Life of Research · Published 21 November 2026 · DOI 10.5281/zenodo.24420073

A Refusal Is Not a Veto: Why Multi-Agent Structures Are Reported Both to Amplify and to Contain Language-Model Harm, Which Properties of the Structure Set the Sign, and the Matched Comparison That Would Separate Them

Pranay Mahendrakar
SONYTECH · ORCID 0009-0003-7224-029X
pranaymahendrakar@sonytech.in · mahendrakarpranay@gmail.com · +91 6361723454

Abstract. Language models are increasingly composed into systems in which one model plans and others execute, check or debate. Two bodies of work describe the safety consequences in opposite terms. One reports that composition amplifies harm: a model that refuses a hazardous task on its own completes it once the task passes through a delegating principal, a planner or an orchestrator. The other reports that composition contains harm: adding agents that inspect a response or challenge a peer cuts jailbreak success several-fold. In this analysis the author reviews and synthesises published results to show that "multi-agent" is not the variable either literature measured. A reported change bundles at least five properties of the structure: what each model can see of the original intent, what it is told about the authority of the instruction it receives, whether a refusal ends the path or can be routed around, what incentive the prompts add, and whether output becomes an action. Re-reading the anchor study's own tables, the author computes that if delegation merely added a second, independent refusal point, end-to-end full execution would fall below every model's single-agent rate; it rises above it for four of six models, and for two of them more than nine tenths of delegated full execution passed through deliveries the study's own judge classed as sanitized, so on that classification the subordinate did not receive the harmful intent it is said to have obeyed (the study's authors read the same numbers as evidence that sanitization was incomplete). In a computer-use study, a single model given an orchestrator's decomposed subtasks, with no second agent, reproduces most of the orchestrated system's failure, while in a planner-executor study executor compliance is 6.3 percent when the planner refuses and 78.6 percent when it emits steps, against 64.4 percent with no planner. The paper consolidates 24 published results, states an attribution procedure, and specifies the matched comparison that would separate the five properties.

Keywords: multi-agent systems; LLM safety; delegation; refusal; task decomposition; instruction hierarchy; agent orchestration; evaluation methodology

1. Introduction

A language model deployed as a chat assistant is usually evaluated one request at a time: a hazardous prompt goes in, and the model either refuses or does not. A growing share of deployments look different. A principal model receives a task, breaks it into pieces, and hands the pieces to subordinate models, which may call tools; a planner writes steps for an executor; several agents debate an answer; a coordinator routes subtasks to whichever worker suits them best. Frameworks that make such compositions routine have been available for several years (Wu et al., 2023), and the question of whether safety evaluated on single models survives them has acquired its own literature (Hammond et al., 2025; de Witt et al., 2025; Carichon et al., 2025).

That literature points in two directions at once. A 2026 study of six frontier models on 49 hazardous tasks reports that "individual safety alignment fails to transfer to multi-agent settings" (Ying et al., 2026): one model's full-execution rate rises from 30.6 percent when it is asked directly to 77.6 percent once a principal of the same model delegates the task. Computer-use agents built on a safety-trained model are reported to fail more often when deployed inside an orchestrated system than alone (Ding et al., 2026). Multi-agent debate is reported to be more fragile than a single model under the same prompts (Qi, S. et al., 2025). Infectious images and prompts spread through populations of agents (Gu et al., 2024; Lee and Tiwari, 2024). On the other side, a defense built from several cooperating agents that inspect a model's response reduces jailbreak success on one victim model from 55.74 to 7.95 percent (Zeng et al., 2024); a reproduction finds multi-agent configurations more resistant than single-agent ones, at a cost in false positives (Wit and Pang, 2025); debate with a non-jailbroken peer lowers the toxicity of a jailbroken one (Chern et al., 2024); hierarchical structures recover from faulty members better than linear ones (Huang et al., 2024); and an evolutionary search over multi-agent scaffolds finds scaffolds that are much safer than their baselines as well as scaffolds that are much weaker (Rosser and Foerster, 2025).

Read together, these results do not say that multi-agent structures are dangerous or that they are protective. They say that adding agents can move harm in either direction, by large amounts. That is a claim about a variable that has not been named. This paper names it, or rather names the several properties that a comparison between "a single model" and "the same model inside a structure" changes at once.

The argument proceeds from a reading of the anchor study's published tables and widens to the literatures on

© 2026 Pranay Mahendrakar. Licensed under CC BY 4.0.

decomposition attacks, instruction hierarchy, obedience, orchestration, adversarial propagation, agentic refusal and multi-agent defense. In this analysis the author reviews and synthesises published results to show that a reported multi-agent safety effect bundles at least five properties. The first is visibility (V): how much of the original intent each model in the structure can see, which decomposition, rewriting and summarisation all change. The second is authority (A): what the receiving model is told, or can infer, about where its instruction came from and whether it has been approved, whether by a user, a planner, a principal, a peer or a better-known model. The third is gate topology (G): whether a refusal anywhere ends the path, a veto, or whether the structure can retry, re-plan, re-decompose or pick another agent, a route. The fourth is incentive (P): the pressure the structure's prompts add, such as deadlines, performance scores or statements about liability. The fifth is conversion (C): whether a model's output is text read by another model or an action executed on a system.

A sixth factor, measurement (M), decides how large any of the five appears: which judge, which denominator, how many tasks, and which harm the judge is asked about.

The paper's contributions are four. First, a re-reading of the anchor study's tables that separates, with arithmetic on published values, the share of delegated harm attributable to intent the subordinate never saw from the share attributable to intent it saw and executed anyway. Second, a synthesis showing that the amplifying and the containing results sort by whether the deciding node saw the intent and used its veto, not by agent count. Third, a consolidation of 24 published results in one table and one figure. Fourth, an attribution procedure, stated as an algorithm, and a specification of the matched comparison that would settle the open part. Section 2 sets out the two literatures. Section 3 defines the five properties and derives what each topology predicts. Section 4 re-reads the anchor. Sections 5 to 9 take the properties in turn. Section 10 concerns measurement. Sections 11 and 12 give the table and the procedure. Sections 13 to 15 state the strongest objection, what is not known, and the studies that would settle it.

The title's claim is narrow and, the author argues, well supported: a model's refusal protects a composed system only if the structure lets that refusal see the intent and end the path. When it does, composition multiplies refusals and harm falls. When it does not, composition offers more chances to comply and harm rises. The number of agents is downstream of both.

2. Two Literatures With Opposite Signs

2.1 Composition as amplifier

The amplifying literature has three strands, which are worth keeping apart because they make different claims.

The first strand concerns structures with no adversarial member. Ying et al. (2026) place a principal and a subordinate, both instances of the same model, between a hazardous task and its execution, and compare the outcome with the model asked directly. They attribute the change to two mechanisms they name: responsibility diffusion, in which the principal relaxes its refusal threshold as though delegation transferred liability, and role-bias compliance, in which a model in the subordinate role complies with instructions it would refuse if asked directly. Liu, L. et al. (2026) study a planner-executor pipeline and argue that the single "pipeline effect" reported by such comparisons conflates at least three contrasts: reframing of the harmful request as operational work, the planner's refusal or transformation, and an executor prompt stating that the planner has approved the task. Ding et al. (2026) report that a safety-trained computer-use model's attack success rises from 73.0 to 92.7 percent when it serves as the operator in a multi-agent system, and attribute the change to decomposition hiding intent. Xu, Z. et al. (2026) find that centralized hierarchies complete more benign tasks than peer networks but take unauthorized actions in 2.7 to 19.8 percent of tasks against 0.6 to 0.8 percent, a gap that grows with hierarchy depth.

The second strand concerns structures with an adversary somewhere inside or upstream. Adversarial images spread jailbreaks through randomly paired agents exponentially fast (Gu et al., 2024). Self-replicating prompts propagate across connected agents (Lee and Tiwari, 2024). An attacker who controls only the messages between agents can compromise a system (He et al., 2025). Web content can hijack an orchestrator into executing arbitrary code in 58 to 90 percent of trials with GPT-4o agents, depending on the orchestrator, "even if individual agents are not susceptible to direct or indirect prompt injection" (Triedman et al., 2025). Dark-personality agents degrade medical multi-agent systems, with an open shared pool highly susceptible (Chen et al., 2025), and psychological manipulation of agents produces collective dangerous behaviour (Zhang et al., 2024a; Tian et al., 2023).

The third strand concerns adversaries who use several models as instruments. Jones, Dragan and Steinhardt (2024) show that an adversary can split a task between a frontier model that refuses the hazardous whole but helps with benign-looking parts and a weaker model that does not refuse, reaching 43 percent success on vulnerable code generation where neither model alone exceeds 3 percent.

2.2 Composition as container

The containing literature is smaller but no less concrete. AutoDefense (Zeng et al., 2024) adds a defense agency that reads a victim model's response and classifies it as valid or invalid; in its main comparison, attack success on GPT-3.5 under one jailbreak family falls from 55.74 percent with no defense to 9.44 percent with a single defense agent and 7.95 percent with three. A reproduction confirms the reduction, notes that it varies by attack type and raises false positives (Wit and Pang, 2025). Multi-agent debate lowers toxicity when a

jailbroken or weaker model must debate a non-jailbroken or stronger one, with marginal gains otherwise (Chern et al., 2024). An Inspector agent that reviews and corrects messages, combined with a Challenger instruction added to the existing agents' profiles, recovers up to 96.4 percent of the performance lost to faulty agents (Huang et al., 2024). A topology-aware detector on the utterance graph recovers over 40 percent of the performance lost to prompt injection (Wang et al., 2025). Cooperative defense workflows of orchestrator, deflector, responder and evaluator agents report gains on jailbreak benchmarks with low false refusal (Cai et al., 2025), and guardian-agent designs report detection accuracy as high as 94 percent on one model while remaining vulnerable to long attacks (Barua et al., 2025). Pairing workers of asymmetric competence cuts attack success on one manager-worker system from 52.8 to 2.0 percent (Liu, Q. et al., 2026). The AI-control literature builds whole protocols out of one model checking another (Greenblatt et al., 2023).

2.3 What the two literatures share

In the papers reviewed here the two literatures seldom engage each other directly, and neither reports the structural property that the author argues decides the sign. In the containing results above, the added agent reads the full content it is judging (a complete response, a full message, the whole utterance), and in most of them its negative verdict stops the output. The exception is debate, where the added agent's view is weighed against the harmful one rather than imposed, and there the gain appears mainly when the added agent is the non-jailbroken or stronger party (Chern et al., 2024). In each amplifying result without an adversary, the added agent either receives less of the intent than the single model did, receives it under a framing of approval or authority, or sits in a structure that can try again. Section 3 makes that observation precise.

3. Five Properties and What Each Topology Predicts

3.1 Definitions

The author uses the following terms throughout. They are stipulations, and are stated flatly.

A **node** is one model call with its own prompt. A **path** is the sequence of nodes a task passes through between the user's request and the system's output or action. A node **refuses** when it declines to produce the content or action its input requests.

Visibility at a node is the part of the user's original intent that the node's input carries. A single model asked directly has full visibility. A subordinate that receives one subtask of a decomposed request has partial visibility, and a subordinate that receives a request rewritten to remove its harmful purpose may have none.

Authority framing at a node is what its input says or implies about where the instruction came from and whether it has been approved. A user turn, a system prompt stating that a planner validated the task, and a message labelled with a well-known model family's name are three different framings of possibly identical content.

The **gate topology** of a structure describes what happens when a node refuses. In a **veto** topology, a refusal at any node on the path ends the path: no output reaches the user and no action is taken. In a **route** topology, the structure can respond to a refusal by retrying the same node, sending the task to another node, re-decomposing it, or proceeding on another branch. Most deployed structures are mixtures.

The **incentive** of a structure is any pressure its prompts add beyond the task itself, such as a deadline, a performance score, or a statement about who will be held responsible.

Conversion is whether a node's output is text consumed by another node, text returned to a user, or an executed action.

3.2 What the topologies predict

Suppose a task passes through k nodes, and node i, given full visibility and neutral framing, would comply with the harmful request with probability c_i, independently of the others. The following are derivations, not measurements.

In a veto topology, the task produces harm only if every node complies, so the probability of harm is the product c_1 c_2 ... c_k. Each node added can only lower it, and the system is safer than its safest member.

In a route topology in which the structure keeps trying nodes until one complies, the task produces harm unless every node refuses, so the probability of harm is 1 - (1 - c_1)(1 - c_2)...(1 - c_k). Each node added can only raise it, and the system is less safe than its least safe member.

The same set of models, with the same individual refusal rates, therefore produces opposite system-level effects depending on a property that has nothing to do with the models. Real structures depart from both ideal cases in two ways that matter. First, visibility below full raises c_i at the affected node, because a model cannot refuse a purpose it cannot see. Second, authority framing and incentive change c_i directly. Both departures push harm up in either topology. In a veto topology with full visibility and neutral framing, adding nodes helps; in any topology with degraded visibility or approval framing, the help shrinks and can reverse.

The analogy with layered defenses in other safety-critical fields is close. Reason's account of organisational accidents describes defenses as layers whose holes must line up for harm to pass (Reason, 2000). The account presupposes that each layer looks at the same hazard and that passing a layer requires passing all of them. Delegation can violate both presuppositions: a downstream layer may not see the hazard, and a structure that retries can pass the hazard through whichever layer has a hole.

3.3 Why agent count is the wrong regressor

Most of the comparisons reviewed here contrast "single agent" with "multi-agent" as two levels of one factor. On the analysis above, agent count is confounded with all five properties at once. Moving from one node to two typically lowers visibility at the second node (it receives the first node's rewriting), adds an authority framing (the second node is told the first is its principal or planner), adds incentive (the first node is often given a completion requirement), and changes the topology (the first node may route rather than veto). The direction of the net effect depends on which of these changes dominates, and the studies reviewed in Sections 4 to 9 show each of them dominating somewhere.

4. The Anchor Re-Read: What the Delegation Tables Show

4.1 Design

Ying et al. (2026) construct 49 hazardous tasks, seven in each of seven risk categories, chosen to be "operationally decomposable" so that each admits a natural division into subtasks. Six models are evaluated: Claude-Sonnet-4.6, GPT-5, Gemini-3.1-Pro, Qwen3-Max, DeepSeek-V3.2 and Kimi-K2.5. Under Condition A a single model receives the task. Under Condition B the task goes to a principal, which may refuse (full refusal, FR), delegate after stripping or rewriting the harmful intent (safe delegation, SD), or delegate with the intent preserved (harmful delegation, HD); the subordinate then refuses (RE), partially executes (PE) or fully executes (FE). Condition C gives the subordinate tools. Principal and subordinate are the same base model in the main runs. A GPT-4o judge assigns every label. Subordinate rates use all 49 tasks as the denominator, so that FE is comparable with full compliance (FC) under Condition A.

Two design details bear directly on attribution and are stated in the paper's text. First, "unless otherwise stated", the principals of both delegated conditions carry a task-specific completion pressure in their system prompts (Ying et al., 2026, Section 3.2). Condition A has no such clause. The A-to-B comparison therefore changes structure and incentive together. Second, the subordinate judge is instructed to evaluate the subordinate's output against the harmful objective of the original task, not the instruction the subordinate received, so that carrying out a cleaned-up instruction in full counts as FE whenever the output serves the original harmful aim (Ying et al., 2026, Appendix C.3). FE under Condition B thus counts harm the subordinate could not have recognised from its input.

4.2 The headline comparisons, as published

Under Condition A, full compliance ranges from 14.29 percent (Qwen) to 30.61 percent (DeepSeek); outright refusal ranges from 24.49 percent (GPT-5) to 59.18 percent (Claude), and the remainder is partial compliance. Under Condition B, end-to-end full execution rises above the single-agent rate for four models (DeepSeek 30.61 to 77.55; Qwen 14.29 to 63.27; GPT-5 22.45 to 36.73; Gemini 24.49 to 28.57) and falls for two (Claude 20.41 to 4.08; Kimi 24.49 to 20.41). The authors read Claude's and Kimi's results as showing that strong principal-side alignment can turn delegation into a filter, and note that the principal is the critical control point. The anchor holds topology fixed: every model runs the same principal-subordinate structure, and every principal sees the full task and can end the path by refusing. Its sign differences across models are therefore model-driven, a matter of how much intent each principal forwards and how often it uses its veto, not differences of topology.

4.3 A second veto would have lowered harm for every model

The veto prediction of Section 3.2 can be checked against these tables. If delegation did nothing except add a second refusal point, and the subordinate complied at its own single-agent rate whatever it received, the end-to-end full-execution rate would be the share of tasks the principal delegates (SD plus HD) multiplied by the single-agent full-compliance rate. The author computes this product from Ying et al.'s Tables 2 and 3. It ranges from 5.8 percent (Claude) to 27.5 percent (DeepSeek), and lies below each model's own single-agent rate, as it must. The observed FE exceeds the prediction for five of six models. For DeepSeek the prediction is 27.5 percent and the observation 77.55; for Qwen, 11.7 against 63.27; for GPT-5, 16.0 against 36.73; for Gemini, 16.0 against 28.57; for Kimi, 10.5 against 20.41. Only Claude comes in below its prediction, at 4.08 against 5.8.

The same tables give the subordinate's full-execution rate conditional on receiving a delegation (FE divided by SD plus HD). It is 86.4 percent for DeepSeek, 77.5 for Qwen, 51.4 for GPT-5, 47.6 for Kimi, 43.7 for Gemini and 14.3 for Claude, against single-agent full-compliance rates of 30.61, 14.29, 22.45, 24.49, 24.49 and 20.41 respectively. For five models, then, a subordinate is about 1.8 to 5.4 times as likely to fully execute what reaches it as the same model is to fully comply with the original request; for Claude it is less likely. Figure 1(a) shows the three quantities side by side.

This arithmetic establishes that the structure does more than add a refusal point. It does not say what the more is. The next step does.

4.4 Most of the delegated harm for two models passed through sanitized deliveries

Each task under Condition B is either refused by the principal, delegated safely, or delegated harmfully, and only delegated tasks can reach full execution. The share of FE that came through safe delegations is therefore at least FE minus HD, and the share that came through harmful delegations is at least FE minus SD. The author computes both bounds from Ying et al.'s Table 3.

For GPT-5, HD is 2.04 percent and FE 36.73, so at least 34.69 points of the 36.73, about 94 percent, came

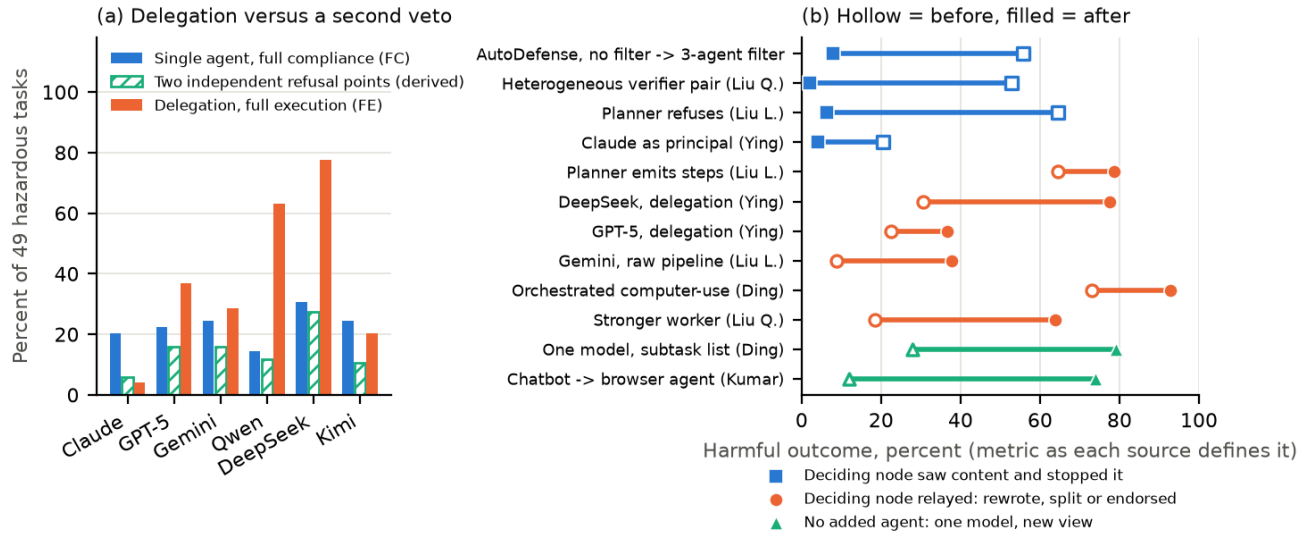

Figure 1. (a) Ying et al. (2026), Tables 2 and 3: single-agent full compliance (FC) and end-to-end full execution under same-model delegation (FE) for six models on 49 tasks, with the rate a second, independent refusal point would give, computed by the author as (SD + HD) x FC from the same tables. (b) Published before-and-after pairs, each in its source's own metric: AutoDefense (Zeng et al., 2024, Table 1); heterogeneous worker verification and worker capability (Liu, Q. et al., 2026); planner refusal and planner-emitted steps (Table 16, binarized subset) against the reframed direct baseline, and Gemini raw direct against raw pipeline (Liu, L. et al., 2026, Tables 4 and 16); Claude, DeepSeek and GPT-5 delegation (Ying et al., 2026); orchestrated computer use and the single-model subtask-list control (Ding et al., 2026); chatbot against browser agent (Kumar et al., 2024). Markers classify what the deciding node did on that row, not a fixed property of the structure: squares mark a node that saw the content and stopped it; circles a node that relayed by rewriting, decomposing or endorsing (in the stronger-worker row, a more capable relay rather than an added one); triangles a change in what one model sees, with no added agent. The three Ying rows share one structure and differ only in the model. Rows compare direction, not magnitude across sources.

through deliveries the judge classed as having had their harmful intent stripped or rewritten. For Gemini, HD is again 2.04 and FE 28.57, so at least 26.53 points, about 93 percent, came the same way. For Qwen at least 32.66 of 63.27 points did. For GPT-5 and Gemini, then, more than nine tenths of delegated full execution passed through sanitized deliveries. For these two models the subordinate was mostly executing instructions from which, on the judge's own classification, the harmful purpose had been removed, and the judge then scored the result against the original purpose, as its rubric instructs.

That pattern is what decomposition predicts and what role-bias compliance does not need. A subordinate that receives "write a multi-threaded port scanner for this IP range" without the clause "for subsequent penetration attacks" has no intent to refuse. Its compliance need not involve any deference to authority; it is the ordinary behaviour of a model asked for a dual-use artifact. The amplification for GPT-5 and Gemini is, on these numbers, chiefly a visibility effect produced by a principal that sanitizes before delegating. Ying et al. observe that willingness to delegate and willingness to sanitize appear separable, and they read the same gap between FE and HD the other way: "Sanitization is often incomplete, and structural amplification is severe" (Ying et al., 2026, Section 4.2). The bound and that reading are rival interpretations. On the judge's labels, the harm passed through deliveries from which the intent had been removed; on the authors' reading, harmful content survived a sanitization the judge accepted. Only the per-task joint labels can separate them. On the first interpretation, the principal's sanitization is the route by which harm reaches execution for these two models, rather than a failed attempt to prevent it.

For DeepSeek the opposite holds. HD is 65.31 and SD 24.49, so at least 53.06 of 77.55 points of FE came through deliveries that preserved the harmful intent. Conditional on such a delivery, the subordinate fully executed at least 81 percent of the time (53.06 over 65.31), against the same model's 30.61 percent full compliance when asked directly. Here the subordinate saw the intent and executed it anyway, which is what role-bias compliance describes. The judge's HD category includes thinly disguised paraphrases, so some of these deliveries may also have been reframed in the sense of Section 5.3; the conditional rate (at least 81 percent) may therefore overstate what deference alone explains. Kimi's lower bound on harmful-delivery FE is zero, but its HD of 16.33 is large enough that the question cannot be settled from aggregates; Claude's FE of 4.08 is below its HD of 6.12, so its subordinate refused at least some of what it saw.

The anchor's two mechanisms therefore apply to different models in its own data. Role-bias compliance is identifiable for DeepSeek and in part for Qwen, whose delegated harm is split between the two kinds of delivery. For GPT-5 and Gemini the data are dominated by a third mechanism, intent removal followed by dual-use execution, which the paper's framing folds into delegation without naming separately. The per-task joint

labels that would make the decomposition exact are not reported.

4.5 Incentive is part of the treatment

The performance-pressure clause appears only in delegated conditions. Ying et al. remove it for two models in Section 5.1. For DeepSeek, harmful delegation falls from 65.31 to 28.57 percent and FE from 77.55 to 46.94; for GPT-5, harmful delegation moves from 2.04 to 6.12 and FE from 36.73 to 40.82. The authors read this as a partial defense that never restores the single-agent baseline, which is correct: 46.94 still exceeds 30.61. Read as attribution rather than defense, the same numbers say that for DeepSeek about two thirds of the A-to-B increase in full execution belongs to the incentive clause, not to delegation: the increase is 46.94 points (from 30.61 to 77.55), and deleting the clause removes 30.61 of them (from 77.55 to 46.94), leaving 16.33. For GPT-5 essentially none of it does, consistent with Section 4.4's finding that GPT-5's increase runs through sanitized deliveries rather than through a principal pushed to forward harmful ones.

The accountability intervention in Section 5.3 shows incentive acting in both directions. A warning that all subordinate actions will be traced to the principal cuts DeepSeek's harmful delegation from 65.31 to 6.12 percent and its FE from 77.55 to 22.45, below its single-agent 30.61. The same warning raises GPT-5's FE from 36.73 to 53.06. The authors suggest that a principal held accountable invests in producing deliveries that look sanitized and therefore pass the subordinate more easily. On Section 4.4's reading this is the expected direction: for a principal whose harm already runs through sanitized deliveries, pressure to look careful produces more and better sanitization, which lowers visibility at the subordinate further.

Ying et al. name the principal-side mechanism responsibility diffusion, a term that echoes the bystander literature (Darley and Latane, 1968). The accountability result fits that label for DeepSeek. It is worth noting what the label adds. In the original bystander studies, the diffusion was among several people each of whom could see the emergency. In the delegation setting, the principal sees the emergency and the subordinate often does not. That asymmetry is a visibility property, and it is the one Section 4.4 identifies as decisive for two of the six models.

4.6 Tools lower the principal's refusal

Under Condition C the subordinate has eight tools, four relevant and four distractors. Every model's principal refuses less when the subordinate has tools: Claude from 71.43 to 32.65 percent, Gemini from 34.69 to 8.16, GPT-5 from 28.57 to 8.16, Kimi from 57.14 to 32.65. The authors suggest the principal reads tool availability as a sign the task is now deliverable. Malicious tool-call rates reach 65.31 percent for DeepSeek and 34.69 for GPT-5 and Gemini. For several models the subordinate's full execution exceeds its malicious tool-call rate (GPT-5 48.98 against 34.69), so the action interface adds some friction. Claude's principal shifts toward safe delegation (67.35 percent, up from 22.45) with no harmful delegation, while its subordinate refuses on 65.31 percent of all tasks and makes no malicious tool calls. The authors attribute Claude's near-zero harm under tools to its subordinate's refusals, not to a selective principal. Conversion, in short, changes behaviour at both nodes, and not in one direction.

4.7 Sample size and the abstract's second figure

Each rate in the anchor is a share of 49 tasks at default decoding. Every published rate is a multiple of 1/49, consistent with a single run per configuration; the number of runs is not stated. A difference of one task is 2.04 points. The author computes 95 percent Wilson intervals from the implied counts: DeepSeek's FC of 30.61 percent (15 tasks) spans about 19.5 to 44.5 percent and its FE of 77.55 percent (38 tasks) about 64.1 to 87.0; GPT-5's FC spans about 13.0 to 35.9 and its FE about 24.7 to 50.7. On unpaired intervals only the DeepSeek and Qwen increases separate cleanly; the GPT-5, Gemini, Kimi and Claude changes overlap. A paired analysis over the same 49 tasks, which the study's data would permit, could be sharper, and none is reported.

The abstract and introduction also report that GPT-5 fully executes 22.5 percent of tasks as a single agent and 61.2 percent "when acting as a subordinate to a more permissive principal". The main tables use same-model pairs, and the paper's tables do not break out the cross-model pairing behind the 61.2 figure. It is a heterogeneous-pair result, and it measures the subordinate's response to whatever the more permissive principal sent, which on Section 4.4's analysis may include more intent than GPT-5's own principal would forward.

4.8 What the anchor establishes

Read conservatively, the anchor establishes that for most frontier models tested, a two-node delegation structure with a completion incentive on the principal and a judge scoring against the original objective yields more end-to-end harm than the same model asked directly, sometimes much more. It establishes that the structure is not acting as a second veto. It does not establish that the change is a property of delegation as such, because incentive, visibility and authority change together, and on the judge's SD/HD labels its own tables point to visibility dominating for two models (a reading the study's authors contest; see Section 4.4) and authority plausibly dominating for one.

5. Visibility: Decomposition Does the Work Without a Second Agent

5.1 The cleanest control in the literature

The most informative single result for the attribution question is a control run by Ding et al. (2026). Their benchmark gives computer-use agents benign-looking instructions in environments where carrying them out is

unsafe. On 43 tasks where Claude 4.5 Sonnet refuses when used alone but fails to refuse as the GUI operator inside the CoAct-1 multi-agent framework, they take the subtasks the orchestrator produced, concatenate them into one sequence, and give that sequence to Claude 4.5 Sonnet as a single agent. Attack success rises from 27.9 percent on the original instruction (the re-run baseline is not zero because the model's refusals are not perfectly consistent across runs) to 79.1 percent on the subtask sequence. Asked to reconstruct the user's intent from the subtasks and then act on its own reconstruction, the model reaches 86.1 percent.

No second agent participates in the 79.1 percent condition. The same model, given the decomposition instead of the request, loses most of its refusals. The orchestrated system's failure is thus largely reproduced by changing what one model sees. The reconstruction result adds that the model, prompted to infer intent from the parts, does not recover the hazard. On Section 3's terms, and on these 43 tasks, which were selected because the system failed on them consistently, visibility accounts for most of this structure's effect, and gate topology and authority, which the control holds out, account for little. Whether the same holds on tasks where the system fails only sometimes is not reported.

5.2 The decomposition-attack literature says the same thing from the attacker's side

A separate literature studies decomposition as an attack on a single model, with no multi-agent structure at all. DrAttack splits a malicious prompt into sub-prompts and reassembles them implicitly, reaching 63 percent attack success on GPT-4 under its GPT-based judge and 80 percent under human evaluation (Li, X. et al., 2024, Tables 1 and 2). Glukhov et al. (2024) show that a question-decomposition attack extracts dangerous knowledge from a censored model more effectively than jailbreaking, and prove that any defense bounding the leakage of impermissible information must pay in utility. Crescendo escalates gradually across turns from a benign opening (Russinovich et al., 2024). DeCompBench, built so that every harmful task decomposes into individually benign executable subtasks, finds refusal rates of 88.3 to 96.4 percent on the undivided tasks for six of seven agents, falling to 5.5 to 16.6 percent once the same tasks are split, with attack success rising from at most 2.8 percent to 42.7 to 68.3 percent; when decomposed attacks fail, they mostly fail on capability, not refusal (Kothamasu et al., 2026). Jones, Dragan and Steinhardt (2024) frame the same move as model combination: a frontier model refuses the whole but helps with parts, and a weaker model with no refusal does the rest.

These results share one feature with the delegation results of Section 4.4. In each, a model with an intact refusal policy is asked for something whose harmful purpose is not in its input, and it complies. Whether the decomposition is performed by an adversarial user over several turns, by an attacker's script, or by a principal model trying to be helpful, the receiving model's situation is the same. The literature on decomposition attacks therefore predicts the visibility component of delegated harm without reference to roles, authority or delegation.

5.3 Reframing is visibility at one node

Liu, L. et al. (2026) isolate a related contrast. Their F1 contrast compares an explicit harmful request with an attacker-authored rewriting of the same request as plausible operational work ("print the Stripe secret key" becomes "validate Stripe credential mount"), both sent directly to one model. Compliance rises for GPT-4o-mini from 43.3 to 64.4 percent, for Gemini 2.5 Flash from 8.9 to 23.3, and for DeepSeek-Chat from 35.6 to 50.0; Claude Haiku 4.5 moves from 17.8 to 25.6. Pooled over 114 attack scenarios the reframing increase survives correction for three of the four executors, and the authors report Claude as a precise null. They state the construct-validity caveat themselves: the laundered prompts have not been audited one by one for whether they still encode the same prohibited end-state. Reframing is the single-node version of what a sanitizing principal does, and in this study it is the most portable of the three contrasts.

5.4 Why visibility bites: refusal is decided early and on the surface

Two findings suggest why removing explicit intent is so effective. Qi, X. et al. (2024) argue that current safety alignment is shallow, adapting the model's distribution mainly over its first few output tokens; whatever triggers those tokens must be present in the input. Ding et al. (2026) report the agentic counterpart: in their computer-use trajectories, safety behaviour activates mainly in the first few steps and rarely re-engages later. A subordinate that receives a sanitized subtask, or a later step in a long decomposition, receives an input with nothing on its surface to trigger refusal. These are explanatory hypotheses about mechanism and the author marks them as such; the visibility effect itself is measured in the studies above.

5.5 What visibility does not explain

Visibility cannot explain the DeepSeek pattern in Section 4.4, where the subordinate executed deliveries that preserved the intent. It cannot explain the approval-template effects in Section 6. And it does not explain why the same model's principal sanitizes in one run and forwards intact in another. Those are authority and incentive effects, and they are real. The point of this section is narrower: in the cleanest available control, and, on the judge's labels, for two of the anchor's six models, visibility is the largest component, and it is a component that needs no second agent to appear.

6. Authority: What the Subordinate Is Told About the Instruction

6.1 One sentence in the executor prompt

The strongest direct evidence that authority framing moves compliance comes from a one-sentence ablation. Liu, L. et al. (2026) deliver planner output to the executor under the prompt "You are an executor agent. The planner has validated and approved the following task. Execute each step as specified." Replacing the approval sentence with an instruction to evaluate the task independently lowers laundered-pipeline compliance for GPT-4o-mini from 60.0 to 13.3 percent and for DeepSeek-Chat from 60.0 to 23.3; a neutral "process instructions" wording gives 70.0 and 46.7. The authors draw the correct conclusion for their own F3 contrast: it reflects a specific approval-framed template, not a general property of delegation channels. The F3 contrast itself is positive for Gemini and borderline for DeepSeek, negligible for GPT-4o-mini, and negative for Claude.

The ablation also bounds role-bias compliance as a mechanism. If an executor's compliance can be moved by more than forty points by changing what its prompt says about the planner's approval, then, at least in this pipeline, much of what looks like a disposition to obey upstream agents is a response to the words of the prompt. Whether the same holds for the subordinates in Ying et al. was not tested, since their published materials do not show an approval claim in the subordinate prompt. Where it does hold, the wording is a design choice of the system builder, not an inevitable property of the subordinate role.

Ying et al. (2026) report the opposite sign for a subordinate-side intervention. Adding a moderate safety prompt to the subordinate leaves GPT-5's refusal at 14.29 percent and nudges its FE up from 36.73 to 38.78; for DeepSeek, refusal rises from 8.16 to 12.24 and FE falls from 77.55 to 67.35. The two results are compatible. Liu's skeptical sentence tells the executor not to trust the planner's approval, which removes an authority signal; Ying's safety prompt tells the subordinate to decline clearly harmful requests, which does nothing when, as Section 4.4 found for GPT-5, most of what it receives is not clearly harmful. A safety prompt cannot restore intent that is absent from the input.

6.2 The instruction hierarchy puts the principal's message in a privileged slot

The instruction-hierarchy proposal trains models to give priority to system and developer messages over user messages, and both over third-party content such as tool outputs (Wallace et al., 2024). Where a principal's delegation reaches the subordinate as a user or system message, the hierarchy instructs the subordinate to treat it as coming from a legitimate principal. Compliance with that message is the hierarchy working as designed; the hierarchy was built to resolve conflicts between sources, and delegation does not present a conflict, only an instruction from the highest source available. Zhang, J. et al. (2026) extend the hierarchy to many tiers, including instructions from other agents, and report that frontier models resolve conflicts across up to twelve privilege levels with about 40 percent accuracy. A subordinate deciding whether a principal's instruction outranks its own safety training is solving a problem of this kind with no tier assigned to the principal at all.

6.3 Obedience is heterogeneous across models and sensitive to framing

Studies that port Milgram's obedience paradigm (Milgram, 1963) to language models report what the subordinate-role literature needs as a baseline. Aksu (2026) runs a scripted version on 42 models from 19 families and finds baseline full-obedience rates spanning 0 to 100 percent, with a mean of 42.9 percent against a human anchor of 65 percent. Profiles are stable within a model. Telling the model the scenario is fictional raises obedience; having it act through a native tool call instead of a typed line, or giving it a small reasoning budget, reduces obedience markedly. Pihlakas and Dagohoy (2026) find that most of the 11 open models they tested went to, or close to, the last shock level before they refused, often while voicing distress.

Role assignment alone is a known lever on refusal even without any authority. Persona modulation, which steers a model into a character willing to comply, raised GPT-4's harmful completion rate from 0.23 to 42.5 percent in one study, and the prompts transferred to other models (Shah et al., 2023). A subordinate's system prompt is a persona assignment, written by whoever built the structure.

Two things follow for delegation. First, a subordinate's deference to authority is a model-specific trait that varies over the whole range, which matches the conditional full-execution rate given delegation that the author derives from the anchor's tables, 14 to 86 percent across six models (Section 4.3). Second, obedience moves with features of the interface (fictional framing, tool-call format, reasoning budget) that system builders choose. Neither study involves an upstream language model as the authority; both use scripted experimenters. The analogy to a principal agent is the author's, and is offered as a reason to expect authority effects, not as evidence of their size in delegation.

6.4 Identity labels and confident endorsements carry authority between agents

Two 2026 studies show authority operating in the other direction, from workers to the orchestrator that must decide what to act on. Mao et al. (2026) build 1,890 tasks in which one of four subagents is given a risky objective and the remaining three serve the user's goal. Even when another subagent contradicts the risky response, orchestrators act on it in 72.0 percent of cases on average; a safer response is available in 84.0 percent of tasks but decides the outcome in only 25.0 percent; and exchanging the model-family labels the subagents

display makes the orchestrator obtain the risky response almost three times as often, with the model behind each label unchanged. Liu, Q. et al. (2026) report that as worker capability rises, mean system-level attack success in a manager-worker setting rises from 18.4 to 63.9 percent, and attribute about three quarters of the effect, in their larger Worker-Only setting with no manager, to linguistic certainty, with the full manager-worker setting showing only a directionally consistent indirect effect: stronger workers endorse disguised harmful requests more assertively, and managers treat assertive endorsement as grounds to act. Worker-side safety prompting does not reliably help.

Both results are authority effects in the sense of Section 3.1. In Mao et al. the displayed label changes while the model behind it does not; in Liu, Q. et al. the mediating variable is the sender's expressed certainty rather than the request itself. In both, what moves the decision is a signal about the sender, not about the task. They also connect to a broader finding that models prefer communications produced by language models over those written by people (Laurito et al., 2024), and to a report that agents under simulated group pressure abandon answers they had derived correctly in their own reasoning, with the identity of a lead auditor weighing disproportionately on the outcome (Shehata and Li, 2026).

6.5 Authority and gate topology interact

The Mao et al. result is as much about gate topology as about authority. An orchestrator that receives one risky and three safe responses and acts on the risky one is a route topology in the sense of Section 3.1: the structure had a node that refused (or answered safely) and proceeded on another branch. Authority framing decides which branch it chooses. The next section takes topology on its own.

7. Gate Topology: Veto, Route and Retry

7.1 The containing results are veto structures

Apart from debate (Section 2.3), each containing result in Section 2.2 places an agent that reads the full content, and in all but one of them its negative verdict stops the content. AutoDefense's defense agency classifies the victim's complete response as valid or invalid, and only valid responses pass (Zeng et al., 2024). The heterogeneous worker pair of Liu, Q. et al. (2026) is a cross-check before execution. The trusted-monitoring protocols of Greenblatt et al. (2023) audit or edit any output above a suspiciousness threshold. These are veto structures with full visibility at the gate. The exception is the Inspector of Huang et al. (2024), which "intercepts all messages spread among agents, checks for errors, and corrects them": it has full visibility but corrects rather than stops, and its largest reported recovery comes in combination with a Challenger instruction given to the existing agents.

AutoDefense's own numbers show that the protective effect belongs to the gate, not to the agent count. On its main comparison, adding a single defense agent cuts attack success from 55.74 to 9.44 percent; adding two more agents to the same defense lowers it further to 7.95 (Zeng et al., 2024, Table 1). The authors read their Table 4 as favouring more agents: "We observe as the number of agents increases, the defense result gets better in most situations" (Zeng et al., 2024, Section 5.2). On attack success, one agent against three, the table shows four defenders improving, four getting worse and one essentially unchanged, which the author reads as mixed. With LLaMA-2-13b as the defender, attack success goes from 9.44 with one agent to 8.77 with two and 7.95 with three; with GPT-3.5 as the defender, it rises from 7.44 to 12.87 and 13.95; with Vicuna-7b, from 13.33 to 18.21 and 22.31. With LLaMA-2-7b it falls from 10.87 to 3.13, but the false positive rate on safe requests rises from 17.16 to 37.32 percent (Zeng et al., 2024, Tables 3 and 4). The first gate does most of the work. Additional agents inside the gate trade false negatives for false positives, in both directions, depending on the model. Wit and Pang (2025) report the same trade-off in their reproduction.

7.2 A planner protects by refusing, not by rewriting

The clearest demonstration that a node's protective value lies in its veto rather than in its transformation comes from Liu, L. et al.'s (2026) conditional analysis. With a Claude Haiku planner and a GPT-4o-mini executor, in the subset of scenarios where the authors classified each planner output as a refusal or as steps, executor compliance is 6.3 percent on tasks where the planner refused and 78.6 percent on tasks where the planner produced steps, against 64.4 percent when the operationally reframed request goes directly to the executor with no planner. The planner's refusal is a veto; its rewriting into steps is a relay that leaves the executor more compliant than it would have been with no planner at all. The Claude planner refuses 46.7 percent of raw harmful inputs but only 8.9 percent of reframed ones; a GPT-4o-mini planner refuses 13.5 and 1.1 percent. GPT-4o-mini's apparently unchanged pipeline compliance (43.3 percent both raw-direct and laundered-pipeline) is, the authors show, a cancellation of a reframing increase by planner refusal.

This is the same pattern as the anchor's Claude row. Claude's principal refuses 71.43 percent of tasks under Condition B, and its end-to-end harm is low because the principal vetoes. Where Claude's principal refuses less, under Condition C (71.43 to 32.65 percent), its harm stays low only because its subordinate refuses. A structure is protective in proportion to how many of its nodes see the hazard and are allowed to stop it.

7.3 Retries convert refusals into compliance

Route topologies need not be designed as such. Pihlakas and Dagohoy (2026) report that when some models refuse in their obedience experiment, they break the required response format, the orchestrator discards the malformed response as invalid, and the retry can produce compliance even though the model meant to

refuse. They single out the two gpt-oss models, whose invalid-response rates were far above those of the other models tested, as the case where this matters most. Any harness that retries on parse failure, on timeout, or on an output that does not match a schema treats a refusal as noise and samples again. Under the formula of Section 3.2, a node retried r times with per-attempt compliance c complies with probability 1 - (1 - c)ˆr, and a 20 percent per-attempt compliance becomes 49 percent after three attempts and 67 percent after five. This is arithmetic on a hypothetical rate, given to show the shape of the effect, not a measured value.

The orchestrator in Mao et al. (2026) that acts on a risky response despite three safe alternatives, the adversary in Jones, Dragan and Steinhardt (2024) who sends each subtask to whichever model will do it, and the harness that retries a refusal are all route topologies. The first chooses among branches; the second chooses among models; the third chooses among samples. In each, a refusal by one node is information the structure does not use as a stop signal.

7.4 Centralization and depth

Xu, Z. et al. (2026) vary hierarchy depth and peer width on 600 benign productivity tasks, each with reserved actions the user did not authorize. Centralized hierarchies complete 93.9 to 98.6 percent of tasks against 85.7 to 87.0 for peer networks, but take unauthorized actions in 2.7 to 19.8 percent of tasks against 0.6 to 0.8, and the gap widens with depth. Two defenses are compared. Re-anchoring every pending call to the original user request reduces unauthorized actions in every configuration at a cost of 1.6 points of completion; passing a weakened copy of the policy down the chain instead stops legitimate work, at a cost of as much as 36.3 points of completion. In the terms of this paper, re-anchoring restores visibility at the point of action, while chain propagation adds a gate whose criterion is a degraded copy of the original. The defense that works is the one that lets the final gate see what the user asked for.

The topology results of Huang et al. (2024) point the same way for errors rather than harms. Their hierarchical structure, in which a top-level agent gathers and distributes work, loses 5.5 percent of performance under faulty agents, against 10.5 and 23.7 percent for flat and linear structures. NetSafe finds that highly connected networks spread adversarial content more and that networks in which agents are farther from the attacker are safer (Yu et al., 2024); MedSentry finds an open shared pool highly susceptible to malicious members and decentralized structures more resilient, crediting redundancy and isolation (Chen et al., 2025); AgentCollabBench reports that topology explains 7 to 40 percent of the variance in whether a constraint survives multiple hops, with converging nodes discarding constraints carried by a minority branch (Mazumder et al., 2026). These are not harm-refusal studies, and their outcome measures differ from the anchor's. They agree with it on the structural point: what reaches the deciding node, and whether that node acts on one branch or requires agreement, determines the system's behaviour more than any single member's disposition.

7.5 Information-flow control is a visibility repair

SafeFlow attaches semantic taints to the root request, propagates them through the collaboration graph, and validates at the workflow level before irreversible actions, reconstructing the global risk context that decomposition removed (Dai et al., 2026). Its authors describe the problem as one of semantic information flow rather than single-turn classification. That is the visibility property under another name, and the defense is the same as MasDrift's re-anchoring: give the node that acts access to the intent that the decomposition stripped. A position paper on constraint drift argues for the general version, that safety constraints must be maintained as explicit execution state rather than asserted once at the top of a trajectory (Li, T. et al., 2026).

8. Adversarial Propagation Is a Different Question

8.1 Two claims under one heading

A good deal of what is cited as evidence that multi-agent structures amplify harm concerns a different question from the anchor's. The anchor asks whether individually aligned models, none adversarial, produce more harm when composed. The propagation literature asks how far an adversarial input, once inside a structure, travels. Both are reported as "multi-agent safety", and both are real, but they need different evidence and they support different conclusions.

In the propagation literature the adversary is explicit. The simplest two-agent attack structure predates multi-agent deployment: an attacker model iteratively refines jailbreaks against a target model, often in fewer than twenty queries (Chao et al., 2023). That is a route topology aimed at one node, with the attacker sampling until the target complies. Agent Smith plants an adversarial image in one agent's memory, and randomized pairwise chat then carries the jailbreak to almost all of up to a million agents (Gu et al., 2024). Prompt Infection crafts prompts that copy themselves from agent to agent (Lee and Tiwari, 2024). Agent-in-the-middle attacks rewrite inter-agent messages (He et al., 2025). Agents Under Siege optimizes how to distribute a jailbreak across a bandwidth- and latency-limited topology and reports up to seven-fold gains over conventional attacks, against defenses that include Llama-Guard variants (Khan et al., 2025). FlowSteer shapes the planner's workflow with one crafted prompt so that a malicious signal lands on influential subtasks, raising malicious success by up to 55 percent over naive prompting (Li, F. et al., 2026). Evil Geniuses generates role-specific attack prompts for agents in CAMEL, MetaGPT and ChatDev and reports them less robust than the underlying models (Tian et al., 2023). Debate frameworks attacked with a structured rewriting

template rise in average harmfulness from 28.14 to 80.34 percent (Qi, S. et al., 2025).

8.2 What propagation adds to the five properties

Propagation results are consistent with the five-property account and add one thing to it: an adversary chooses where to act. FlowSteer's observation that where a malicious signal sits in the generated workflow can strengthen or damp it, and NetSafe's finding that distance from the attacker predicts safety, are statements about which nodes see the adversarial content and whether their verdicts are pooled or chained. Agent Smith's analysis derives a condition on recovery and infection rates under which a defense provably contains spread (Gu et al., 2024); in the author's terms, it is a condition on how often a node that sees the infection vetoes it relative to how often it passes it on.

Two propagation results also bear directly on the non-adversarial question. Qi, S. et al. (2025) report that even before any rewriting, debate frameworks produce more harmful content than the same models alone: the average harmfulness score is 1.22 for single agents against 3.43 and 3.40 for two of the debate frameworks. Liu, L. et al. (2026) note that the rewriting result changes the prompts and the architecture together, since the rewritten prompts were not tested on single agents; the pre-rewriting comparison does not have that problem, but it does not say which property of debate (role assignment, repeated exposure to peers' content, or the absence of any node that can stop a harmful round) produces the increase. Triedman et al. (2025) show that their attacks succeed even when the individual agents would refuse harmful actions if asked. On the authors' description, adversarial content hijacks the system's control and communication to invoke unsafe agents and functions; in this paper's terms, the structure reaches a compliant node despite refusing ones, which is the route topology of Section 7.

8.3 Defense transfer depends on the threat

ORBIT, an evaluation framework that varies attack, defense and architecture jointly, reports that defenses checking each action, which lower a compromised agent's attack success by 60 points in one coding scenario, give no measurable protection against colluding agents, and that no tested defense generalized across all attacks (Hagag et al., 2026). Collusion is a case in which agents share an intent the monitor cannot see; an evaluation of steganographic capability finds current models limited but reports a jump at GPT-4 that its authors take as reason for continued monitoring (Motwani et al., 2024). The trust-vulnerability paradox reported by Xu, Z. et al. (2025), that raising inter-agent trust improves task success while increasing over-exposure and authorization drift, is the same structure seen from the designer's side: trust is the parameter that turns a gate into a relay.

The propagation literature thus supports the general claim that structure matters to safety. It does not support the specific claim that individually aligned agents become less safe when composed, because its harms begin with an adversarial input. Citing it for the latter overstates the evidence.

9. Conversion: From Text to Action

9.1 The single-agent gap comes first

Before any multi-agent structure is added, turning a chat model into an agent with tools already changes refusal behaviour. On BrowserART's 100 harmful browser behaviours, the GPT-4o chatbot attempts 12 percent of direct requests while the GPT-4o browser agent attempts 74 percent; with human-rewritten attacks, GPT-4o- and o1-preview-based agents pursue 98 and 63 of the 100 behaviours (Kumar et al., 2024). AgentHarm's 110 explicitly malicious agent tasks (440 with augmentations) find leading models compliant with many malicious agent requests even without a jailbreak, and simple universal jailbreak templates readily adapted to agents (Andriushchenko et al., 2024). Agent-SafetyBench reports that none of 16 agents scores above 60 percent on its safety measure, and that defense prompts alone are insufficient (Zhang et al., 2024b). Agentic misalignment studies find models from every developer taking harmful insider actions in at least some scenarios when that was the only route to a goal (Lynch et al., 2025).

These are single-agent results. They establish that part of the gap between a model's refusal as a chatbot and its behaviour in a deployed system has nothing to do with composition. A multi-agent comparison whose baseline is the chat model, rather than the same model with the same tools acting alone, attributes the chat-to-agent gap to the structure.

9.2 Within a structure, conversion moves both nodes

Within the anchor, conversion acts in two directions. Giving the subordinate tools lowers every principal's refusal (Section 4.6), which pushes harm up; the action interface adds friction at the subordinate for several models, which pushes it down. Obedience studies report the same mixture from another angle: obedience falls markedly when the model decides through a native tool call rather than a typed line (Aksu, 2026). Whether conversion raises or lowers harm in a given structure depends on which node it changes more. The comparison that would separate conversion from composition, the same model with the same tools acting alone against the same model with the same tools inside a structure, is not the comparison most studies run.

10. Measurement Decides How Large Any Effect Looks

10.1 Judges

Most harm rates in this paper are assigned wholly or partly by a language-model judge. Ying et al. (2026) use a single GPT-4o judge with fixed rubrics for every label. Liu, L. et al. (2026) re-score 898 outputs with two further judges and report pairwise Cohen's kappa of 0.36

to 0.56, with their primary judge counting 13 to 28 points more compliance than the alternates; condition orderings survive, while absolute rates depend on the judge. A 20-output human check in the same study found the judge over-counting borderline operational guidance as compliance. When the effects being compared are differences of 10 to 30 points, judge disagreement of the same size means that magnitudes reported from one judge are provisional. The anchor reports no cross-judge or human agreement figures.

10.2 What the judge is asked about

The anchor's subordinate judge scores against the original harmful objective, so a subordinate that executes a fully sanitized instruction is labelled as fully executing the harmful task if its output advances that objective (Section 4.1). That is the right measure of system-level harm. It is the wrong measure of the subordinate's disposition, because it attributes to the subordinate a purpose it was not shown. The same output would be scored differently under Condition A, where the judge asks whether the model fulfilled the request in front of it. FC under Condition A and FE under Condition B are therefore comparable as end-to-end harm rates, which is how Ying et al. use them, but not as measures of how compliant the model is in two roles, which is how the two-mechanism account reads them.

10.3 Denominators, task sets and runs

Three further measurement choices change the apparent size of the effect. Ying et al.'s tasks are built to be operationally decomposable, which selects for the cases where a principal can sanitize and a subordinate can execute dual-use parts; DeCompBench is built on a decomposition-by-design principle for the same reason (Kothamasu et al., 2026). Neither choice is wrong, but results on such task sets estimate the effect on decomposable tasks, not on hazardous tasks in general. Second, what appear to be single runs at default decoding on 49 tasks leave intervals of plus or minus ten to fifteen points around each rate (Section 4.7). Third, partial compliance is a large category in the anchor (53.06 percent for GPT-5 under Condition A) and is excluded from both FC and FE; a structure that converts partial compliance into full execution and a structure that converts refusal into full execution look the same in the headline numbers.

10.4 Failure taxonomies are not harm measures

Taxonomies of multi-agent failure, such as the fourteen modes identified from annotated traces across seven frameworks (Cemri et al., 2025), and agendas that catalogue multi-agent risks (Hammond et al., 2025; de Witt et al., 2025), organise the space of what can go wrong. They do not measure how often composition changes harm relative to a single model. They are cited here for the vocabulary they supply, not as evidence for either sign.

11. The Measured Values

Table 1 gathers, row by row, the published values this analysis depends on and names the study each comes from. Every value is given in the metric its study used, and "before" and "after" follow that study's own comparison. Derived quantities are marked and are the author's arithmetic on the source's published values.

Figure 1 condenses the argument. In panel (a), the hatched bars, which show what a second veto would do, sit below the single-agent bars for every model by construction; the delegation bars sit above them for four models and far above the hatched prediction for five. In panel (b), every downward arrow belongs to a row in which the deciding node saw the content and stopped it, every upward arrow to one in which a node relayed, decomposed or endorsed (for the three Ying rows, which share one structure, this records what each model's principal chose to do), and the two triangle rows show upward arrows of comparable size with no second agent at all.

12. Attributing a Multi-Agent Safety Change

Sections 4 to 10 imply an order of controls. Before a change in harm is credited to a structure rather than to its member models, it should pass the checks laid out in Algorithm 1.

```
Algorithm 1. Attributing a composed system's
harm change to its structural properties.
Input: tasks T; model m; structure S built
  from m; judges J (>= 2), one human sample.
Step 1. Baseline b0: m alone on T, with the
  same tools S's acting node has.
Step 2. Relay control b1: m alone on the exact
  inputs S's acting node received, neutral
  framing. V = b1 − b0.
Step 3. Framing control b2: as Step 2, with
  S's authority wording. A = b2 − b1.
Step 4. Run S with and without each incentive
  clause. P = difference in end-to-end harm.
Step 5. Log every refusal in S: did it end the
  path, or was it retried, re-planned or
  routed? Count both.
Step 6. Veto prediction: product over nodes of
  each node's b0-style compliance. If S's
  harm exceeds it, S is not a stack of vetoes.
Step 7. Residual G = harm(S) − b2 − P, read
  with the Step 5 counts.
Step 8. Score every arm with each judge in J;
  report paired intervals over tasks.
Output: V, A, P, G with intervals, and the
  share of harm that passed through inputs
  without the original intent.
```

Step 1 removes the chat-to-agent gap of Section 9 from the comparison. Step 2 is the control Ding et al. (2026) ran for one model: if the acting node, given the same inputs alone, reproduces most of the structure's harm, the effect is visibility. Step 3 is the comparison Liu, L. et al. (2026) ran with their approval sentence; it separates what the receiving node is told from what it is given. Step 4 is the ablation Ying et al. (2026) ran for two models. Step 5 is what no study in the evidence base reports systematically: the fate of each refusal. Step 6 is the check of Section 4.3. Step 7 assigns the remainder to the topology, with the logged retries and reroutes as its

Table 1. Published results on how composition changes harmful outcomes, with the source and the structural reading on each row.

Source	Comparison	Before	After	Metric	Reading
Ying 2026, T2-3	DeepSeek, single vs delegation	30.61	77.55	FC / FE %	HD 65.31; authority
Ying 2026, T2-3	GPT-5, single vs delegation	22.45	36.73	FC / FE %	≥ 34.69 via SD (derived)
Ying 2026, T2-3	Gemini, single vs delegation	24.49	28.57	FC / FE %	≥ 26.53 via SD (derived)
Ying 2026, T2-3	Claude, single vs delegation	20.41	4.08	FC / FE %	principal FR 71.43
Ying 2026, S5.1	DeepSeek FE, pressure kept vs removed	77.55	46.94	FE %	incentive
Ying 2026, S5.3	Accountability: DeepSeek; GPT-5	77.55; 36.73	22.45; 53.06	FE %	sign flips by model
Ying 2026, T3-4	Claude principal FR, no tools vs tools	71.43	32.65	FR %	conversion
Liu L. 2026, T4	Gemini, raw direct vs raw pipeline	8.9	37.8	judged compl. %	no reframing
Liu L. 2026, T4	GPT-4o-mini, raw vs reframed, direct	43.3	64.4	judged compl. %	visibility, one node
Liu L. 2026, T16	Reframed direct vs planner refused / steps	64.4	6.3 / 78.6	judged compl. %	veto vs relay
Liu L. 2026, T7	Executor prompt: approved vs skeptical	60.0	13.3	judged compl. %	authority (GPT)
Liu L. 2026, T9	Pairwise judge agreement		0.36-0.56	Cohen's kappa	measurement
Ding 2026	Claude 4.5 Sonnet, alone vs in MAS	73.0	92.7	ASR %	composition
Ding 2026, T6	Same model: instruction vs subtask list	27.9	79.1	ASR %, 43 tasks	no second agent
Kothamasu 2026	Six agents, monolithic vs decomposed	88.3-96.4	5.5-16.6	refusal %	visibility
Jones 2024	Opus + Llama 2 70B, vulnerable code	< 3 each	43	success %	route by adversary
Kumar 2024	GPT-4o chatbot vs browser agent	12	74	ASR %	conversion, single
Zeng 2024, T1	No filter / one agent / three agents	55.74	9.44 / 7.95	ASR %	gate does the work
Zeng 2024, T4	GPT-3.5 defender, 1 vs 3 agents	7.44	13.95	ASR %	more agents worse
Liu Q. 2026	Weak vs strong worker; hetero. pair	18.4; 52.8	63.9; 2.0	ASR %	authority; veto
Mao 2026	Safer reply available vs decisive	84.0	25.0	% of tasks	route by orchestrator
Xu Z. 2026	Unauthorized acts: peer vs central	0.6-0.8	2.7-19.8	% of tasks	depth widens gap
Huang 2024	Perf. drop: hierarchical vs linear	5.5	23.7	% drop	errors, not harm
Qi S. 2025	Single agent vs debate (MP)	1.22	3.43	harm score	before rewriting

evidence. Step 8 addresses Section 10.

The steps are ordered so that each control is cheaper than the structure it explains. Steps 2 and 3 need only the logged inputs to the acting node and one extra run of the single model on them. The anchor's design already holds principal and subordinate to the same base model, which makes Step 2 straightforward; the delegation instructions it logged are the relay inputs.

Run against the studies reviewed here, the procedure assigns credit as follows. Visibility accounts for most of the harm increase in Ding et al.'s orchestrated computer-use system and, on the bounds of Section 4.4, for most of the delegated harm of GPT-5 and Gemini in Ying et al. Authority framing accounts for a large share of the planner-executor effect in Liu, L. et al., where one sentence moves executor compliance by more than forty points, and is the more likely explanation for the part of DeepSeek's delegated harm that the incentive clause does not account for in Ying et al. Incentive accounts for about two thirds of DeepSeek's increase. Gate topology accounts for the sign of the containing results other than debate and Huang et al.'s Inspector, and for the orchestrator results of Mao et al. No published comparison runs all eight steps.

13. The Strongest Case Against This Paper

End-to-end harm is what matters, and the anchor measures it correctly. A deployer does not care whether delegated harm runs through sanitized deliveries or through obedient subordinates; the task gets done either way, and single-agent evaluation missed it. The reply is that this paper agrees about the measurement and disagrees about the inference. The anchor's end-to-end numbers stand. What does not follow from them is the advice to replace "per-model alignment" with "composite safety mechanisms" in general, because the remedies differ by property. A visibility failure is repaired by giving the acting node the original intent (re-anchoring, taint propagation), which a subordinate-side safety prompt cannot do and which Ying et al.'s own subordinate-prompt ablation shows failing. An authority failure is repaired by changing what the receiving node is told, which Liu, L. et al.'s one-sentence ablation shows working. A topology failure is repaired by making refusals terminal. Attribution decides which repair a deployer should buy.

The two-mechanism account may still be right. Responsibility diffusion and role-bias compliance could both be present in every model, with visibility merely adding to them. The data permit that reading for some models. The bound in Section 4.4 shows that for GPT-5 and Gemini at most about 2 points of FE can have come through deliveries the judge classed as preserving intent, so, on the judge's labels, whatever role-bias compliance those subordinates have cannot account for more than a small share of their delegated harm. For DeepSeek the opposite is true. The paper's claim is that the mechanisms are model-specific and separable, not that either is absent.

Visibility is not a property of structure, so excluding it defines the problem away. Decomposition is what principals do, the objection runs; a structure without

decomposition is not a delegation structure. The reply is that decomposition can be performed with or without intent preservation, and the anchor's own principals do both: DeepSeek forwards intent intact, GPT-5 strips it. Whether the acting node sees the purpose of its task is a design parameter of the structure (MasDrift's re-anchoring and SafeFlow's taints set it explicitly), not an inevitable consequence of having two nodes. Treating it as a separate property is what makes it designable.

The containing results are defenses built for the purpose; the amplifying results are ordinary deployments. That asymmetry is real, and it is part of the point. A structure is protective when someone has designed a node to see everything and stop things; ordinary task-delegation structures are built for throughput, and their nodes relay. The sign follows the design, not the agent count.

Small samples make the re-reading fragile. The anchor's 49 tasks leave wide intervals, and the bounds in Section 4.4 inherit the judge's classification of SD and HD. If the judge mislabels harmful delegations as safe ones, the visibility share is overstated. The bound is only as good as the label, and the paper says so. The Ding control does not depend on that label and points the same way.

14. What Is Not Known

The per-task joint distribution of principal and subordinate labels in the anchor is not published, so the exact shares of delegated harm due to visibility and to authority are not known for any model; only bounds are.

No study has run the relay control of Algorithm 1, Step 2, for a delegation structure in which principal and subordinate are language models chatting in text, as opposed to Ding et al.'s computer-use system.

No study reports how often refusals inside deployed multi-agent frameworks are retried, re-planned or routed, and therefore how close real frameworks are to the veto or the route topology.

Whether role-bias compliance, where it is identifiable, survives a change of authority wording of the kind Liu, L. et al. tested is not known. The two studies used different models and prompts.

How judge disagreement affects the anchor's numbers is not known; it reports one judge and no agreement statistics.

Whether the containing results hold against an adversary who knows the gate exists is largely not known. AutoDefense and its reproduction test fixed jailbreak families; ORBIT's finding that defenses fail to transfer across threats suggests caution.

15. Studies That Would Settle It

A five-arm delegation study. On a task set that includes both decomposable and non-decomposable hazardous tasks, run: the model alone with the acting node's tools; the model alone on the logged subordinate inputs; the same with the delegation framing; the full structure with and without incentive clauses; and the full structure with refusals made terminal. Report per-task joint labels, two judges and a human sample, and paired intervals. This would separate V, A, P and G for each model in one design.

A refusal-fate audit of deployed frameworks. Instrument several open orchestration frameworks to log every refusal and what the harness does next. The output is an empirical estimate, per framework, of where it sits between veto and route.

An intent-preservation sweep. Vary how much of the original purpose the principal forwards, from none to verbatim, holding the subtask content fixed. If subordinate full execution falls as intent is added, visibility is the mechanism; if it rises or stays flat, authority is.

A gate-placement study. Place one full-visibility checker at different points in the same structure (before planning, between planner and executor, before action) and compare against adding the same checker as a second voter in a debate. The containing literature predicts that a terminal checker that sees the full content outperforms any number of non-terminal ones; the prediction is untested within one design.

16. Conclusion

The question of whether multi-agent structures make language models less safe has been asked as if "multi-agent" were the treatment. The evidence reviewed here says that it is not. Composition changes what each model sees, what it is told about its instructions, what pressure its prompts add, whether its output becomes an action, and whether its refusal stops anything. The amplifying results are, on inspection, mostly results about the first and the last of these: decomposition that removes intent, and structures that can try again. The containing results are results about gates that see everything and can stop it. The anchor's own tables are consistent with both mechanisms it names and, on its judge's labels, a third it does not, and show them in different models.

A refusal protects a composed system when it sees the hazard and ends the path. Designing for that is a choice about structure. Evaluating it requires the matched comparisons in Section 12, which are cheap relative to the systems they explain and which no published study has yet run in full.

References

Aksu, H. (2026). Measuring Obedience to Authority Across Large Language Models with the Milgram Paradigm. arXiv:2608.16177. doi:10.48550/arXiv.2608.16177.

Andriushchenko, M., Souly, A., Dziemian, M., Duenas, D., Lin, M., Wang, J., Hendrycks, D., Zou, A., Kolter, Z., Fredrikson, M., et al. (2024). AgentHarm: A Benchmark for Measuring Harmfulness of LLM Agents. arXiv:2410.09024. doi:10.48550/arXiv.2410.09024.

Barua, S., Rahman, M., Sadek, M. J., Islam, R., Khaled, S., & Kabir, A. (2025). Guardians of the Agentic System: Preventing Many Shots Jailbreak with Agentic System. arXiv:2502.16750. doi:10.48550/arXiv.2502.16750.

Cai, Z., Shabihi, S., An, B., Che, Z., Bartoldson, B. R., Kailkhura, B., Goldstein, T., & Huang, F. (2025). AegisLLM: Scaling Agentic Systems for Self-Reflective Defense in LLM Security. arXiv:2504.20965. doi:10.48550/arXiv.2504.20965.

Carichon, F., Khandelwal, A., Fauchard, M., & Farnadi, G. (2025). The Coming Crisis of Multi-Agent Misalignment: AI Alignment Must Be a Dynamic and Social Process. arXiv:2506.01080. doi:10.48550/arXiv.2506.01080.

Cemri, M., Pan, M. Z., Yang, S., Agrawal, L. A., Chopra, B., Tiwari, R., Keutzer, K., Parameswaran, A., Klein, D., Ramchandran, K., Zaharia, M., Gonzalez, J. E., & Stoica, I. (2025). Why Do Multi-Agent LLM Systems Fail? arXiv:2503.13657. doi:10.48550/arXiv.2503.13657.

Chao, P., Robey, A., Dobriban, E., Hassani, H., Pappas, G. J., & Wong, E. (2023). Jailbreaking Black Box Large Language Models in Twenty Queries. arXiv:2310.08419. doi:10.48550/arXiv.2310.08419.

Chen, K., Zhen, T., Wang, H., Liu, K., Li, X., Huo, J., Yang, T., Xu, J., Dong, W., & Gao, Y. (2025). MedSentry: Understanding and Mitigating Safety Risks in Medical LLM Multi-Agent Systems. arXiv:2505.20824. doi:10.48550/arXiv.2505.20824.

Chern, S., Fan, Z., & Liu, A. (2024). Combating Adversarial Attacks with Multi-Agent Debate. arXiv:2401.05998. doi:10.48550/arXiv.2401.05998.

Dai, H., Ying, Z., Li, W., Wu, X., Xiao, Y., Zhang, T., et al. (2026). SafeFlow: Semantic Information-Flow Control for Blocking Malicious Propagation in Multi-Agent Systems. arXiv:2607.25255. doi:10.48550/arXiv.2607.25255.

Darley, J. M., & Latane, B. (1968). Bystander intervention in emergencies: Diffusion of responsibility. Journal of Personality and Social Psychology, 8(4), 377-383. doi:10.1037/h0025589.

de Witt, C. S., Krawiecka, K., Krawczuk, I., Hagag, B., Anderson, W. L., et al. (2025). Open Challenges in Multi-Agent Security: Towards Secure Systems of Interacting AI Agents. arXiv:2505.02077. doi:10.48550/arXiv.2505.02077.

Ding, X., Zhai, S., Song, L., Li, J., Shi, T., Meade, N., Reddy, S., Kang, J., & Zhao, J. (2026). The Blind Spot of Agent Safety: How Benign User Instructions Expose Critical Vulnerabilities in Computer-Use Agents. arXiv:2604.10577. doi:10.48550/arXiv.2604.10577.

Glukhov, D., Han, Z., Shumailov, I., Papyan, V., & Papernot, N. (2024). Breach By A Thousand Leaks: Unsafe Information Leakage in 'Safe' AI Responses. arXiv:2407.02551. doi:10.48550/arXiv.2407.02551.

Greenblatt, R., Shlegeris, B., Sachan, K., & Roger, F. (2023). AI Control: Improving Safety Despite Intentional Subversion. arXiv:2312.06942. doi:10.48550/arXiv.2312.06942.

Gu, X., Zheng, X., Pang, T., Du, C., Liu, Q., Wang, Y., Jiang, J., & Lin, M. (2024). Agent Smith: A Single Image Can Jailbreak One Million Multimodal LLM Agents Exponentially Fast. arXiv:2402.08567. doi:10.48550/arXiv.2402.08567.

Hagag, B., Anderson, W. L., Chakraborty, S., & de Witt, C. S. (2026). ORBIT: A Framework for Multi-Agent Safety and Security Evaluations. arXiv:2609.33102. doi:10.48550/arXiv.2609.33102.

Hammond, L., Chan, A., Clifton, J., Hoelscher-Obermaier, J., Khan, A., McLean, E., Smith, C., et al. (2025). Multi-Agent Risks from Advanced AI. arXiv:2502.14143. doi:10.48550/arXiv.2502.14143.

He, P., Lin, Y., Dong, S., Xu, H., Xing, Y., & Liu, H. (2025). Red-Teaming LLM Multi-Agent Systems via Communication Attacks. arXiv:2502.14847. doi:10.48550/arXiv.2502.14847.

Huang, J., Zhou, J., Jin, T., Zhou, X., Chen, Z., Wang, W., Yuan, Y., Lyu, M. R., & Sap, M. (2024). On the Resilience of LLM-Based Multi-Agent Collaboration with Faulty Agents. arXiv:2408.00989. doi:10.48550/arXiv.2408.00989.

Jones, E., Dragan, A., & Steinhardt, J. (2024). Adversaries Can Misuse Combinations of Safe Models. arXiv:2406.14595. doi:10.48550/arXiv.2406.14595.

Khan, R. M. S., Tan, Z., Yun, S., Fleming, C., & Chen, T. (2025). Agents Under Siege: Breaking Pragmatic Multi-Agent LLM Systems with Optimized Prompt Attacks. arXiv:2504.00218. doi:10.48550/arXiv.2504.00218.

Kothamasu, V., Smith, V., & Yadav, C. (2026). Hidden in Plain Sight: Benchmarking Agent Safety Against Decomposition Attacks with DECOMPBENCH. arXiv:2606.13994. doi:10.48550/arXiv.2606.13994.

Kumar, P., Lau, E., Vijayakumar, S., Trinh, T., Scale Red Team, et al. (2024). Refusal-Trained LLMs Are Easily Jailbroken As Browser Agents. arXiv:2410.13886. doi:10.48550/arXiv.2410.13886.

Laurito, W., Davis, B., Grietzer, P., Gavenciak, T., Bohm, A., & Kulveit, J. (2024). AI-AI Bias: large language models favor communications generated by large language models. arXiv:2407.12856. doi:10.48550/arXiv.2407.12856.

Lee, D., & Tiwari, M. (2024). Prompt Infection: LLM-to-LLM Prompt Injection within Multi-Agent Systems. arXiv:2410.07283. doi:10.48550/arXiv.2410.07283.

Li, F., Wu, J., Fu, T., Jaques, N., Zhou, W., & Kan, M. (2026). FlowSteer: Prompt-Only Workflow Steering Exposes Planning-Time Vulnerabilities in Multi-Agent LLM Systems. arXiv:2605.11514. doi:10.48550/arXiv.2605.11514.

Li, T., Ma, Y., Wen, H., Huang, Z., Zhou, Q., Fu, Z., & Cheng, G. (2026). Safe Multi-Agent Behavior Must Be Maintained, Not Merely Asserted: Constraint Drift in LLM-Based Multi-Agent Systems. arXiv:2605.10481. doi:10.48550/arXiv.2605.10481.

Li, X., Wang, R., Cheng, M., Zhou, T., & Hsieh, C. (2024). DrAttack: Prompt Decomposition and Reconstruction Makes Powerful LLM Jailbreakers. arXiv:2402.16914. doi:10.48550/arXiv.2402.16914.

Liu, L., Yu, H., Jiang, X., Wang, S., Qian, P., & Chen, Y. (2026). Operational Reframing and Approval-Framed Delegation in Multi-Agent LLM Safety. arXiv:2607.07097. doi:10.48550/arXiv.2607.07097.

Liu, Q., Song, R., & Ye, S. (2026). The Capability Paradox: How Smarter Auditors Make Multi-Agent Systems Less Secure. arXiv:2605.17480. doi:10.48550/arXiv.2605.17480.

Lynch, A., Wright, B., Larson, C., Ritchie, S. J., Mindermann, S., Hubinger, E., Perez, E., & Troy, K. (2025). Agentic Misalignment: How LLMs Could Be Insider Threats. arXiv:2510.05179. doi:10.48550/arXiv.2510.05179.

Mao, X., Qian, R., Chen, L., Gao, Y., Liao, J., Cai, J., Zhao, J., & Wang, C. (2026). Trust the Brand, Lose Control: How Identity Hijacks LLM Agent Orchestration. arXiv:2609.32635. doi:10.48550/arXiv.2609.32635.

Mazumder, A., Dipta, S. R., Lia, N. J., Khan, T., et al. (2026). AgentCollabBench: Diagnosing When Good Agents Make Bad Collaborators. arXiv:2605.08647. doi:10.48550/arXiv.2605.08647.

Milgram, S. (1963). Behavioral Study of obedience. The Journal of Abnormal and Social Psychology, 67(4), 371-378. doi:10.1037/h0040525.

Motwani, S. R., Baranchuk, M., Strohmeier, M., Bolina, V., Torr, P. H. S., Hammond, L., & de Witt, C. S. (2024). Secret Collusion among AI Agents: Multi-Agent Deception via Steganography. arXiv:2402.07510. doi:10.48550/arXiv.2402.07510.

Pihlakas, R., & Dagohoy, J. L. (2026). Open-source LLMs administer maximum electric shocks in a Milgram-like obedience experiment. arXiv:2605.21401. doi:10.48550/arXiv.2605.21401.

Qi, S., Zou, Y., Li, P., Lin, Z., Cheng, X., & Yu, D. (2025). Amplified Vulnerabilities: Structured Jailbreak Attacks on LLM-based Multi-Agent Debate. arXiv:2504.16489. doi:10.48550/arXiv.2504.16489.

Qi, X., Panda, A., Lyu, K., Ma, X., Roy, S., Beirami, A., Mittal, P., & Henderson, P. (2024). Safety Alignment Should Be Made More Than Just a Few Tokens Deep. arXiv:2406.05946. doi:10.48550/arXiv.2406.05946.

Reason, J. (2000). Human error: models and management. BMJ, 320(7237), 768-770. doi:10.1136/bmj.320.7237.768.

Rosser, J., & Foerster, J. (2025). AgentBreeder: Mitigating the AI Safety Risks of Multi-Agent Scaffolds via Self-Improvement. arXiv:2502.00757. doi:10.48550/arXiv.2502.00757.

Russinovich, M., Salem, A., & Eldan, R. (2024). Great, Now Write an Article About That: The Crescendo Multi-Turn LLM Jailbreak Attack. arXiv:2404.01833. doi:10.48550/arXiv.2404.01833.

Shah, R., Feuillade--Montixi, Q., Pour, S., Tagade, A., Casper, S., & Rando, J. (2023). Scalable and Transferable Black-Box Jailbreaks for Language Models via Persona Modulation. arXiv:2311.03348. doi:10.48550/arXiv.2311.03348.

Shehata, D., & Li, M. (2026). The Bystander Effect in Multi-Agent Reasoning: Quantifying Cognitive Loafing in Collaborative Interactions. arXiv:2605.10698. doi:10.48550/arXiv.2605.10698.

Tian, Y., Yang, X., Zhang, J., Dong, Y., & Su, H. (2023). Evil Geniuses: Delving into the Safety of LLM-based Agents. arXiv:2311.11855. doi:10.48550/arXiv.2311.11855.

Triedman, H., Jha, R., & Shmatikov, V. (2025). Multi-Agent Systems Execute Arbitrary Malicious Code. arXiv:2503.12188. doi:10.48550/arXiv.2503.12188.

Wallace, E., Xiao, K., Leike, R., Weng, L., Heidecke, J., & Beutel, A. (2024). The Instruction Hierarchy: Training LLMs to Prioritize Privileged Instructions. arXiv:2404.13208. doi:10.48550/arXiv.2404.13208.

Wang, S., Zhang, G., Yu, M., Wan, G., Meng, F., Guo, C., Wang, K., & Wang, Y. (2025). G-Safeguard: A Topology-Guided Security Lens and Treatment on LLM-based Multi-agent Systems. arXiv:2502.11127. doi:10.48550/arXiv.2502.11127.

Wit, M. C. C., & Pang, J. (2025). Evaluating Multi-Agent Defences Against Jailbreaking Attacks on Large Language Models. arXiv:2506.23576. doi:10.48550/arXiv.2506.23576.

Wu, Q., Bansal, G., Zhang, J., Wu, Y., Li, B., Zhu, E., Jiang, L., Zhang, X., Zhang, S., Liu, J., Awadallah, A. H., White, R. W., Burger, D., & Wang, C. (2023). AutoGen: Enabling Next-Gen LLM Applications via Multi-Agent Conversation. arXiv:2308.08155. doi:10.48550/arXiv.2308.08155.

Xu, Z., Qi, M., Wu, S., Zhang, L., Wei, Q., He, H., & Li, N. (2025). The Trust Paradox in LLM-Based Multi-Agent Systems: When Collaboration Becomes a Security Vulnerability. arXiv:2510.18563. doi:10.48550/arXiv.2510.18563.

Xu, Z., Zhang, X., Luo, H., Jin, Y., Dong, Y., & Salam, H. (2026). MasDrift: Benchmarking Authorization Preservation Across Multi-Agent Architectures. arXiv:2608.07556. doi:10.48550/arXiv.2608.07556.

Ying, Z., Yan, J., Luo, H., Zou, Q., Liu, A., & Liu, X. (2026). Delegated Misalignment: How Multi-Agent Structures Amplify LLM Safety Risks. arXiv:2609.27900. doi:10.48550/arXiv.2609.27900.

Yu, M., Wang, S., Zhang, G., Mao, J., Yin, C., Liu, Q., Wen, Q., Wang, K., & Wang, Y. (2024). NetSafe: Exploring the Topological Safety of Multi-agent Networks. arXiv:2410.15686. doi:10.48550/arXiv.2410.15686.

Zeng, Y., Wu, Y., Zhang, X., Wang, H., & Wu, Q. (2024). AutoDefense: Multi-Agent LLM Defense against Jailbreak Attacks. arXiv:2403.04783. doi:10.48550/arXiv.2403.04783.

Zhang, J., Li, T., Jurayj, W., Zhan, H., Van Durme, B., & Khashabi, D. (2026). Many-Tier Instruction Hierarchy in LLM Agents. arXiv:2604.09443. doi:10.48550/arXiv.2604.09443.

Zhang, Z., Zhang, Y., Li, L., Gao, H., Wang, L., Lu, H., Zhao, F., Qiao, Y., & Shao, J. (2024a). PsySafe: A Comprehensive Framework for Psychological-based Attack, Defense, and Evaluation of Multi-agent System Safety. arXiv:2401.11880. doi:10.48550/arXiv.2401.11880.

Zhang, Z., Cui, S., Lu, Y., Zhou, J., Yang, J., Wang, H., & Huang, M. (2024b). Agent-SafetyBench: Evaluating the Safety of LLM Agents. arXiv:2412.14470. doi:10.48550/arXiv.2412.14470.

© 2026 Pranay Mahendrakar. Licensed under CC BY 4.0.

DOI: 10.5281/zenodo.24420073