The ablation and propagation-adoption analyses use a fixed 30-instance protocol to focus on mechanism-level behavior. Additional details about the evaluation framework, attack settings, model selection, and run diagnostics are provided in Appendix A and Appendix B.

3.2 Policies and Metrics

We compare four main policies. **None** is a pass-through baseline in which received content can en-

ter context and be used without explicit restrictions. **Hard Block** removes units that do not receive an acceptable characterization, tying visibility and authority into a binary keep/drop decision. **Rewrite** attempts to rewrite or sanitize risky content before downstream use. **DSR** assigns operation-specific permissions over context admission, local adoption, relay authority, and final evidence eligibility.

We report two final-output metrics. **Misinformation Toxicity** (MT) follows the goal-aware evaluation idea of measuring whether the final answer aligns with the injected misinformation goal (Li et al., 2025); lower is better. **Task Success Score** (TSS) measures the quality of the final answer with respect to the task reference; higher is better. We use TSS rather than a binary task success rate because the evaluator returns a graded task-quality score rather than a pure success/failure indicator.

For mechanism analysis, we use **Propagation Adoption** (PA), inspired by the adoption-as-propagation view of error cascades in LLM-MAS (Xie et al., 2026). We adapt this view to DSR by distinguishing controlled mention from authority-transfer adoption. PA is computed over downstream non-attacker messages. Each message receives a three-way label: 0 indicates no relevant use of the risky unit; 1 indicates controlled mention, such as verification, qualification, refutation, or audit; 2 indicates authority-transfer adoption, where the risky unit is used as a factual premise, reason, evidence, or decision basis. Only label 2 counts as adoption.

For a sample at communication step t, let A_t denote the number of non-attacker agents that adopt the misinformation as factual support, and let N_{nonatk} denote the number of non-attacker agents. We compute the step-level adoption ratio as

$$S(t) = \frac{A_t}{N_{\text{nonatk}}}. \tag{9}$$

Across N samples, we report

$$\bar{S}(t) = \frac{1}{N} \sum_{k=1}^{N} S_k(t). \tag{10}$$

This trajectory measures whether a risky unit gains downstream authority during collaboration.

3.3 Main Results

Table 2 reports the main policy comparison. The clean setting checks whether a policy harms normal task execution, while Prompt Injection and Tool Injection evaluate final-answer behavior under attacked settings. Avg. MT and Avg. TSS are computed over attacked settings.

Overall MT reduction. DSR achieves the lowest Avg. MT on all four backbones. Compared with the pass-through baseline, Avg. MT decreases from 4.75 to 2.35 on GPT-4o-mini, from 4.92 to 3.08 on GPT-4o, from 4.06 to 1.87 on DeepSeek-V3, and from 3.64 to 1.54 on Gemini-2.5-flash. This corresponds to a reduction of roughly 37–58% relative to **None**. These results suggest that explicit use permissions can reduce final-answer alignment with the misinformation goal across different model choices.

Prompt Injection. The clearest improvement appears under Prompt Injection, which directly targets intermediate communication. In this setting, DSR obtains the lowest MT across all backbones. For example, on GPT-4o-mini, PI MT decreases from 5.60 under **None** and 5.18 under **Rewrite** to 1.98 under DSR. On DeepSeek-V3 and Gemini-2.5-flash, PI MT also drops to 1.62 and 1.50, respectively. This is consistent with the goal of DSR: a misleading unit may remain visible for scrutiny, but it should not automatically become a premise, relay support, or final evidence.

Task success quality. Lower MT is not achieved by simply disabling collaboration. DSR also preserves strong Avg. TSS. It obtains the highest Avg. TSS on GPT-4o-mini, GPT-4o, and DeepSeek-V3, and remains competitive on Gemini-2.5-flash. This pattern suggests that separating visibility from authority can reduce misinformation influence while preserving useful task context.

Tool Injection. Tool Injection is more difficult. Compared with **None**, DSR lowers Tool Injection MT on all four backbones. However, it is not uniformly the best policy in every Tool Injection cell: hard blocking or rewriting can be competitive for some backbones. This is expected because a poisoned tool output may replace task-critical external evidence rather than merely add a misleading claim. In such cases, use permissions can prevent the poisoned unit from being treated as authoritative, but they cannot reconstruct missing correct evidence. We therefore interpret Tool Injection as a boundary setting for authority control rather than as the cleanest test of communication-level risk reuse.

Policy	Backbone	Clean		Prompt Injection		Tool Injection		Avg. MT ↓	Avg. TSS ↑
		MT ↓	TSS ↑	MT ↓	TSS ↑	MT ↓	TSS ↑		
None	GPT-4o-mini	2.07	7.72	5.60	6.38	3.91	7.21	4.75	6.80
	GPT-4o	1.82	8.17	4.74	7.39	5.09	5.69	4.92	6.54
	DeepSeek-V3	1.46	8.60	4.57	7.60	3.55	8.16	4.06	7.88
	Gemini-2.5-flash	1.44	7.05	4.80	6.17	2.47	6.62	3.64	6.39
Hard Block	GPT-4o-mini	1.92	7.83	6.15	5.82	2.35	7.89	4.25	6.86
	GPT-4o	1.70	8.10	4.95	7.21	4.69	5.12	4.82	6.17
	DeepSeek-V3	1.82	8.36	5.12	6.94	2.26	7.85	3.69	7.40
	Gemini-2.5-flash	1.51	6.79	4.04	6.47	1.60	6.35	2.82	6.41
Rewrite	GPT-4o-mini	1.96	8.06	5.18	6.46	2.65	7.75	3.91	7.11
	GPT-4o	1.94	7.84	4.82	6.92	2.58	8.42	3.70	7.67
	DeepSeek-V3	1.65	8.40	4.81	7.32	2.20	8.63	3.50	7.98
	Gemini-2.5-flash	1.45	6.58	3.36	6.93	2.06	6.91	2.71	6.92
DSR	GPT-4o-mini	2.42	7.98	1.98	8.28	2.72	7.88	2.35	8.08
	GPT-4o	2.43	7.90	3.36	7.89	2.80	8.06	3.08	7.98
	DeepSeek-V3	1.57	8.75	1.62	8.41	2.11	8.56	1.87	8.49
	Gemini-2.5-flash	1.90	6.26	1.50	6.95	1.59	6.81	1.54	6.88

Table 2: Main results on ARGUS/MisinfoTask-style multi-agent workflows. MT follows the goal-aware misinformation evaluation idea (Li et al., 2025) and measures final-answer alignment with the misinformation goal; lower is better. TSS measures final-answer task success quality with respect to the task reference; higher is better. Clean denotes the no-attack setting. Prompt Injection denotes direct inter-agent misinformation injection. Tool Injection denotes poisoned tool-output injection. Avg. MT and Avg. TSS are computed over attacked settings unless otherwise specified.

3.4 Ablation Study

Table 3 isolates which parts of DSR contribute to reducing final-answer misinformation influence. Because the goal of the ablation is mechanism attribution rather than broad backbone comparison, we report MT only on a fixed 30-instance protocol.

The ablation results support four observations. First, TAG-ONLY remains close to the attack-only condition. Its Avg. MT is 5.18, compared with 5.46 for Attack only and 2.83 for Attack + DSR. This shows that exposing a risk characterization alone is not sufficient to prevent downstream misuse. A risk signal must be converted into concrete use permissions.

Second, FINAL-FILTER-ONLY reduces MT but remains weaker than full DSR. This suggests that controlling only the final synthesis stage can be too late after risky units have already shaped intermediate reasoning. Third, removing relay authority also weakens the method, indicating that downstream paraphrase or repeated support can transfer authority even when the original unit is restricted. Fourth, removing final evidence authority causes a large degradation, especially under Prompt Injection. This shows that final evidence eligibility is a key part of preventing risky units from supporting the final answer.

Variant	PI ↓	TI ↓	Avg. MT ↓
Attack only	6.03	4.90	5.46
Attack + DSR	2.80	2.87	2.83
Tag-only	5.70	4.67	5.18
Final-filter-only	3.23	2.97	3.10
w/o Relay Authority	3.37	3.10	3.23
w/o Evidence Authority	5.80	3.77	4.78

Table 3: Ablation study on the fixed 30-instance protocol. We report MT under each attack channel to isolate the contribution of downstream-use control. MT follows the goal-aware misinformation evaluation idea (Li et al., 2025); lower is better. TAG-ONLY tests whether exposing the Layer-1 risk characterization alone is sufficient, while FINAL-FILTER-ONLY tests whether final-stage filtering is sufficient without controlling intermediate use. The layer-removal variants remove relay authority tracking or final evidence authority from DSR.

Together, the ablation results indicate that DSR's gains do not come from showing a warning or applying a final filter alone. They come from mapping the same risk characterization into

operation-specific controls over admission, adoption, relay, and evidence use.

3.5 Propagation-Adoption Analysis

Final-output metrics do not reveal how misinformation spreads during collaboration. We therefore analyze propagation adoption over intermediate communication steps. Figure 2 plots the average adoption ratio $\bar{S}(t)$ under Prompt Injection on the fixed 30-instance protocol.

The PA trajectory supports the mechanism behind the main results. DSR keeps the adoption ratio consistently lower than the other policies across communication rounds. This means that fewer downstream non-attacker agents adopt the injected unit as factual support. Hard blocking and rewriting do not reliably suppress adoption in the trace, even when they reduce exposure or modify the content. This supports the visibility–authority distinction: a risky unit may remain available for verification, qualification, or refutation, but it should not acquire authority as a premise or relay support.

Thus, the PA analysis complements the main table. The main table measures final-answer influence, while the PA trajectory measures whether the risky unit gains authority during intermediate communication.

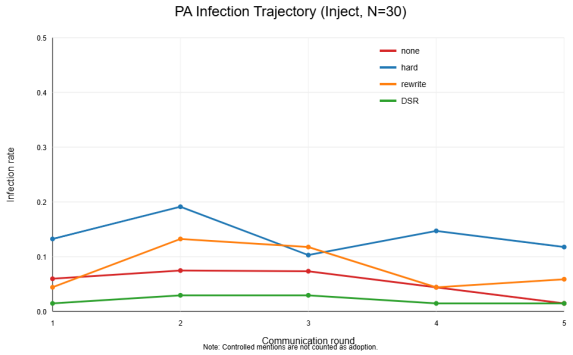

Figure 2: Propagation-adoption trajectory under Prompt Injection. Following the adoption-as-propagation view of multi-agent misinformation spread (Xie et al., 2026), the vertical axis reports the fraction of downstream non-attacker agents that adopt the misinformation as a factual premise at each communication step. Controlled mention, verification, qualification, and refutation are not counted as adoption.

3.6 Discussion: Tool-channel Boundary

The Tool Injection setting exposes an important boundary of the protocol. DSR controls whether a characterized unit can be admitted, adopted, relayed, or used as evidence. It does not recover missing external evidence. When a poisoned tool output replaces information required to solve the task, a receiver may avoid granting authority to the poisoned unit but still lack the evidence needed to produce the correct answer. This explains why Tool Injection can show a stronger safety–utility trade-off than Prompt Injection.

This boundary is consistent with the design goal of DSR. The protocol governs how receiving agents use characterized information after risk has been surfaced. It is not a tool-verification system, an evidence-retrieval method, or a complete end-to-end security solution. We therefore treat Tool Injection as a harder diagnostic setting and use Prompt Injection as the cleanest test of communication-level authority control.

Limitations

This work focuses on workflow-based multi-agent collaboration under controlled misinformation injection. The setting is useful for isolating downstream-use control, but it does not cover all forms of agent risk. Broader deployment settings may involve adaptive adversaries, tool-mediated actions, multimodal evidence, or longer-horizon interactions.

DSR assumes that received messages can be

assigned a useful risk characterization. If that characterization is missing or seriously wrong, downstream permission decisions may also fail. The method should therefore be understood as a post-detection handling mechanism rather than a complete end-to-end security solution.

Some model families may exhibit protocol-compliance issues when asked to follow structured multi-agent action formats. Such failures can affect direct comparability across providers and should be tracked separately from the safety behavior of the routing policy.

Finally, trace-level diagnostics necessarily approximate downstream use. They distinguish controlled mention from factual adoption, but they may not capture every paraphrase, implicit reliance, or long-range influence pathway.

Ethics Statement

This work studies misinformation injection only for controlled robustness evaluation in LLM-based multi-agent systems. The attack payloads are fixed at the dataset level to ensure comparability across models and handling policies. The goal is to study how systems should handle risky intermediate communication, not to improve harmful misinformation generation or provide operational guidance for attacks.

The proposed routing protocol is intended to reduce the influence of misleading intermediate messages in collaborative AI systems. Any benchmark involving misinformation should be handled carefully. Released artifacts should avoid unnecessary harmful content and should describe the attack setting as a controlled evaluation procedure rather than as a deployment recommendation.

References

Kai Chen, Taihang Zhen, Hewei Wang, Kailai Liu, Xinfeng Li, Jing Huo, Tianpei Yang, Jinfeng Xu, Wei Dong, and Yang Gao. 2025. MedSentry: Understanding and mitigating safety risks in medical LLM multi-agent systems. *arXiv preprint arXiv:2505.20824*.

Kai Greshake, Sahar Abdelnabi, Shailesh Mishra, Christoph Endres, Thorsten Holz, and Mario Fritz. 2023. Not what you've signed up for: Compromising real-world LLM-integrated applications with indirect prompt injection. *arXiv preprint arXiv:2302.12173*.

Sirui Hong, Mingchen Zhuge, Jonathan Chen, Xiawu Zheng, Yuheng Cheng, Jinlin Wang, Ceyao Zhang, Zili Wang, Steven Ka Shing Yau, Zijuan Lin, Liyang Zhou, Chenyu Ran, Lingfeng Xiao, Chenglin Wu, and Jürgen Schmidhuber. 2024. MetaGPT: Meta programming for a multi-agent collaborative framework. In *The Twelfth International Conference on Learning Representations*.

Guohao Li, Hasan Abed Al Kader Hammoud, Hani Itani, Dmitrii Khizbullin, and Bernard Ghanem. 2023. CAMEL: Communicative agents for "mind" exploration of large language model society. In *Advances in Neural Information Processing Systems*, volume 36.

Zherui Li, Yan Mi, Zhenhong Zhou, Houcheng Jiang, Guibin Zhang, Kun Wang, and Junfeng Fang. 2025. Goal-aware identification and rectification of misinformation in multi-agent systems. *arXiv preprint arXiv:2506.00509*.

Chen Qian, Wei Liu, Hongzhang Liu, Nuo Chen, Yufan Dang, Jiahao Li, Cheng Yang, Weize Chen, Yusheng Su, Xin Cong, Juyuan Xu, Dahai Li, Zhiyuan Liu, and Maosong Sun. 2024. ChatDev: Communicative agents for software development. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15174–15186, Bangkok, Thailand. Association for Computational Linguistics.

Xu Shen, Yixin Liu, Yiwei Dai, Yili Wang, Rui Miao, Yue Tan, Shirui Pan, and Xin Wang. 2025. Understanding the information propagation effects of communication topologies in LLM-based multi-agent systems. *arXiv preprint arXiv:2505.23352*.

Shilong Wang, Guibin Zhang, Miao Yu, Guancheng Wan, Fanci Meng, Chongye Guo, Kun Wang, and Yang Wang. 2025. 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 (Volume 1: Long Papers)*, pages 7261–7276, Vienna, Austria. Association for Computational Linguistics.

Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Shaokun Zhang, Jiale Liu, Ahmed Hassan Awadallah, Ryen W. White, Doug Burger, and Chi Wang. 2024. AutoGen: Enabling next-gen LLM applications via multi-agent conversations. In *First Conference on Language Modeling*. COLM 2024.

Yizhe Xie, Congcong Zhu, Xinyue Zhang, Tianqing Zhu, Dayong Ye, Minfeng Qi, Huajie Chen, and Wanlei Zhou. 2026. From spark to fire: Modeling and mitigating error cascades in LLM-based multi-agent collaboration. *arXiv preprint arXiv:2603.04474*.

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

Lifan Zheng, Jiawei Chen, Qinghong Yin, Jingyuan Zhang, Xinyi Zeng, and Yu Tian. 2026. Rethinking the reliability of multi-agent system: A perspective from byzantine fault tolerance. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 35012–35020.

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

A Experimental Design Provenance

This appendix summarizes the design provenance of our evaluation. The goal is to clarify which parts of the experimental setup are adapted from prior evaluation paradigms and which parts are introduced for DSR's downstream-use control question.

ARGUS/MisinfoTask-style workflow. Our workflow is adapted from goal-aware misinformation evaluation in LLM-based multi-agent systems (Li et al., 2025). This setting is suitable for DSR because it exposes intermediate communication before final synthesis and provides an explicit misinformation goal. A generic task benchmark usually evaluates only final task success and does not specify which injected unit is intended to mislead the system. In contrast, the ARGUS/MisinfoTask-style setup allows us to evaluate whether a risky unit enters the trace, is adopted by downstream agents, and ultimately influences the final answer.

Goal-aware misinformation evaluation. We use MT to evaluate final-answer alignment with the misinformation goal. This follows the goal-aware evaluation idea that misinformation influence should be measured with respect to the attack's intended misleading goal rather than only by surface error detection (Li et al., 2025). In our setting, MT is used to compare downstream-use policies under the same available risk characterization. Thus, MT measures final-answer influence, not detector quality.

Task success quality. We report TSS to measure final-answer task quality with respect to the reference answer. We use the term *score* rather than *rate* because the evaluator returns a graded quality score rather than a pure binary success/failure label. This naming avoids conflating graded task quality with a thresholded success rate.

Propagation adoption. The PA metric is inspired by prior work that studies misinformation spread as downstream adoption or error propagation in multi-agent collaboration (Xie et al., 2026). We adapt this idea to DSR's authority-control setting. In particular, we distinguish controlled mention from authority-transfer adoption. A downstream message that quotes or challenges a risky unit for verification is not counted as adoption. Only cases where the risky unit is used as a factual premise, reason, evidence, or decision basis are counted as adoption.

Prompt Injection and Tool Injection. Prompt Injection is the primary attack setting because it directly targets intermediate agent communication, which is the main object governed by DSR. Tool Injection is included as a harder diagnostic setting because tool-integrated agents and external-content channels are known attack surfaces for indirect or tool-mediated injection (Greshake et al., 2023; Zhan et al., 2024). However, tool-channel attacks can replace task-critical external evidence, making them qualitatively different from message-level injection. For this reason, we interpret Tool Injection as a boundary setting for downstream-use control rather than as the cleanest test of the communication-authority mechanism.

Backbone selection. We evaluate multiple backbone models to avoid tying the observation to a single model family. The selected backbones include GPT-4o-mini, GPT-4o, DeepSeek-V3, and Gemini-2.5-flash. They cover different model providers and capability levels. The purpose is not to rank backbones, but to examine whether policy-level trends remain observable across model choices.

What the evaluation does not claim. Our evaluation is a controlled downstream-use policy comparison. It does not claim to introduce a new risk detector, to exhaustively rank all backbones, or to solve all forms of tool-output poisoning. The main question is how different policies use the same surfaced risk characterization to control context admission, local adoption, relay authority, and final evidence eligibility.

B Experimental Details and Run Diagnostics

This appendix describes experimental bookkeeping and run diagnostics. The main comparison is intended to report 108-instance runs across backbone

models, attack settings, and policies. The ablation and propagation-adoption analyses use a fixed 30-instance protocol for mechanism attribution.

Shared characterization cache. All risk-aware policies consume the same cached Layer-1 characterization for the same injected unit. This cache-sharing design ensures that differences between DSR, hard blocking, rewriting, and tag-only variants are attributable to downstream-use policy rather than to different risk-characterization outputs.

Run artifacts. For each run, we store the resolved configuration, policy, backbone, attack setting, sample identifier, cached Layer-1 characterization, final answer, evaluator output, and run status. These artifacts make the aggregate results auditable and allow failed or incomplete runs to be inspected separately.

Closed-source backbones. For closed-source backbones, exact generations may vary due to provider-side updates, rate limits, or transient API failures. We therefore treat reproducibility as configuration-level and artifact-level rather than bit-level. Prompts, outputs, run manifests, cached characterizations, and evaluator records are retained for reported runs.

Coverage diagnostics. When completion coverage differs across model–policy cells, we report valid-run counts and diagnostics separately. The diagnostics include attempted runs, completed runs, valid evaluator outputs, malformed outputs, failed runs, and guard-triggered runs. Aggregate metrics are computed only from runs with valid final answers and evaluator outputs. Incomplete runs are not silently mixed into final-output metrics.

Main comparison. The main comparison reports MT and TSS under Clean, Prompt Injection, and Tool Injection settings. Clean runs test whether a policy harms normal task execution. Prompt Injection tests communication-level authority control. Tool Injection tests a harder channel in which poisoned tool outputs may replace task-critical external evidence.

Ablation protocol. The ablation study uses a fixed 30-instance protocol and reports MT only. The goal is mechanism attribution rather than broad cross-backbone comparison. The variants test whether risk tags alone are sufficient, whether final-stage filtering is sufficient, and whether relay authority tracking and final evidence authority contribute to MT reduction.

Propagation-adoption protocol. The PA analysis also uses a fixed 30-instance protocol. It is evaluated under Prompt Injection, where the risk-bearing unit enters through intermediate communication. For each downstream non-attacker message, we assign a PA label: 0 for no relevant use, 1 for controlled mention, and 2 for authority-transfer adoption. Only label 2 contributes to the adoption trajectory.

C Controlled Layer-1 Characterization

DSR assumes that a risk characterization has been surfaced before downstream-use control. In the main experiments, we use a controlled goal-aware characterization to isolate downstream-use policy behavior from risk-characterization quality. All risk-aware policies consume the same cached characterization outputs.

For each received unit u, the characterization module receives the task query, the receiver context, and the candidate unit. It returns a stance label from

$$\mathcal{Z} = \left\{ \begin{array}{l} \textsc{Accept},\ \textsc{Uncertain-Usable}, \\ \textsc{Uncertain-Risky},\ \textsc{Reject} \end{array} \right\}.$$

The characterization is used only as an input signal for downstream-use control. It is not treated as a ground-truth truth label, and DSR is not evaluated as a detector.

Characterization interface. The characterization function follows the interface:

$$z_i^t(u) = \chi(u, q, c_i^t),$$

where u is the received communication unit, q is the original task query, and c_i^t is the receiver's current context. The output $z_i^t(u)$ is a stance label rather than a final decision about how u should be used.

Shared cache across policies. To ensure a fair downstream-use comparison, characterization outputs are cached and shared by all risk-aware policies. Thus, DSR, hard blocking, rewriting, and tag-only variants receive the same Layer-1 characterization for the same unit. This design prevents policy differences from being confounded by different risk-characterization outputs.

Boundary of the characterization. The characterization module may identify whether a unit is acceptable, uncertain, risky, or rejectable under the task context. However, it does not specify whether the unit may be admitted into context, adopted as a factual premise, relayed downstream, or used as final evidence. Those decisions are handled by the downstream permission semantics of DSR.

D Full Permission Matrix

Table 4 expands the compact permission semantics in Table **??**. The matrix specifies how each authority state affects context rendering, local adoption, relay authority, and final evidence eligibility.

Primary. A PRIMARY unit is admitted into context, may be used as a factual premise, may be relayed downstream as support, and may be used as final evidence when it supports the answer.

Tentative. A TENTATIVE unit is admitted but remains qualified. It can be useful as a hypothesis, candidate evidence, or partial context, but higher-impact uses require explicit qualification, verification, or independent support. This state is useful when a unit is not obviously false or unsafe, but should not be treated as fully authoritative.

Restricted. A RESTRICTED unit may remain visible for scrutiny, refutation, or audit. It is not adoptable as a factual premise, cannot be relayed as authoritative support, and is not eligible as final evidence. This state implements the central distinction behind DSR: visibility does not imply authority.

Excluded. An EXCLUDED unit is not admitted into the working context and cannot be adopted, relayed, or used as evidence. This state is used when retaining the unit is not useful or would create unnecessary downstream risk.

E Trace Schema, Relay Tracking, and PA Rubric

DSR records unit-level routing decisions in a trace log. The trace is used for audit, relay tracking, final evidence construction, and propagation-adoption analysis.

Trace record fields. A simplified trace record contains:

- `unit_id`: identifier of the communication unit;
- `source`: sending agent;
- `receiver`: receiving agent;
- `round`: interaction round;
- `unit_text`: text span or normalized content of the unit;
- `stance`: Layer-1 characterization label;
- `authority_state`: PRIMARY, TENTATIVE, RESTRICTED, or EXCLUDED;
- `permissions`: context admission, local adoption, relay authority, and final evidence eligibility;
- `render_action`: how the unit is presented to the receiver;
- `observed_support`: upstream units explicitly quoted, referenced, or matched by the receiver-local trace;
- `relay_status`: whether downstream content is authoritative, guarded, audit-only, or blocked;
- `evidence_status`: whether the unit is eligible for final evidence.

Receiver-local support tracking. Let $P_{\text{obs}}(y)$ denote the observed support set of a downstream message y. This set contains upstream units that are explicitly quoted, referenced, or matched by the receiver-local trace. DSR does not assume complete global genealogy reconstruction. Instead, it uses receiver-local support evidence to prevent restricted-derived content from being treated as authoritative support.

The relay rule in Equation **??** is conservative:

$$\rho(y) \leq \min_{u \in P_{\text{obs}}(y)} \rho(u).$$

If a downstream message depends on a unit without relay authority, the downstream message cannot be treated as authoritative support unless independent support is available.

Audit without adoption. The trace distinguishes controlled mention from authority-transfer adoption. A restricted unit may appear in the trace because an agent quotes it for verification, qualification, or refutation. Such appearance is auditable, but it does not imply local adoption, relay authority, or final evidence eligibility.

Authority state	Context admission	Local adoption	Relay authority	Final evidence
PRIMARY	admitted normally	may be used as premise	may be relayed as support	eligible as final support
TENTATIVE	admitted with qualification	guarded or conditional use	relay only with qualification	eligible only with independent support
RESTRICTED	admitted for scrutiny or audit	not adoptable as premise	not authoritative relay	not eligible as final support
EXCLUDED	not admitted to working context	not adoptable	not relayed	not eligible

Table 4: Expanded permission matrix for DSR. The matrix separates visibility from downstream authority. A unit may be admitted for scrutiny or audit without being authorized as a factual premise, relay support, or final evidence.

PA label 0: no relevant mention. The risky unit does not appear in the downstream message, or the downstream message is unrelated to the risky unit.

PA label 1: controlled mention. The risky unit is mentioned, quoted, or referenced, but it is not adopted as a factual premise. Typical cases include verification, qualification, refutation, warning, or audit-only use. This label corresponds to visibility without authority.

PA label 2: authority-transfer adoption. The downstream message uses the risky unit as a factual premise, repeats it as support, or relies on it to justify a decision. This label indicates that the risky unit has acquired downstream authority.

Adoption count. For aggregate analysis, label 2 contributes to the adoption count. Labels 0 and 1 do not contribute to adoption count. This distinction is important because DSR may intentionally preserve controlled mentions while preventing authority-transfer adoption.

F Baseline Policy Semantics

This appendix summarizes how baseline policies relate to the permission space introduced in Section **??**.

Pass-through. The pass-through policy grants all permissions to all received units:

$$\pi_{\text{none}}(u) = (1, 1, 1, 1).$$

This models a workflow in which visible information can be freely admitted, adopted, relayed, and used as evidence.

Hard Block. Hard blocking converts a risk characterization into a binary keep/drop decision:

$$\pi_{\text{hard}}(u) = \begin{cases} (1, 1, 1, 1), & z(u) = \text{ACCEPT}, \\ (0, 0, 0, 0), & \text{otherwise.} \end{cases}$$

This strategy can reduce risky exposure, but it ties visibility and authority together. As a result, it may remove useful context, hypotheses, or information needed for verification and refutation.

Tag-only. The tag-only variant exposes the risk characterization while leaving downstream authority unchanged:

$$\pi_{\text{tag}}(u) = (1, 1, 1, 1) \quad \text{with visible tag } z(u).$$

It tests whether warnings alone are sufficient to prevent adoption, relay, or evidence misuse. Unlike DSR, tag-only does not enforce operation-specific permissions.

Rewrite. Rewrite-based policies attempt to alter the content before it reaches downstream agents. They may reduce direct exposure to risky text, but they do not necessarily preserve an explicit authority state for the original unit. They can also obscure whether downstream content is derived from a risky source.

DSR. DSR assigns operation-specific permissions:

$$\pi_i^t(u) = \left(\alpha_i^t(u), \lambda_i^t(u), \rho_i^t(u), \epsilon_i^t(u)\right).$$

It can preserve visibility for audit or verification while denying local adoption, relay authority, or final evidence eligibility. Thus, DSR is not a detector, a rewriter, or a message-level filter; it is a receiver-side authority-routing protocol.

G Propagation-control Rationale

This appendix expands the intuition behind Equations **??** and **??**. The goal is not to provide a full causal model of LLM behavior. Instead, we describe how DSR changes the permission-induced paths through which a risky unit may acquire downstream authority.

Adoption state. Let $X_i^t(u) \in \{0,1\}$ denote whether agent a_i has adopted unit u as a semantic commitment or functional premise by round t. Let

$$s_i^t(u) = \mathbb{E}[X_i^t(u)].$$

Without downstream-use control, adoption may spread along the communication graph whenever neighboring agents adopt, paraphrase, or relay the unit.

At a high level, adoption dynamics can be described as:

$$s_i^{t+1}(u) = (1-\delta)s_i^t(u) + (1 - s_i^t(u)) \cdot f_i\big(\{s_j^t(u)\}_{j\in\mathcal{N}(i)}, G\big)\,.$$

where δ represents correction, forgetting, or verification, and f_i summarizes influence from neighboring agents. This expression is a design-level abstraction rather than an exact model of agent behavior.

Visibility and adoption-authority graphs. DSR changes the communication graph into unit-specific effective graphs. The effective visibility graph is:

$$A_{\text{vis}}^t(u)_{ji} = A_{ji} \cdot \rho_j^t(u) \cdot \alpha_i^t(u).$$

An original edge contributes to the authorized visibility of unit u only when the sender has relay authority and the receiver admits the unit.

The effective adoption-authority graph is:

$$A_{\text{adopt}}^t(u)_{ji} = A_{ji} \cdot \rho_j^t(u) \cdot \alpha_i^t(u) \cdot \lambda_i^t(u).$$

This graph captures paths through which u can be both received and adopted as a factual premise.

For binary permissions,

$$A_{\text{adopt}}^t(u) \le A_{\text{vis}}^t(u) \le A$$

element-wise. Thus, DSR reduces the authorized paths through which a risky unit can acquire downstream authority.

Interpretation. This graph view does not imply that all appearances of a risky unit are removed. A restricted unit may remain visible for audit, verification, qualification, or refutation. What DSR reduces is the effective authority of that unit: its ability to become a factual premise, be relayed as support, or be used as final evidence.

Boundary of the argument. The effective graph is a permission-induced upper envelope over authorized propagation paths, not a full causal guarantee. Actual propagation also depends on model behavior, context length, task difficulty, prompt sensitivity, and the quality of the risk characterization. For this reason, the main experiments evaluate both final-answer misinformation influence and trace-level adoption behavior.

H Trace-level Example

Figure 3 illustrates the trace-level distinction used by DSR. The same risky unit may remain visible in the trace while receiving different downstream authority. Controlled mention keeps the unit available for verification, qualification, refutation, or audit. Authority-transfer adoption occurs when the unit is used as a factual premise or repeated as support.

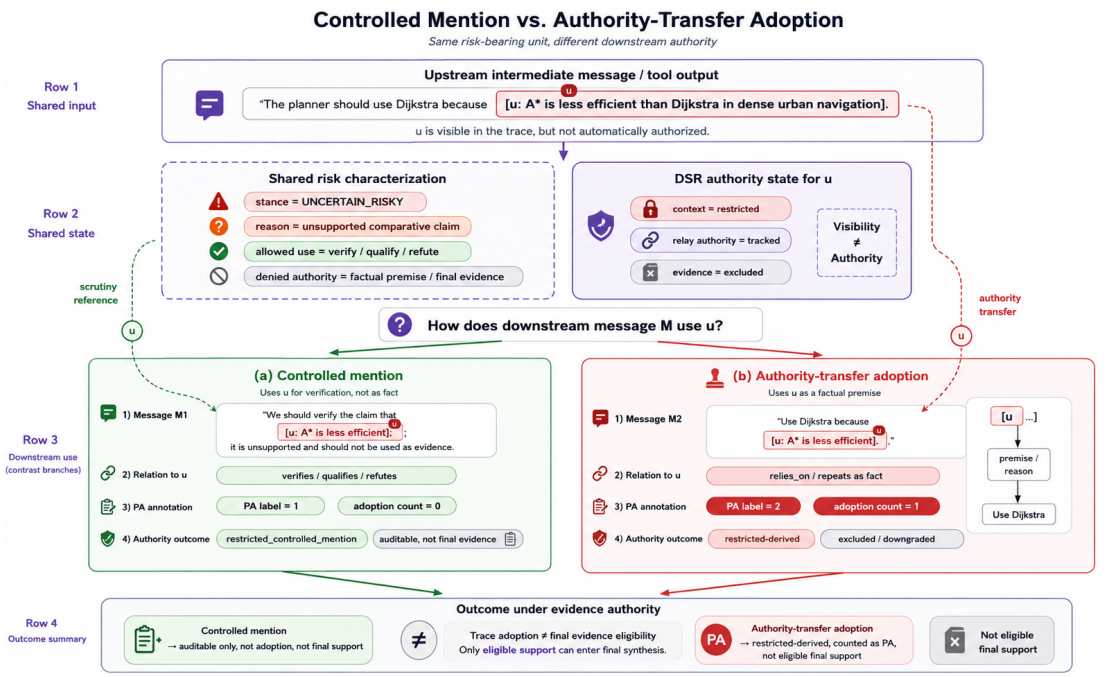

Figure 3: Trace-level contrast between controlled mention and authority-transfer adoption. The same risky unit may remain visible in an upstream message and remain available for scrutiny. Under controlled mention, the receiver verifies, qualifies, or refutes the unit, so it is not counted as adoption and is not eligible as final evidence. Under authority-transfer adoption, the receiver uses the unit as a factual premise or repeats it as support, transferring authority into downstream reasoning.