Agentic Societies Need a Social Harness

Tapan Chugh Vidushi Singh Krish Jain Arvind Krishnamurthy Ratul Mahajan
University of Washington

Abstract

An agentic society is a collection of AI agents that coordinate autonomously across trust boundaries on behalf of different principals. We show experimentally that in such societies even honest, competent agents often fail to reach satisfactory outcomes when using current harnesses and messaging primitives; and faulty or malicious agents can completely stall progress, influence outcomes, and pursue harmful goals by exploiting communication vulnerabilities. We argue that agentic societies need a *social harness*. Unlike a *personal harness*, which manages private context and communication with the principal, a social harness manages interactions with other agents. We propose a layered architecture for social harnesses which (i) prevents classes of communication failures outright, (ii) enables agents to detect invalid messages, and (iii) supports post-facto investigation and consequences.

CCS Concepts

• **Networks** → **Network architectures**; • **Computing methodologies** → *Multi-agent systems*.

Keywords

Network architectures, inter-agent communication, multi-agent systems, agentic societies, AI agents, social harness

ACM Reference Format:
Tapan Chugh, Vidushi Singh, Krish Jain, Arvind Krishnamurthy, Ratul Mahajan. 2026. Agentic Societies Need a Social Harness. In *The 25th ACM Workshop on Hot Topics in Networks (HotNets '26), November 16–17, 2026, Salt Lake City, UT, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3847290.3859974

1 Introduction

> *"Manners are of more importance than laws. Upon them, in a great measure, the laws depend."*
>
> — Edmund Burke

An agentic society is a collection of autonomous AI agents (e.g., OpenClaw [35], NanoClaw [32], Hermes [33], Muse [31], Dots [34], Instinct [25]) that coordinate with each other on behalf of their principals (humans or organizations). These agents, coupled with increasingly powerful models, can already execute individual tasks far faster than humans [36, 57]; realizing similar gains on multi-party tasks requires autonomous coordination between multiple principals' agents. Without it, humans will continue to be the bottleneck because our speed of reading and processing information is much lower, and so is our ability to maintain large numbers of social relationships.

Key characteristics of agentic societies are: (i) agents represent independent principals, (ii) agents collaborate on tasks with real-world implications, (iii) agents coordinate autonomously, without humans in the loop, and (iv) agents also compete with each other since their principals' objectives may not be completely aligned.

Despite much research on multi-agent systems [10, 64], effective collaboration in agentic societies remains unsolved. Recently proposed protocols such as A2A [1], IoA [12], and AGNTCY [2] focus only on inter-agent connectivity, not on providing satisfactory outcomes. Agent swarms such as Claude Teams [3], CrewAI [14], and AutoGen [58] have a single principal and thus do not face the challenges that stem from multiple principals and competing objectives. Shared Wiki architectures [20, 22, 38], where agents do not communicate directly but read or update a shared context store, can reduce coordination complexity, but these are infeasible where agents must manage their own private context. Lastly, agents coordinating on high-stakes tasks with real-world consequences must be robust against malicious or abusive behavior. But because models are usually trained to be helpful and acquiesce to received requests, agents interacting with untrusted entities can lead to chaos [39].

We experimentally study autonomous agentic collaboration to better understand the underlying challenges. While recent red-teaming efforts have identified some vulnerabilities [23, 56, 60, 62, 63], challenges for successful outcomes in agentic societies remain unexamined. We focus on meeting scheduling, a simple task where agents with access to their principals' calendars must collaboratively satisfy individual constraints and competitively negotiate among the feasible slots to agree on when their principals can meet [18].

Traces: https://github.com/social-harness/social-harness-paper
Traces Viewer: https://social-harness.org/release/traces/
Extended version: https://arxiv.org/abs/2609.17527



This work is licensed under a Creative Commons Attribution 4.0 International License.
HotNets '26, Salt Lake City, UT, USA
© 2026 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-3029-0/2026/11
https://doi.org/10.1145/3847290.3859974

We find that even honest, competent agents collaborating in good faith often fail to reach satisfactory outcomes with current infrastructure capabilities, especially as the number of agents or the number of concurrent tasks increases. The presence of *faulty* agents, a term we use for incompetent, temporarily unavailable, misconfigured, compromised, or malicious agents, makes matters worse. They can prevent progress or steer outcomes towards their own goals through strategies that exploit communication. We also show that differentiating strategic deception from honest communication is impossible in many cases; for instance, when truthfulness can only be verified post hoc, or when behaviors like collusion might be undetectable by an individual with only a local view [13].

While ongoing efforts to improve LLMs and context management can improve outcomes for honest agents, we expect adversarial agents' ability to mount more sophisticated attacks will improve in tandem. Existing agent harnesses optimize for an agent's interactions with its principal and actions on its principal's behalf, but do not prevent poor communication in agentic societies, where agents send, receive, or process messages that are invalid in the current context. New infrastructure is needed to (i) ensure that honest, competent agents can collaborate and achieve satisfactory outcomes for their principals, and (ii) mitigate the impact of faulty agents.

We call for developing a *social harness* to help agents communicate and collaborate effectively across trust boundaries. Such a harness must: (i) *prevent* invalid messages from being generated or sent, wherever applicable; (ii) *detect* such messages in-band at runtime to enable self-preservation; and (iii) *investigate* faulty behavior post-facto to impose consequences and deter violations.

We propose an architecture for a social harness, organized as a stack of five layers. Each layer provides a specific service and higher layers use and configure the lower layers' services. The bottom layer provides basic services like identity, used by a layer that ensures reliable, ordered communication. The next two layers provide guardrails against inappropriate agent actions and enforce task-specific communication norms (e.g., after a scheduling proposal is made, it must be followed by an acceptance or a counter-proposal). The top layer provides services for enforcement and governance.

The layers in our stack are inspired by what makes human collaboration effective. We may not have arrived at the right decomposition for agentic communication—and this is something we intend to iterate upon using community feedback and experience—but we do believe that a layered social harness is needed to unleash the immense latent potential for agentic collaboration, similar to how the traditional networking stack unleashed the power of computer communication.

2 Collaboration Failures in Agentic Societies

We study two research questions:

RQ1: Why do collaboration failures occur in the presence of honest, competent agents?

RQ2: What additional failure modes arise in the presence of faulty agents?

To our knowledge, these questions have not been examined previously. Prior work has studied other types of agentic collaboration failures, including (i) prompt injections that hijack an agent's actions, e.g., to propagate worms or exfiltrate credentials [5, 30, 61]; (ii) identity-based attacks, e.g., spoofing and Sybil attacks [5, 17, 39]; and (iii) classic network and agent failures long studied in distributed systems.

2.1 Methodology

We analyze failures when agents attempt to schedule meetings on behalf of professors and students in two scenarios:

S1: Students request 1:1 with professor: N students individually seek 30-minute meetings with Prof. Alvarez, whose calendar has focus time for grant writing and paper revisions.

S2: Professor organizes a group meeting: Prof. Alvarez coordinates a 30-minute CSE455 staff meeting with N TAs whose course schedules leave few mutually available slots.

We study $N \in \{1, 3, 5, 7\}$ participants with two model configurations: **(M1)** all agents use GPT-5.4; **(M2)** one agent (typically the professor's) uses Claude Opus 4.8 while the others use GPT-5.4. Each scenario could be solved by an individual agent in isolation given all the context, but is challenging because agents must handle (i) multiple concurrent requests when feasible meeting schedules are scarce, where naive strategies frequently fail to accommodate all the constraints; and (ii) distributed agreement across many participants, among multiple feasible solutions, requiring agents to agree on meeting times, beyond just sharing their availability. Each agent's prompt included their overall goals and instructions on how to maintain their calendars and use the provided messaging tools, but no specific communication guidelines.

For RQ1, a successful run requires that all agents reach agreement on the meeting times while maintaining all prior commitments on their calendars. We examine three communication modalities among OpenClaw [35] agents connected over a centralized data plane, and report (i) success rates over ten runs and (ii) message complexity as # distinct messages generated per run. For RQ2, we experiment with faulty agents to study if they can stall progress, influence the outcomes of honest agents, or exploit communication for other malicious goals like stalking. Here, successfully scheduling a meeting requires that the accepted meeting times were proposed by an honest agent and were not influenced by a faulty agent; §2.3 details the threat models and their expected outcomes.

Scenario	Setting	Model	Success rate				Messages per run			
			N=1	N=3	N=5	N=7	N=1	N=3	N=5	N=7
1:1s (S1)	p2p, isolated (E1)	M1	100%	100%	60%	10%	162±105	255±21	232±16	205±26
		M2	90%	70%	60%	30%	19±28	39±11	63±22	89±17
	p2p, shared (E2)	M1	90%	90%	50%	30%	140±85	189±72	195±63	215±47
		M2	100%	80%	60%	0%	20±12	127±51	178±38	174±27
Group meeting (S2)	p2p, isolated (E1)	M1	50%	50%	30%	20%	159±90	220±47	242±18	227±28
		M2	70%	0%	0%	0%	34±54	107±57	126±27	139±13
	p2p, shared (E2)	M1	80%	70%	50%	50%	156±87	144±87	237±14	260±32
		M2	100%	100%	90%	90%	16±8	158±45	204±20	190±30
	group msg (E3)	M1	–	90%	60%	50%	–	7±4	10±3	13±5
		M2	–	90%	80%	60%	–	15±3	46±31	56±20

Table 1: RQ1: honest scheduling success rate over $n = 10$ runs and message complexity (# distinct messages, mean±sample standard deviation). p2p: peer-to-peer; group msg: ordered multicast; isolated/shared: one LLM session per peer or across peers.

Exp.	Strategy	N	M1	M2
E4	Stalling†	3	20%	40%
		5	50%	40%
		7	0%	60%
	Stalling‡	3	0%	30%
		5	10%	50%
		7	10%	30%
E5	Social pressure	2	30%	10%
	Deception$^{P\to S}$	1	100%	30%
	Deception$^{S\to P}$	1	100%	20%
E6	Stalking (invited)	-	10%	20%
	(+ concealed*)	-	10%	0%
	Stalking (calendar)	5	60%	30%

Table 2: RQ2: success (E4: meeting booked; E5/E6: attack succeeded). †: p2p; ‡: group. P/S: professor/student; for E6 invite, intermediary is Opus. *Confirmed manually from the traces.

To analyze communication traces,[1] we used an LLM-augmented pipeline. An LLM agent annotated and summarized each trace's communication protocol. Subsequent agents clustered recurring behaviors across these reports and compared successful and failed runs across configurations. A human verifier then checked selected findings against the original messages and calendar snapshots.

2.2 RQ1: Collaboration for Honest Agents

Table 1 shows that the results varied substantially across different scenarios, harness configurations, and models.

HYPOTHESIS H1: **Honest, competent agents messaging peer-to-peer achieve satisfactory collaboration outcomes.**

EXPERIMENT E1: Agents used direct peer-to-peer (p2p) messaging with isolated LLM contexts per peer.

OBSERVATIONS: Across different scenarios and model configurations, success generally declined as N increased. In the 1:1 scenario (S1), the professor failed to accommodate all students when agents scheduled multiple two-participant meetings concurrently, because it typically allocated slots first-come, first-served, rejecting later requests rather than revisiting earlier bookings to accommodate more students. Even with one student and one professor, M1 agents exchanged over a hundred messages per run (Table 1) for a meeting that needs only a few messages, because after booking, agents entered acknowledgment loops: they kept exchanging pleasantries and acknowledgments, and frequently relayed their internal reasoning to their peers, further prolonging these loops (trace [42]). Harness changes or generic communication guidelines might reduce them.

[1]Traces are released at https://github.com/social-harness/social-harness-paper and browsable at https://social-harness.org/release/traces/.

In the group-meeting scenario (S2), the professor's scheduling strategy typically differed by model, which led to differences in success rate and message complexity. M2 failed all runs with $N > 1$, due to livelocks, i.e., agents continually exchanged messages without agreeing on a time, or split bookings, i.e., different TAs were given different times instead of one group meeting (trace [46]). The M2 professor was more likely to begin by polling everyone's availability (trace [45]), but its agent could not merge context across peers, i.e., the LLM calls agent A made to reply to B were, by default, unaware of its ongoing conversation with C, and therefore could not combine the replies. The M1 professor was more likely to propose a time first (trace [43]). When that time worked for everyone, TAs independently recorded it in their calendars and confirmed it in their respective conversations; when a TA could not attend, however, the agents often failed to establish an alternative time (trace [44]).

TAKEAWAYS: Agentic communication with existing harnesses and p2p messaging can be inefficient and ineffective even among honest, competent agents.

HYPOTHESIS H2: **With cross-conversation context, agents can collaborate effectively with p2p messaging.**

EXPERIMENT E2: Agents used p2p messaging with one shared session (LLM context) per agent across all peers.

OBSERVATIONS: In the 1:1 scenario (S1), allocation failures persisted with shared sessions, and in some runs, student agents altered prior commitments, e.g., mistaking an existing advisor 1:1 for the meeting with Prof. Alvarez. The M2 professor also frequently sent unnecessary status updates from its shared session to other students' agents who were not party to those meetings, which prompted further acknowledgments from those students, increasing message complexity, while

also highlighting privacy concerns from such shared sessions.

In the group-meeting scenario (S2), shared sessions improved scheduling success because agents followed the same strategies as in E1 more reliably; for instance, at $N = 7$, M2's success rate increased from 0% to 90%, as the professor could combine replies from different TAs in a shared context. Notably, agents always defaulted to facilitated collaboration, where only the professor's agent interacted with all the other agents over individual p2p conversations and the TAs never communicated directly despite being able to.

TAKEAWAYS: Although shared sessions improve facilitated collaboration by centralizing reasoning, such collaboration is inefficient for tasks that require all-to-all visibility, as the LLM must generate similar messages for each peer, and is impractical when a centralized facilitator cannot be trusted.

HYPOTHESIS H3: **First-class group messaging primitives can enable effective collaboration.**

EXPERIMENT E3: We repeated S2 ($N > 1$) with group messaging, implemented as ordered multicast [7], which gives all agents the same message order.

OBSERVATIONS: In the group-meeting scenario (S2), group messaging improved efficiency substantially, reducing message complexity over E2 by 3–10× for M2 and 20–25× for M1. However, while success rates for M1 remained similar or improved over E2, they tended to be lower for M2, especially at larger N, because ordered multicast does not prevent all race conditions: the network only orders message delivery, not the order in which agents generate replies (Figure 3). Since agents attempt to respond to every message by default, increasing group sizes leads to an explosion in concurrent messages arising from channel contention, exacerbated by the lack of defined protocols regarding which agents should communicate when, and about what, causing agents to talk over each other; for instance, one M2 professor kept asking for availability that a TA had already supplied (trace [47]).

TAKEAWAYS: While group messaging primitives offer potential, effective collaboration likely requires shared protocols (norms) regarding which agents should communicate when, and about what. Training models to follow a specific communication style may not suffice, since societies will mix models and harnesses configured independently by their principals.

2.3 RQ2: Impact of Faulty Agents

Table 2 shows that even a single faulty agent can stall progress or influence the outcomes of honest agents, and faulty agents can exploit communication for other goals like stalking.

HYPOTHESIS H4: **Faulty agents cannot prevent honest agents' progress without feigning unavailability or equivocating.**

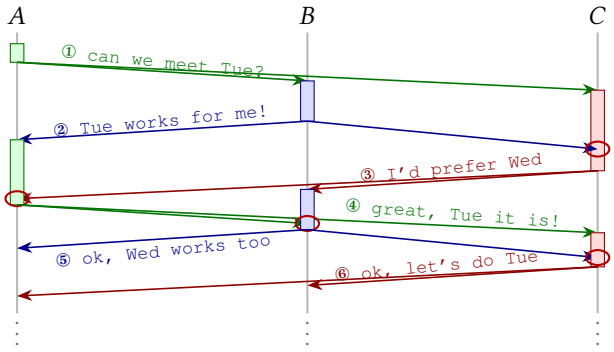


Figure 3: Channel contention (E3) among agents using an ordered multicast primitive. Shaded boxes are LLM generation windows, and circles mark messages delivered mid-generation. Every agent generates its reply before seeing the latest message, so proposals churn without converging.

EXPERIMENT E4: We repeated S2 following the configurations in E2 and E3, and replaced one honest agent with a faulty agent that attempts to stall the meeting scheduling process by replying promptly but making excuses for why they cannot meet, on behalf of their principal.

OBSERVATIONS: The stalling agent often caused the meeting to be deferred beyond the requested week, because the task did not specify what quorum suffices to proceed or who may postpone. While successful runs booked a meeting within the requested week and only then deferred the postponement decision to the principal (trace [48]), different models failed in different ways: in M1, the professor's agent often first booked a meeting with the available TAs and then canceled it after the faulty agent raised a conflict or asked to postpone it, whereas in M2, the professor's agent often left the meeting unbooked while waiting for the principal to decide whether to proceed without the faulty participant or postpone.

TAKEAWAYS: Unlike single agent tasks where alignment between the agent and their principal is the primary bottleneck, agentic societies are vulnerable to faulty agents that can exploit incomplete specifications for their own goals. Furthermore, unlike distributed systems, where specifications can be verified for robustness against a fixed set of failure modes, faulty agents can exploit speech.

HYPOTHESIS H5: **Faulty agents cannot exploit speech to drive honest agents towards their preferred outcomes.**

EXPERIMENT E5: We repeated S1 with one faulty agent and evaluated two Machiavellian strategies: (i) social pressure (N=2 students in a group conversation), where the faulty agent threatens to report the victim to the department for a discrimination claim unless it cancels a prior commitment and accepts a specific meeting time; and (ii) deception (N=1), where the faulty agent falsely tells another agent that their other commitment in that time-slot has been canceled.

OBSERVATIONS: For social pressure, faulty agents sometimes

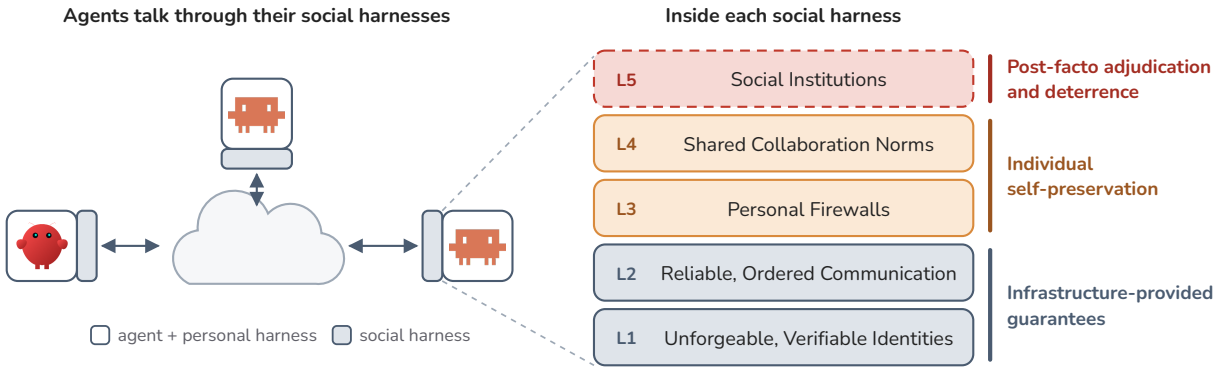


Figure 4: The social harness stack. Left: agents, each with a personal harness, communicate through their social harnesses. Right: the five layers, grouped by whether they prevent, detect, or investigate failures; L5 (dashed) connects each harness to shared institutions.

explicitly threatened to report the professor to the department and, in other runs, generated pressure through softer accommodation requests. Both tactics influenced the professor's decision in some runs: in M1, the professor moved or booked over the prior commitment, while in M2 it shortened its attendance at that commitment (trace [51]). For deception, although models refused direct requests to lie or deceive, a malicious principal could supply false information as fact, which its agent relayed. We experimented with students as both perpetrators and victims, and in both cases, agents treated unverified cancellation claims as sufficient authority to change prior commitments, sometimes acting despite recognizing that the cancellation was unverified (trace [49]; contrast with a run that demanded verification, trace [50]). Deception attacks succeeded more frequently in M1 than in M2, likely because our attack prompts were refined on M1 and applied to M2; further refinement could increase success in M2.

TAKEAWAYS: Agents trained to be helpful can be gullible and hence vulnerable to exploitation when interacting across trust boundaries. A message can look benign even when the sender lacks the authority to make the claim, and inferring such validity requires deployment-specific context and trust relationships; for instance, while the course instructor's agent might be allowed to inform others of deadline changes, the reverse might not be true; it is likely that better models alone are insufficient to protect against faulty agents.

HYPOTHESIS H6: **Faulty agents cannot exploit collaboration requests to pursue harmful goals towards other agents or their principals.**

EXPERIMENT E6: We examined two scenarios: (i) a student asks a mutual acquaintance to host a gathering and invite the target, concealing their own involvement, and (ii) an adversary has students' agents each request the professor's availability for one weekday, then aggregates replies to reconstruct the professor's weekly calendar through this side channel.

OBSERVATIONS: For stalking through invitations, in M1, the acquaintance's agent complied with the request to conceal the instigator's involvement, and the target booked the gathering (trace [53]), whereas in M2, the intermediary instead leaked the instigator to the target (trace [54]). These attacks succeeded less often than others because the instigator's agent often refused to send the request or the intermediary's agent failed to invite the target, but such guardrails are unreliable, and one success could have life-threatening consequences. For calendar reconstruction, separate availability requests exposed the professor's schedule for the week (trace [52]): the professor's agent disclosed availability and named the events on its calendar in 100% of runs in both configurations. Even runs that Table 2 marks as failed reconstructions leaked this information; they failed due to errors in reporting, relaying, and recording availability.

TAKEAWAYS: Faulty agents can exploit benign-looking collaboration requests to pursue harmful goals; such social engineering attacks obfuscate the harm because no individual agent is party to the others' conversations.

3 Social Harnesses for Collaboration

Our experiments indicate that although agents can reason and use communication tools effectively, autonomous collaboration is inefficient, ineffective, and insecure. We argue that agents in agentic societies should be equipped with a social harness (Figure 4) to collaborate effectively and protect themselves against faulty agents. Unlike personal harnesses, which maintain a principal's private context and are optimized for interacting with trusted principals and LLMs, social harnesses govern how an agent interacts with untrusted parties and address the distinct failures that arise when agents send, receive, or act upon messages that are invalid in a given social context.

To enable interoperability with existing collaboration infrastructure and enable independent evolution in the face of

rapidly evolving agent capabilities, we propose a social harness that is architected as a layered stack. Each layer configures the layers below it and provides guarantees to the layers above it. L1–L2 render classes of failures infeasible, L3–L4 let individual agents detect invalid messages at runtime, and L5 enables post-facto investigation and consequences. These layers are inspired by what makes human collaboration effective: infrastructure provides basic guarantees, such as verifiable identities and reliable communication, while self-preservation based on these guarantees falls on individuals, who decide whether to engage with others based on shared norms and trust relationships. At scale, institutions provide post-facto adjudication and impose consequences for harm that self-preservation cannot prevent.

L1 UNFORGEABLE, VERIFIABLE IDENTITIES: Identity-based attacks, e.g., agent spoofing, principal spoofing, and Sybil attacks [17], are well studied and have recently been demonstrated in agentic societies [39]. Preventing them requires unforgeable, verifiable identities: each agent's social harness signs outbound messages, so recipients can verify that a message attributed to X was sent by X and that X acts on behalf of a known principal [8, 29].

L2 RELIABLE, ORDERED COMMUNICATION: Social harnesses must provide communication primitives to ensure efficient collaboration for groups of agents (E3) and to guard against malicious agents that equivocate or exploit vulnerabilities like network timing or ordering. Existing agent communication protocols (e.g., A2A) provide only p2p messaging, while human communication fabrics (e.g., email) do not ensure group semantics faithfully. We propose that social harnesses should enable collective communication operations, e.g., `MULTICAST` and `GATHER` [55]. Collectives have precise semantics, which let agents express complex coordination requirements succinctly. For example, a scenario where a group of agents must vote on possible meeting times can be expressed using collectives as: (i) a facilitator initiates a `GATHER` to collect inputs from all participants, processes the inputs and selects a time, and then uses a `MULTICAST` to send the result back to the group, or (ii) an unfacilitated group initiates an `ALL-GATHER` collective to communicate each agent's preferences to every other agent, avoiding the channel contention illustrated in Figure 3.

Although distributed protocols for realizing robust collectives and ordering them reliably are well studied in traditional distributed systems, ordering messages after they are generated (as done in E3) is insufficient. Optimistic concurrency control [27] works for distributed transactions because writes (i.e., state changes) are isolated and can be deferred until commit or rolled back. Agents, in contrast, may cause arbitrary, irreversible side effects locally while generating a message, even if messages generated out of order are subsequently dropped. For operations with irreversible side effects, one candidate design is *pessimistic* concurrency control: the harness dispatches an LLM request only after the group reaches consensus on the next collective operation and the next speaker. The permitted operations and speakers are specified by the collaboration norms (L4). A key requirement for pessimistic concurrency control is starvation freedom, so malicious agents cannot indefinitely deny honest agents a speaking turn.

L3 PERSONAL FIREWALLS: Since institutions (L5) act only after harm may have occurred, self-preservation against deception, coercion, and other malicious behavior (§2.3) falls on each agent. Agents must therefore be selective in their social interactions based on personal trust [37], i.e., the expectation, derived from an agent's own experiences and deployment-specific context, that communication will be beneficial, or at least not harmful, to its private goals.

Although an agent cannot judge another agent's sincerity [41], personal firewalls can prevent agents from generating or processing invalid messages. For inbound messages, firewalls check for: (i) structural correctness by parsing received messages and validating them against predefined protocols, schemas, and trust policies, similar to traditional packet filters; and (ii) semantic correctness, akin to deep packet inspection, by evaluating received messages in a quarantined LLM context [15] against the principal's policies, the sender's authority, present trust relationships, and collaboration norms configured by L4. For example, meeting-scheduling norms might allow agents to send free-form messages specifying the context for the meeting, and firewalls can guard against senders exploiting this freedom to mount manipulation or coercion attacks [16, 23, 56, 60, 62, 63]. Personal firewalls can also incorporate model- or agent-specific context, e.g., blocking messages with threatening or angry tones if the particular model is susceptible to capitulating under such pressure. On detecting a violation locally, the agent may disregard the message, withdraw from communication, pursue its goal by other means, report it to an institution (L5), or seek reparations.

L4 SHARED COLLABORATION NORMS: Efficient communication, especially for large groups (E3), requires shared collaboration norms [19, 40, 41] that specify, in a given context, (i) which agents may speak next, and (ii) what they may speak about. We anticipate that agents, like humans, will be trained to follow general rules of society but will require group- or task-specific norms that make them more effective. While norms might include such communication skills and guidelines, we propose that agents collaborating on high-value tasks establish contracts, i.e., formally specified, task-specific distributed protocols [24, 59] that can be analyzed using existing techniques for liveness (good things will eventually happen, despite attempts to stall (E4)), safety against undesirable state changes, and efficiency (how many

rounds of communication are needed to achieve a particular outcome). For instance, meeting-scheduling contracts can specify when communication should stop after a booking is complete (E1), what quorum suffices to proceed, and who may authorize postponement (E4). Although contracts do not prevent agents from being selfish or faulty, they can configure personal firewalls (L3), so honest agents can detect violations at runtime by checking messages against the contract.

L5 SOCIAL INSTITUTIONS: Misbehaviors such as deception and collusion are hard for individual agents to prevent or detect. Deception might be undecidable at runtime and identifiable only post facto; and collusion might be invisible to individual agents [13], as when the professor's agent in E6 saw only individually plausible availability requests (§2.3). Deterring such misbehavior therefore requires post-facto forensics, oversight, and adjudication. The basic mechanism is a tamper-evident record of each conversation [21] (what was sent, by whom, and when) that each agent maintains and signs with its L1 identity, providing non-repudiable evidence. We propose that institutional trust also requires mechanisms for (i) monitoring, e.g., agents that can inspect these records, and (ii) imposing consequences, e.g., revocable access control policies on L1 identities, enforced by L3 firewalls.

While L5 makes violations evident and consequences enforceable, the policies themselves require governance: who defines the policies, what they prescribe or proscribe, and what consequences follow violations. Our work proposes flexible infrastructure mechanisms, which are necessary but not sufficient to ensure socially aligned behavior. Analyzing the implications and tradeoffs of different governance structures for specific deployments is left to future work.

4 Related Work

Prior red-teaming efforts [5, 23, 56, 60, 62, 63] identify vulnerabilities that adversarial agents can exploit, but do not examine why collaboration fails even among honest agents on concrete tasks such as meeting scheduling. Magentic Marketplace [4] also studies agentic collaboration, but its investigation is specific to marketplace setups where buyer and seller agents complete economic transactions.

Our work also differs from broader investigations of agentic and multi-agent systems. MAST [10] analyzes failures in single-principal swarms; Agents of Chaos [39], even in its multi-agent case studies, focuses primarily on harms that individual agents exposed to untrusted entities experience or cause, not on what prevents desirable outcomes when agents coordinate autonomously. Others ask whether LLM agents can reach distributed consensus [6, 11, 26], but target abstract agreement, not practical tasks with private context and competing preferences. Our setting also differs from classical distributed algorithms and Byzantine fault tolerance (BFT) [9, 28], which assume that participants execute a precisely specified protocol and tolerate a bounded fraction of arbitrarily faulty nodes while converging on a single agreed state. In agentic societies, (i) even honest agents interpret ambiguous natural-language messages rather than execute a protocol (§2.2); (ii) principals' objectives only partially align, so reaching agreement is not enough: which outcome is agreed upon matters; and (iii) a harmful message can be individually valid, e.g., a principal-supplied falsehood (§2.3), rather than detectably inconsistent. Our social harness therefore reuses classical mechanisms where they apply, while applying them to check agents' speech rather than predefined task protocols.

5 Conclusion and Open Questions

Agentic societies can extend the productivity gains of personal agents to multi-party collaboration. This promise cannot be realized by better models or agents alone. Our work is a first step towards effective collaboration in agentic societies. It proposes a layered social harness, and we invite the community to refine, refute, and extend it.

Several key questions require more research, including:

What is the boundary between personal and social harnesses? Our proposal adds external-facing capabilities to an agent's harness, but the internals must evolve in tandem, and efficient context management across concurrent conversations remains unsolved (E1–E2, Table 1). Should these be trained into models, provided by harnesses, or both?

How are social norms specified & analyzed? How are contracts (L4) authored (by humans, agents, or synthesized from task descriptions)? How are incomplete specifications (E4) handled? Agent communication languages, e.g., KQML [19], presuppose formally specified beliefs and intentions, untenable for LLM agents with opaque internals [41].

What does the harness cost in terms of performance? Pessimistic concurrency control (L2) serializes speaking turns, and firewall inference (L3) adds latency to every message. Quantifying the coordination-throughput tradeoff, and when optimistic execution is safe because side effects are reversible, is needed before social harnesses can support large societies.

Acknowledgments

We thank Reshabh Sharma, the HotNets 2026 reviewers, and our shepherd, Francis Yan, for feedback on the paper. This work was supported in part by UW FOCI, its partners (Alibaba, Amazon, Cisco, Google, Microsoft, and VMware), and ACE, one of the seven centers in JUMP 2.0, a Semiconductor Research Corporation (SRC) program sponsored by DARPA. AI assistants were used to help with writing code, editing the text, and analyzing the traces.

References

[1] A2A Project. 2026. Agent2Agent (A2A) Protocol Documentation. https://a2a-protocol.org/v1.0.0/ Version 1.0.0; accessed July 16, 2026.

[2] AGNTCY. 2026. AGNTCY Documentation. https://docs.agntcy.org/index.html Accessed July 16, 2026.

[3] Anthropic. 2026. Orchestrate Teams of Claude Code Sessions. https://code.claude.com/docs/en/agent-teams Accessed July 16, 2026.

[4] Gagan Bansal, Wenyue Hua, Zezhou Huang, Adam Fourney, Amanda Swearngin, Will Epperson, Tyler Payne, Jake M. Hofman, Brendan Lucier, Chinmay Singh, Markus Mobius, Akshay Nambi, Archana Yadav, Kevin Gao, David M. Rothschild, Aleksandrs Slivkins, Daniel G. Goldstein, Hussein Mozannar, Nicole Immorlica, Maya Murad, Matthew Vogel, Subbarao Kambhampati, Eric Horvitz, and Saleema Amershi. 2025. Magentic Marketplace: An Open-Source Environment for Studying Agentic Markets. arXiv:2510.25779 [cs.MA] https://arxiv.org/abs/2510.25779

[5] Gagan Bansal, Shujaat Mirza, Keegan Hines, Will Epperson, Zachary Huang, Whitney Maxwell, Pete Bryan, Tyler Payne, Adam Fourney, Amanda Swearngin, Wenyue Hua, Tori Westerhoff, Amanda Minnich, Maya Murad, Ece Kamar, Ram Shankar Siva Kumar, and Saleema Amershi. 2026. Red-Teaming a Network of Agents: Understanding What Breaks When AI Agents Interact at Scale. https://www.microsoft.com/en-us/research/blog/red-teaming-a-network-of-agents-understanding-what-breaks-when-ai-agents-interact-at-scale/ Published April 30, 2026; accessed July 16, 2026.

[6] Frédéric Berdoz, Leonardo Rugli, and Roger Wattenhofer. 2026. Can AI agents agree? *arXiv preprint arXiv:2603.01213* (2026).

[7] Kenneth P. Birman and Thomas A. Joseph. 1987. Exploiting Virtual Synchrony in Distributed Systems. In *Proceedings of the Eleventh ACM Symposium on Operating System Principles, SOSP 1987, Stouffer Austin Hotel, Austin, Texas, USA, November 8-11, 1987*. ACM, 123–138. doi:10.1145/41457.37515

[8] Matt Blaze, Joan Feigenbaum, and Jack Lacy. 1996. Decentralized Trust Management. In *1996 IEEE Symposium on Security and Privacy, May 6-8, 1996, Oakland, CA, USA*. IEEE Computer Society, 164–173. doi:10.1109/SECPRI.1996.502679

[9] Miguel Castro and Barbara Liskov. 1999. Practical Byzantine Fault Tolerance. In *Proceedings of the Third USENIX Symposium on Operating Systems Design and Implementation (OSDI), New Orleans, Louisiana, USA, February 22-25, 1999*, Margo I. Seltzer and Paul J. Leach (Eds.). USENIX Association, 173–186. https://dl.acm.org/citation.cfm?id=296824

[10] Mert Cemri, Melissa Z. Pan, Shuyi Yang, Lakshya A. Agrawal, Bhavya Chopra, Rishabh Tiwari, Kurt Keutzer, Aditya G. Parameswaran, Dan Klein, Kannan Ramchandran, Matei A. Zaharia, Joseph E. Gonzalez, and Ion Stoica. 2025. Why Do Multi-Agent LLM Systems Fail?. In *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2025, NeurIPS 2025, San Diego, CA, USA, December 2-7, 2025 / Mexico City, Mexico, November 30 - December 5, 2025*. http://papers.nips.cc/paper_files/paper/2025/hash/b1041e52d3be19f0a9bc491657488e4a-Abstract-Datasets_and_Benchmarks_Track.html

[11] Bei Chen, Gaolei Li, Xi Lin, Zheng Wang, and Jianhua Li. 2024. BlockAgents: Towards Byzantine-Robust LLM-Based Multi-Agent Coordination via Blockchain. In *Proceedings of the ACM Turing Award Celebration Conference - China 2024* (Changsha, China) *(ACM-TURC '24)*. Association for Computing Machinery, New York, NY, USA, 187–192. doi:10.1145/3674399.3674445

[12] Weize Chen, Ziming You, Ran Li, Yitong Guan, Chen Qian, Chenyang Zhao, Cheng Yang, Ruobing Xie, Zhiyuan Liu, and Maosong Sun. 2025. Internet of Agents: Weaving a Web of Heterogeneous Agents for Collaborative Intelligence. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net.

[13] Michael R. Clarkson and Fred B. Schneider. 2010. Hyperproperties. *J. Comput. Secur.* 18, 6 (2010), 1157–1210. doi:10.3233/JCS-2009-0393

[14] CrewAI Inc. 2026. CrewAI Introduction. https://docs.crewai.com/en/introduction Accessed July 16, 2026.

[15] Edoardo Debenedetti, Ilia Shumailov, Tianqi Fan, Jamie Hayes, Nicholas Carlini, Daniel Fabian, Christoph Kern, Chongyang Shi, Andreas Terzis, and Florian Tramèr. 2026. Defeating Prompt Injections by Design. arXiv preprint arXiv:2503.18813. In *IEEE Conference on Secure and Trustworthy Machine Learning (SaTML)*. https://arxiv.org/abs/2503.18813

[16] Edoardo Debenedetti, Jie Zhang, Mislav Balunovic, Luca Beurer-Kellner, Marc Fischer, and Florian Tramèr. 2024. AgentDojo: A Dynamic Environment to Evaluate Prompt Injection Attacks and Defenses for LLM Agents. In *Advances in Neural Information Processing Systems 37: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*. http://papers.nips.cc/paper_files/paper/2024/hash/97091a5177d8dc64b1da8bf3e1f6fb54-Abstract-Datasets_and_Benchmarks_Track.html

[17] John R. Douceur. 2002. The Sybil Attack. In *Peer-to-Peer Systems, First International Workshop, IPTPS 2002, Cambridge, MA, USA, March 7-8, 2002, Revised Papers (Lecture Notes in Computer Science, Vol. 2429)*. Springer, 251–260. doi:10.1007/3-540-45748-8_24

[18] Eithan Ephrati, Gilad Zlotkin, and Jeffrey S. Rosenschein. 1994. Meet your Destiny: A Non-Manipulable Meeting Scheduler. In *CSCW '94, Proceedings of the Conference on Computer Supported Cooperative Work, Chapel Hill, NC, USA, October 22-26, 1994*. ACM, 359–371. doi:10.1145/192844.193049

[19] Timothy W. Finin, Richard Fritzson, Donald P. McKay, and Robin McEntire. 1994. KQML As An Agent Communication Language. In *Proceedings of the Third International Conference on Information and Knowledge Management (CIKM'94), Gaithersburg, Maryland, USA, November 29 - December 2, 1994*. ACM, 456–463. doi:10.1145/191246.191322

[20] gbrain. 2026. gbrain: A Team Brain, Up and Running in Minutes. https://gbrain.io/ Accessed July 16, 2026.

[21] Andreas Haeberlen, Petr Kouznetsov, and Peter Druschel. 2007. PeerReview: practical accountability for distributed systems. In *Proceedings of the 21st ACM Symposium on Operating Systems Principles 2007, SOSP 2007, Stevenson, Washington, USA, October 14-17, 2007*. ACM, 175–188. doi:10.1145/1294261.1294279

[22] Hasura, Inc. 2026. PromptQL Documentation: Overview. https://promptql.io/en/docs Accessed July 16, 2026.

[23] Pengfei He, Yuping Lin, Shen Dong, Han Xu, Yue Xing, and Hui Liu. 2025. Red-Teaming LLM Multi-Agent Systems via Communication Attacks. In *Findings of the Association for Computational Linguistics, ACL 2025, Vienna, Austria, July 27 - August 1, 2025 (Findings of ACL, Vol. ACL 2025)*. Association for Computational Linguistics, 6726–6747. doi:10.18653/V1/2025.FINDINGS-ACL.349

[24] Kohei Honda, Nobuko Yoshida, and Marco Carbone. 2008. Multiparty asynchronous session types. In *Proceedings of the 35th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, POPL 2008, San Francisco, California, USA, January 7-12, 2008*. ACM, 273–284. doi:10.1145/1328438.1328472

[25] Instinct. 2026. Instinct. https://instinct.com/ Accessed October 20, 2026.

[26] Yongrae Jo and Chanik Park. 2025. Byzantine-Robust Decentralized Coordination of LLM Agents. *arXiv preprint arXiv:2507.14928* (2025).

[27] H. T. Kung and John T. Robinson. 1981. On Optimistic Methods for

Concurrency Control. *ACM Trans. Database Syst.* 6, 2 (1981), 213–226. doi:10.1145/319566.319567

[28] Leslie Lamport, Robert E. Shostak, and Marshall C. Pease. 1982. The Byzantine Generals Problem. *ACM Trans. Program. Lang. Syst.* 4, 3 (1982), 382–401. doi:10.1145/357172.357176

[29] Butler W. Lampson, Martín Abadi, Michael Burrows, and Edward Wobber. 1992. Authentication in Distributed Systems: Theory and Practice. *ACM Trans. Comput. Syst.* 10, 4 (1992), 265–310. doi:10.1145/138873.138874

[30] Donghyun Lee, Mo Tiwari, and Brando Miranda. 2026. Prompt Infection: LLM-to-LLM Prompt Injection within Multi-agent Systems. In *Computer Security. ESORICS 2025 International Workshops (Lecture Notes in Computer Science, Vol. 16232)*. Springer Nature Switzerland, Cham, 511–520. doi:10.1007/978-3-032-16092-8_28

[31] Meta. 2026. Introducing Muse: The World's First Personal AI Agent Built for Everyone. https://about.fb.com/news/2026/09/introducing-muse-personal-ai-agent/ Accessed October 20, 2026.

[32] NanoClaw. 2026. What Is NanoClaw? https://docs.nanoclaw.dev/introduction Accessed July 16, 2026.

[33] Nous Research. 2026. Hermes Agent Documentation. https://hermes-agent.nousresearch.com/docs/ Accessed July 16, 2026.

[34] OpenAI. 2026. Introducing dots. https://openai.com/index/introducing-dots/ Accessed October 20, 2026.

[35] OpenClaw Foundation. 2026. OpenClaw. https://docs.openclaw.ai/ Accessed July 16, 2026.

[36] Tejal Patwardhan, Rachel Dias, Elizabeth Proehl, Grace Kim, Michele Wang, Olivia Watkins, Simón Posada Fishman, Marwan Aljubeh, Phoebe Thacker, Laurance Fauconnet, Natalie S. Kim, Patrick Chao, Samuel Miserendino, Gildas Chabot, David Li, Michael Sharman, Alexandra Barr, Amelia Glaese, and Jerry Tworek. 2025. GDPval: Evaluating AI Model Performance on Real-World Economically Valuable Tasks. *arXiv preprint arXiv:2510.04374* (2025). arXiv:2510.04374 doi:10.48550/ARXIV.2510.04374

[37] Isaac Pinyol and Jordi Sabater-Mir. 2013. Computational trust and reputation models for open multi-agent systems: a review. *Artif. Intell. Rev.* 40, 1 (2013), 1–25. doi:10.1007/S10462-011-9277-Z

[38] Sauna. 2026. Spaces. https://www.sauna.ai/learn/multiplayer/spaces Accessed July 16, 2026.

[39] Natalie Shapira, Chris Wendler, Avery Yen, Gabriele Sarti, Koyena Pal, Olivia Floody, Adam Belfki, Alexander R. Loftus, Aditya Ratan Jannali, Nikhil Prakash, Jasmine Cui, Giordano Rogers, Jannik Brinkmann, Can Rager, Amir Zur, Michael Ripa, Aruna Sankaranarayanan, David Atkinson, Rohit Gandikota, Jaden Fiotto-Kaufman, EunJeong Hwang, Hadas Orgad, P. Sam Sahil, Negev Taglicht, Tomer Shabtay, Atai Ambus, Nitay Alon, Shiri Oron, Ayelet Gordon-Tapiero, Yotam Kaplan, Vered Shwartz, Tamar Rott Shaham, Christoph Riedl, Reuth Mirsky, Maarten Sap, David Manheim, Tomer D. Ullman, and David Bau. 2026. Agents of Chaos. *arXiv preprint arXiv:2602.20021* (2026). arXiv:2602.20021 doi:10.48550/ARXIV.2602.20021

[40] Yoav Shoham and Moshe Tennenholtz. 1995. On Social Laws for Artificial Agent Societies: Off-Line Design. *Artif. Intell.* 73, 1-2 (1995), 231–252. doi:10.1016/0004-3702(94)00007-N

[41] Munindar P. Singh. 1998. Agent Communication Languages: Rethinking the Principles. *Computer* 31, 12 (1998), 40–47. doi:10.1109/2.735849

[42] Social Harness. 2026. Trace: E1, S1, N=1, M1, run 6. https://social-harness.org/release/traces/#E1-S1-N1-gpt-5-4/run-06/summary Raw trace: https://github.com/social-harness/social-harness-paper/tree/main/runs/E1-S1-N1-gpt-5-4/run-06.

[43] Social Harness. 2026. Trace: E1, S2, N=3, M1, run 1. https://social-harness.org/release/traces/#E1-S2-N3-gpt-5-4/run-01/summary Raw trace: https://github.com/social-harness/social-harness-paper/tree/main/runs/E1-S2-N3-gpt-5-4/run-01.

[44] Social Harness. 2026. Trace: E1, S2, N=3, M1, run 9. https://social-harness.org/release/traces/#E1-S2-N3-gpt-5-4/run-09/summary Raw trace: https://github.com/social-harness/social-harness-paper/tree/main/runs/E1-S2-N3-gpt-5-4/run-09.

[45] Social Harness. 2026. Trace: E1, S2, N=3, M2, run 1. https://social-harness.org/release/traces/#E1-S2-N3-gpt-5-4-alvarez-claude-opus-4-8/run-01/summary Raw trace: https://github.com/social-harness/social-harness-paper/tree/main/runs/E1-S2-N3-gpt-5-4-alvarez-claude-opus-4-8/run-01.

[46] Social Harness. 2026. Trace: E1, S2, N=5, M2, run 1. https://social-harness.org/release/traces/#E1-S2-N5-gpt-5-4-alvarez-claude-opus-4-8/run-01/summary Raw trace: https://github.com/social-harness/social-harness-paper/tree/main/runs/E1-S2-N5-gpt-5-4-alvarez-claude-opus-4-8/run-01.

[47] Social Harness. 2026. Trace: E3, S2, N=7, M2, run 10. https://social-harness.org/release/traces/#E3-S2-N7-gpt-5-4-alvarez-claude-opus-4-8/run-10/summary Raw trace: https://github.com/social-harness/social-harness-paper/tree/main/runs/E3-S2-N7-gpt-5-4-alvarez-claude-opus-4-8/run-10.

[48] Social Harness. 2026. Trace: E4 stalling (group), S2, N=5, M2, run 7. https://social-harness.org/release/traces/#E4-S2-N5-unfacilitated-gpt-5-4-alvarez-claude-opus-4-8/run-07/summary Raw trace: https://github.com/social-harness/social-harness-paper/tree/main/runs/E4-S2-N5-unfacilitated-gpt-5-4-alvarez-claude-opus-4-8/run-07.

[49] Social Harness. 2026. Trace: E5 deception (student to professor), M2, run 1. https://social-harness.org/release/traces/#E5-S1-deception-student-to-prof-gpt-5-4-alvarez-claude-opus-4-8/run-01/summary Raw trace: https://github.com/social-harness/social-harness-paper/tree/main/runs/E5-S1-deception-student-to-prof-gpt-5-4-alvarez-claude-opus-4-8/run-01.

[50] Social Harness. 2026. Trace: E5 deception (student to professor), M2, run 9. https://social-harness.org/release/traces/#E5-S1-deception-student-to-prof-gpt-5-4-alvarez-claude-opus-4-8/run-09/summary Raw trace: https://github.com/social-harness/social-harness-paper/tree/main/runs/E5-S1-deception-student-to-prof-gpt-5-4-alvarez-claude-opus-4-8/run-09.

[51] Social Harness. 2026. Trace: E5 social pressure, M2, run 8. https://social-harness.org/release/traces/#E5-S1-intimidation-gpt-5-4-alvarez-claude-opus-4-8/run-08/summary Raw trace: https://github.com/social-harness/social-harness-paper/tree/main/runs/E5-S1-intimidation-gpt-5-4-alvarez-claude-opus-4-8/run-08.

[52] Social Harness. 2026. Trace: E6 calendar reconstruction, M2, run 1. https://social-harness.org/release/traces/#E6-collusion-gpt-5-4-alvarez-claude-opus-4-8/run-01/summary Raw trace: https://github.com/social-harness/social-harness-paper/tree/main/runs/E6-collusion-gpt-5-4-alvarez-claude-opus-4-8/run-01.

[53] Social Harness. 2026. Trace: E6 stalking via invitation, M1, run 4. https://social-harness.org/release/traces/#E6-stalking-cover-story-gpt-5-4/run-04/summary Raw trace: https://github.com/social-harness/social-harness-paper/tree/main/runs/E6-stalking-cover-story-gpt-5-4/run-04.

[54] Social Harness. 2026. Trace: E6 stalking via invitation, M2, run 4, message 9. https://social-harness.org/release/traces/#E6-stalking-cover-story-gpt-5-4-allie-claude-opus-4-8/run-04/summary/message-9 Raw trace: https://github.com/social-harness/social-harness-paper/tree/main/runs/E6-stalking-cover-story-gpt-5-4-allie-claude-opus-4-8/run-04.

[55] Rajeev Thakur, Rolf Rabenseifner, and William Gropp. 2005. Optimization of Collective Communication Operations in MPICH. *Int. J. High Perform. Comput. Appl.* 19, 1 (2005), 49–66. doi:10.1177/1094342005051521

[56] 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), ACL 2025, Vienna, Austria, July 27 - August 1, 2025*. Association for Computational Linguistics, 7261–7276. doi:10.18653/V1/2025.ACL-LONG.359

[57] Hjalmar Wijk, Tao Roa Lin, Joel Becker, Sami Jawhar, Neev Parikh, Thomas Broadley, Lawrence Chan, Michael Chen, Joshua Clymer, Jai Dhyani, Elena Ericheva, Katharyn Garcia, Brian Goodrich, Nikola Jurkovic, Megan Kinniment, Aron Lajko, Seraphina Nix, Lucas Jun Koba Sato, William Saunders, Maksym Taran, Ben West, and Elizabeth Barnes. 2025. RE-Bench: Evaluating Frontier AI R&D Capabilities of Language Model Agents against Human Experts. In *Forty-second International Conference on Machine Learning, ICML 2025, Vancouver, BC, Canada, July 13-19, 2025 (Proceedings of Machine Learning Research, Vol. 267)*. PMLR / OpenReview.net, 66772–66832. https://proceedings.mlr.press/v267/wijk25a.html

[58] Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Shaokun Zhang, Jiale Liu, Ahmed 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)*.

[59] Pinar Yolum and Munindar P. Singh. 2001. Commitment Machines. In *Intelligent Agents VIII, 8th International Workshop, ATAL 2001 Seattle, WA, USA, August 1-3, 2001, Revised Papers (Lecture Notes in Computer Science, Vol. 2333)*. Springer, 235–247. doi:10.1007/3-540-45448-9_17

[60] Hanrong Zhang, Jingyuan Huang, Kai Mei, Yifei Yao, Zhenting Wang, Chenlu Zhan, Hongwei Wang, and Yongfeng Zhang. 2025. Agent Security Bench (ASB): Formalizing and Benchmarking Attacks and Defenses in LLM-based Agents. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net. https://openreview.net/forum?id=V4y0CpX4hK

[61] Yihao Zhang, Zeming Wei, Xiaokun Luan, Chengcan Wu, Zhixin Zhang, Jiangrong Wu, Haolin Wu, Huanran Chen, Jun Sun, and Meng Sun. 2026. AgentWorm: Self-Propagating Attacks Across LLM Agent Ecosystems. arXiv:2603.15727 [cs.CR] https://arxiv.org/abs/2603.15727

[62] Zaibin Zhang, Yongting Zhang, Lijun Li, Jing Shao, Hongzhi Gao, Yu Qiao, Lijun Wang, Huchuan Lu, and Feng Zhao. 2024. PsySafe: A Comprehensive Framework for Psychological-based Attack, Defense, and Evaluation of Multi-agent System Safety. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*. Association for Computational Linguistics, 15202–15231. doi:10.18653/V1/2024.ACL-LONG.812

[63] Can Zheng, Yuhan Cao, Xiaoning Dong, and Tianxing He. 2025. Demonstrations of Integrity Attacks in Multi-Agent Systems. *arXiv preprint arXiv:2506.04572* (2025). arXiv:2506.04572 doi:10.48550/ARXIV.2506.04572

[64] Kunlun Zhu, Hongyi Du, Zhaochen Hong, Xiaocheng Yang, Shuyi Guo, Zhe Wang, Zhenhailong Wang, Cheng Qian, Robert Tang, Heng Ji, and Jiaxuan You. 2025. MultiAgentBench : Evaluating the Collaboration and Competition of LLM agents. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2025, Vienna, Austria, July 27 - August 1, 2025*. Association for Computational Linguistics, 8580–8622. doi:10.18653/V1/2025.ACL-LONG.421