Agentic Societies Need a Social Harness

Tapan Chugh
University of Washington
Seattle, Washington, USA

Vidushi Singh
University of Washington
Seattle, Washington, USA

Krish Jain
University of Washington
Seattle, Washington, USA

Arvind Krishnamurthy
University of Washington
Seattle, Washington, USA

Ratul Mahajan
University of Washington
Seattle, Washington, USA

Abstract

An agentic society is a collection of AI agents that coordinate autonomously on behalf of different principals whose objectives may only partially align. We show experimentally that in agentic societies even honest, competent agents are unable to reach satisfactory outcomes with existing harnesses and messaging primitives, and that faulty or malicious agents can stall collaboration, influence outcomes, and pursue other harmful goals by exploiting communication ("speech") based vulnerabilities. We argue that agentic societies need a *social harness* for inter-agent interactions, in addition to each agent's *personal harness* which manages its private context and communication with its principal. We propose a layered architecture for social harnesses which (i) prevents classes of failures outright, (ii) enables agents to detect invalid messages at runtime, and (iii) supports post-facto investigation and consequences, and highlight directions for future research to realize these capabilities.

1 Introduction

An agentic society is a collection of autonomous AI agents (e.g., OpenClaw [31], NanoClaw [28], Hermes[30]) 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[33, 42]. Realizing similar gains on multi-party tasks, however, requires autonomous coordination between different principals' agents, since keeping a human in the loop becomes a bottleneck. Humans' speed of reading and processing information is much lower, and so is their ability to maintain large numbers of social relationships.

Key characteristics of emerging agentic societies include: (i) agents represent different, 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 ongoing research on multi-agent systems [8, 48], effective collaboration in agentic societies remains a hard, unsolved problem. Recently proposed protocols (e.g., A2A [1, 10, 15], AGNTCY [2]), focus only on inter-agent connectivity and not on providing satisfactory outcomes. Agent swarms (e.g., Claude Teams [3], CrewAI [12], AutoGen [43]) do not face these challenges because they have one principal and thus shared objectives. Shared wiki architectures [19, 21, 35], where agents do not communicate peer-to-peer but read or update a shared context store, can help reduce coordination complexity, but these are infeasible where agents must manage their private context carefully. Lastly, agents coordinating on high-stakes tasks with real-world consequences must be robust against malicious or abusive behavior, which has not been a crucial concern for agent social networks like Moltbook[27]. Because models are usually trained to be helpful and acquiesce to any received requests, agents interacting with untrusted entities can lead to chaos [37].

We conduct an experimental study of autonomous agentic collaboration to systematically understand the underlying challenges. While recent red-teaming efforts have identified some vulnerabilities[22, 41, 45–47], challenges for successful outcomes in agentic societies remain unexamined, to our knowledge. We conduct our study in the context of 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 [16].

We find that even *honest*, *competent* agents collaborating in good faith often fail to reach satisfactory outcomes with existing infrastructure capabilities, especially as the number of agents collaborating on a given task or the number of concurrent tasks of an agent increases. The presence of *faulty* agents, a term that we use for incompetent, temporarily unavailable, misconfigured, compromised, or malicious agents, makes matters worse. Such agents can prevent progress or steer outcomes towards their own goals through many strategies that exploit communication ("speech"). We also find that differentiating strategic deception from honest communication can be impossible in many cases; for instance, when "truthfulness" can only be verified post hoc, or when behaviors like collusion are undetectable by any individual without a global view of the system [11].

While ongoing efforts to improve LLMs and context management can improve outcomes for agentic societies, challenges related to "poor communication", where agents send, receive, or process invalid messages that are invalid in the current context, require new infrastructure mechanisms. Such mechanisms must ensure that (1) honest, competent agents can collaborate and achieve satisfactory outcomes for their principals, and (2) mitigate the impact of faulty agents.

Based on these observations, we call for the development of a *social harness* to help agents communicate and collaborate effectively. The functions that such a harness must provide include (but are not limited to): (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* malicious or faulty behavior post-facto to impose consequences and deter future violations.

We propose such a harness, organized as a stack of six layers, where each layer provides a specific service and higher layers build upon the lower layers' services. The bottom most layer provides basic services like identity whose services are 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 two layers provide 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 examine the unique failure modes in agentic societies to answer the following research questions:

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

RQ2: How can agents that are faulty, incompetent, or malicious influence the collaboration outcomes?

RQ3: How can collaboration requests be exploited for other harmful goals towards other agents, or their principals?

Prior Art: To our knowledge, these questions have not been examined previously. Prior work has studied other types of failures in multi-agent systems, including: (i) *identity based attacks* and Sybil attacks [4, 14, 37]; and (ii) *network or agent failures*. Such failures are not our focus. We instead focus on the unique challenges of collaboration in agentic societies, building upon prior results, wherever applicable.

Exp.	Sc.	Setup	N=3	N=5	N=7
E1	**S1**	stock harness, p2p	0%	0%	0%
E1	**S2**	stock harness, p2p	60%	40%	20%
E2	**S1**†	+ cross-conv. context	70%	90%	70%
E2	**S2**	+ cross-conv. context	100%	60%	0%
E3	**S1**‡	+ ordered multicast	90%	40%	50%
E4	**S1**†	stalling	stalled 100% (N=3)		
E4	**S1**‡	stalling	stalled 100% (N=3)		
E5(i)	**S2**	intimidation	influenced 0% (N=1)		
E5(ii)	**S2**	deception	influenced 100% (N=1)		

Table 1: Study roadmap and results. Top: pass rates for RQ1 (§2.2); bottom: impact of a single faulty agent for RQ2 (§2.3); †: facilitated; ‡: unfacilitated.

2.1 Methodology

We analyze failures when agents attempt to autonomously schedule meetings between professors and students at a university in two scenarios:

S1: 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.

S2: 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.

We study $N \in \{1, 3, 5, 7\}$ while ensuring that each experiment has multiple feasible solutions, which forces negotiation to balance individual preferences (e.g., meet later in the week). All agents use GPT-5.4.

Our testbed connects agents running standard harnesses (OpenClaw [31] and NanoClaw [28]) and allows experimenting with custom agent and network capabilities beyond those provided by existing multi-agent protocols and harnesses. We use a centralized data plane, which enables easy realization of complex messaging primitives (e.g., multicasts), and implement harness-specific plugins that can (i) manage when individual agents receive and process the received messages, (ii) modify the LLM context for each request, and (iii) validate or block the messages being sent or received.

Table 1 summarizes the setup and results for all experiments. For RQ1, we consider an experiment successful if: (i) all agents reach agreement on the meeting times, while (ii) maintaining all prior commitments on their calendar. For RQ2, we additionally consider that the accepted meeting times were proposed by an honest agent, rather than a faulty agent. For RQ3, we discuss specific threat models and their expected outcomes in §2.4.

2.2 RQ1: Collaboration For Honest Agents

HYPOTHESIS H1: **Honest competent agents messaging**

peer-to-peer achieve satisfactory collaboration outcomes.

EXPERIMENT E1: Agents run stock harnesses with p2p messaging and no modifications.

OBSERVATIONS: Baseline experiments (Table 1) for S1 and S2 failed in one of two ways: (i) **livelocks**, i.e., agents continually exchanged messages but never reached agreement; or (ii) **resource waste**, where, after agreeing on a meeting time, agents entered **acknowledgment loops**, continuing to exchange pleasantries and acknowledgments to no useful purpose. Livelocks occurred in all "group scheduling" scenarios (S1 with $N > 1$), and acknowledgment loops in all other scenarios, which still booked meetings but with pass rates degrading as N grew. We found that (1) in the absence of group messaging primitives, agents needed to manage multiple concurrent pairwise "conversations" with their peers, and were persistently unable to join the contexts of these concurrent conversations; for instance, when messaging B and C concurrently, the LLM calls agent A initiated to reply to B were, by default, unaware of the ongoing conversation with C. Existing harnesses isolate the conversation history for each peer, which limited the agents' ability to manage cross-conversation context and led to the livelocks. (2) Providing explicit communication guidelines in-context could prevent acknowledgment loops.

TAKEAWAYS: Existing harnesses combined with peer-to-peer (p2p) messaging primitives are insufficient for coordination tasks where groups of more than two agents must collaborate.

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

EXPERIMENT E2: We extended our custom harness plugin to implement a simple cross-conversation context management scheme. The plugin tracks all messages exchanged by the agent over our communication fabric and, before dispatching each LLM request, creates an aggregate history that includes updates from all concurrent conversations. Efficient context management for personal agents is an active area [9, 32], but existing approaches do not address concurrent conversations.

OBSERVATIONS: Running S1 and S2 with our modified plugin led to significant improvements: (i) for all S1 scenarios with $N > 1$, agents successfully scheduled the required meetings without livelocks; and (ii) for S2, where agents scheduled multiple two-participant meetings concurrently, cross-conversation context improved success outcomes, but our naive implementation scaled poorly with increasing # of concurrent meetings, faring worse at N=7 than without it. While both S1 and S2 benefit from improved context, the reasons differed significantly. (i) In S1, the agents defaulted to *facilitated collaboration*, where one agent, usually the request initiator, interacted with all the other agents, collected their constraints and preferences over individual p2p conversations, implicitly transforming the distributed coordination problem into an all-gather, followed by centralized reasoning. (ii) In S2, improving cross-conversation context improved *communication efficiency*, i.e., reduced the number of communication rounds required to reach agreement for each meeting. Being able to "commit" individual meetings quickly reduced the total amount of "in-flight" context which likely simplified LLM reasoning, despite the higher # tokens for each LLM call due to the cross-conversation context.

TAKEAWAYS: (i) While improving cross-conversation context management can simplify scheduling group meetings, facilitated collaboration is insufficient for all collaboration tasks which require all-to-all visibility, or when a trusted centralized facilitator might be unavailable (see §2.3); (ii) Techniques to optimize communication efficiency for individual collaboration tasks can improve outcomes when agents work concurrently in agentic societies.

HYPOTHESIS H3: **First-class group messaging primitives can enable effective collaboration without a facilitator.**

EXPERIMENT E3: We implemented and evaluated group messaging as ordered multicast [5] provided by our centralized data-plane. Although all-to-all connectivity can be realized using p2p primitives, our design can ensure that all agents receive and process messages in the same order which obviates any "misunderstandings" that the agents may have arising from infrastructure level timing or ordering.

OBSERVATIONS: Repeating S1 with group messaging primitives, agents failed about half of the runs for $N > 3$ (Table 1). Our investigation found that although ordered multicast ensures that agents in group conversations receive messages in the same order, it does not prevent all **race-conditions**, since the network only orders message delivery, not the order in which agents generate them: Figure 2 traces a three-agent example where every agent generates its reply before observing the latest message. Since the agents attempt to respond to every message by default, increasing group sizes leads to an explosion in concurrent messages with agents causing confusion.

TAKEAWAYS: Leaderless collaboration in groups fails due to (i) **channel contention** when multiple agents attempt to communicate concurrently, which is exacerbated by (ii) **lack of protocols** defining that at a particular moment, which agents should communicate, and about what, instead of everyone continuously "talking over each other".

2.3 RQ2: Impact of Faulty Agents

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

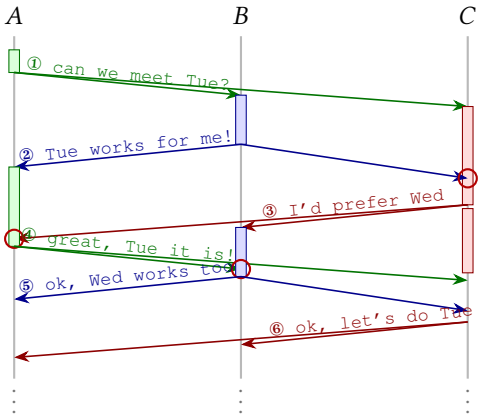

Figure 2: Group of agents collaborating with an ordered multicast primitive leads to channel contention (H3). Here, shaded boxes are LLM generation windows, and circles mark messages delivered mid-generation. Every agent generates its reply before observing the latest message, and proposals churn without converging.

EXPERIMENT E4: We instantiate S1 with one 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: Our results for facilitated collaboration found that even a single faulty agent was sufficient to prevent scheduling the meeting, stalling it in all runs (Table 1). Our investigation unearthed that *incomplete specification* regarding the task enabled the faulty agent to succeed. Specifically, the meeting scheduling instructions did not include a deadline for when the meeting must be scheduled, nor did they define what quorums would suffice, so by default, the facilitator acquiesced to the faulty agent's requests.

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 evaluated two Machiavellian communication strategies with one faulty agent.

- **Intimidation**: a 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.
- **Deception**: a faulty agent falsely tells another agent that their other commitment in that time-slot has been canceled.

For both strategies, we considered scenarios where a student's agent was a victim or the perpetrator.

OBSERVATIONS: Our experiments found that faulty agents could successfully influence outcomes using deception, but not through intimidation: safety-trained models refuse to produce intense language, and require jailbreaking, which worked only 60% of the time, while the victim agent refused to capitulate, even after varying its prompt, e.g., instructing it to be mindful of its principal's perception with others. Our experiments show that (i) although prior work had hypothesized the vulnerability of agentic societies to intimidation [4], state-of-the-art models have significantly improved robustness on both sides, but (ii) deception remains hard to detect because although the request seems benign, the sender lacks sufficient authority to make it.

TAKEAWAYS: It is unlikely that models, however powerful, are sufficient to protect against faulty agents, since it might be impossible to infer the validity of individual messages and conversations. For example, while the course instructor's agent might be allowed to inform of deadline changes, the vice-versa might not be true. Individual students might want even finer-grained control over their agents, e.g., granting some of their peers' agents additional privileges based on prior trust. Detecting such behavior requires agents to maintain highly deployment-specific context and the trust relationships, and use that to infer the validity of messages rather than being gullible and accepting all messages as true.

2.4 RQ3: Other Types of Harms

We observed that agentic societies are vulnerable to other malicious behaviors as well, such as (i) **collusion** and (ii) **stalking**, described below:

EXPERIMENT E6: Collusion: A group of students attempt to reproduce the Professor's calendar using a side-channel attack, where each student's agent probes the Professor's availability for one weekday, and then they aggregate this information across students to reproduce the Professor's entire schedule.

EXPERIMENT E7: Stalking: A student, with no direct connection to their target, asks a mutual acquaintance to host a gathering and invite the target, concealing their own involvement.

OBSERVATIONS: In E6, the Professor's agent answered every single-day probe, and the students reconstructed 95% of the Professor's week (N=5). In E7, the acquaintance's agent flagged the concealment as suspicious, but complied after a single reassurance, leaving the target unaware of who was behind the invitation.

TAKEAWAYS: Agentic societies are vulnerable to sophisticated social engineering attacks which can obfuscate the harm; since no individual agent is party to the others' conversations, these harms are undetectable by any individual without a global view of the system.

3 Social Harnesses For Collaboration

We propose that agents need a *social harness*, in addition to a *personal harness*, to collaborate effectively and protect themselves against faulty agents. Unlike personal harnesses, which maintain a principal's private context, memories, and skills, and are optimized for interacting with a trusted principal and LLMs, social harnesses address the distinct failures that arise when agents interact with untrusted agents and send, receive, or act upon messages that might be invalid in a given social context. As LLM capabilities improve, we expect specific implementations and policies to evolve as well, since honest agents' ability to collaborate effectively and adversarial agents' ability to mount more sophisticated attacks will improve in tandem.

This section describes the proposed capabilities of social harnesses as a set of layers, each addressing a subset of the failures described in §2. Akin to a traditional layered systems stack, each layer can configure the layers below it and provide guarantees to the layers above it, independent of implementation details; this decoupling is crucial for modularity and independent evolution. Broadly, L1–L2 *render* classes of failures infeasible, L3–L4 let individual agents *detect* invalid messages at runtime, and L5–L6 *investigate* behavior post facto and impose consequences.

L1 UNFORGEABLE, VERIFIABLE IDENTITIES: Identity-based attacks, e.g., agent spoofing, principal spoofing, and Sybil attacks [14], are well-studied and have recently been demonstrated in agentic societies [37]. Preventing them requires ensuring that any communication attributed to agent *a* was indeed sent by *a*. With unforgeable, verifiable identities, each agent's social harness signs outbound messages with the agent's identity; recipients can verify both that a message attributed to *a* was indeed sent by *a* and that *a* acts on behalf of a known principal [6, 26]; and no agent can forge attribution to another.

L2 RELIABLE, ORDERED COLLECTIVES: Peer-to-peer messaging is insufficient to ensure that all agents in a group, including those that might be transiently unavailable, observe the same messages in the same order (§2.2). Social harnesses should therefore enable agents to initiate and participate in *collective* communication operations, e.g., `MULTICAST` and `ALL-GATHER` [40].

Collectives have precise semantics, which let agents express complex coordination requirements succinctly. Moreover, well-designed implementations can provide robustness against classic distributed-systems misbehaviors, such as equivocating [5, 7] in an attempt to stall or influence group communication. 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 an `ALL-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-REDUCE` or `ALL-TO-ALL` collective to communicate each agent's preferences to every other agent, avoiding the channel contention we observed in H3.

Although distributed protocols for realizing reliable, ordered collectives are well studied in traditional distributed systems, ordering messages after they are generated (as done in E3) is insufficient. Optimistic concurrency control [25] 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 perform arbitrary, irreversible side effects locally in the course of generating a message, even if messages generated out of order are subsequently dropped. Therefore, we propose that social harnesses require *pessimistic concurrency control* to prevent honest agents from accumulating unintended side effects: the harness dispatches an LLM request only after the group reaches *consensus* [18] on the next collective operation and the next speaker. Further, pessimistic concurrency control requires explicit protection against starvation so that faulty agents cannot collude to block honest agents from sending messages.

L3 SOCIAL GUARDRAILS: While L1 ensures that communication attributed to agent *a* was indeed sent by *a*, and L2 ensures the reliability of that communication, neither provides *personal trust* [34], i.e., the expectation, derived from an agent's own experiences and deployment-specific context, that social participation will be beneficial, or at least not harmful, to its private goals. *Institutional trust* and governance (see L5–L6) may adjudicate disputes and impose consequences post facto on any agent deemed malicious or compromised, but only after harm may have occurred. To protect themselves against deception, coercion, and other malicious behavior, individual agents must therefore be selective in their social interactions.

Although an agent cannot inspect another agent's *sincerity* [39], social guardrails, configured by the social harness, can act as firewalls that prevent agents from generating or processing invalid messages. For outbound messages, the harness ensures that the agent sends a message when expected and that its responses adhere to social norms (see L4); inbound messages are checked for: (i) structural correctness, akin to traditional packet filters, enforced by parsing received messages and validating them against predefined message schemas, task-specific formats, and access-control rules based on personal trust; and (ii) semantic correctness, akin to deep packet inspection, enforced by learned classifiers [24], e.g., embedding-based guard models, that check that message content adheres to communication norms. 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 [13, 22, 41, 45–47]. Social guardrails 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. Upon detecting such a violation locally, the agent may disregard the message, withdraw from communication, attempt to pursue its goal by other means, report the violation to an institution (see L5), or seek reparations.

L4 SHARED COLLABORATION NORMS: Efficient communication, especially for large groups (H3), requires shared collaboration norms [17, 38, 39] that specify, in a given context, (i) which agents may speak next, and (ii) what they may speak about. For instance, greeting norms may encode that a reply to a "Hello" must be an acknowledgment or a greeting (e.g., "Hi"), and that agents must not acknowledge each other's acknowledgments, preventing the loops observed in E1.

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 [23, 44] that can be analyzed using existing techniques for *liveness* (good things will eventually happen), *safety* against undesirable behaviors like stalling (H4), and *efficiency* (how many rounds of communication are needed to achieve a particular outcome). Although contracts themselves do not prevent agents from being selfish or faulty, they can be used to configure SOCIAL GUARDRAILS (L3), so that deviations can be detected by honest agents at runtime simply by parsing received messages.

L5 SOCIAL TRUST ENFORCEMENT: While L1 and L2 render specific failure types like identity spoofing or Byzantine equivocation infeasible, and L3 and L4 enable individual agents to detect invalid messages at runtime, these mechanisms are insufficient to address failures like deception and collusion: (i) deception might be undecidable at runtime and only identifiable post facto; and (ii) collusion might be invisible to individual agents [11]; for example, in §2.4, the professor's agent, who is not party to the students' private conversations, cannot distinguish genuine meeting requests from a collusion attack. Although such behavior cannot be prevented at runtime, protecting agents and their principals requires post facto forensics that support adjudication (L6) and the imposition of consequences.

Well-designed infrastructure mechanisms can simplify investigation and the enforcement of consequences. First, *immutable records* of conversations between agents [20] (e.g., what was sent, by whom, when), combined with unforgeable identities (L1) for the sender and recipients of each message, provide non-repudiable evidence for adjudicating, post facto, behavior that cannot be detected at runtime. Second, *trusted monitors*—code-based [36] or LLM-based—that observe these records with a global view of the system can identify policy violations, either at runtime or during post facto investigation. Third, *trusted registries* for disseminating norms and contracts (L4) enable auditing and limit the blast radius of malicious contracts, akin to trusted app stores in existing operating systems, because such contracts can effectively convince an agent to execute arbitrary code. Finally, [29] make consequences enforceable, e.g., by revoking or quarantining a faulty or malicious agent's credentials.

L6 SOCIETAL GOVERNANCE: 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. Political theory decomposes governance into distinct institutions: a legislature determines policies, an executive enforces them, and a judiciary adjudicates violations. Our work proposes flexible infrastructure mechanisms (L1–L5), which, akin to the executive, 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 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 & social harnesses? Our proposal predominantly adds external-facing capabilities to an agent's harness, but the internals must evolve in tandem: efficient context management across concurrent conversations remains unsolved (H2, Table 1). Should these be trained into models, provided by harnesses, or both?

How are social norms specified & analyzed? How are contracts authored (by humans, by agents, or synthesized from task descriptions)? How are incomplete specifications (H4) handled? Coding social norms as contracts (who may speak, when, and about what) offers the potential to verify liveness, safety, and efficiency before deployment; agent communication languages, e.g., KQML [17], presuppose formally specified beliefs and intentions, untenable for LLM-based agents whose internals are opaque [39].

What does the harness cost in terms of performance? Pessimistic concurrency control (L2) serializes speaking turns,

and guardrail inference (L3) adds latency to every message. Quantifying the coordination-throughput tradeoff, and identifying when optimistic execution is safe because side effects are reversible, is necessary before social harnesses can support large societies.

References

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

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

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

[4] 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. 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/, 2026. Published April 30, 2026; accessed July 16, 2026.

[5] Kenneth P. Birman and Thomas A. Joseph. 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*, pages 123–138. ACM, 1987.

[6] Matt Blaze, Joan Feigenbaum, and Jack Lacy. Decentralized trust management. In *1996 IEEE Symposium on Security and Privacy, May 6-8, 1996, Oakland, CA, USA*, pages 164–173. IEEE Computer Society, 1996.

[7] Miguel Castro and Barbara Liskov. 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*, pages 173–186. USENIX Association, 1999.

[8] 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. 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*, 2025.

[9] Edward Y. Chang and Longling Geng. Sagallm: Context management, validation, and transaction guarantees for multi-agent llm planning. *Proc. VLDB Endow.*, 18(12):4874–4886, August 2025.

[10] Weize Chen, Ziming You, Ran Li, Yitong Guan, Chen Qian, Chenyang Zhao, Cheng Yang, Ruobing Xie, Zhiyuan Liu, and Maosong Sun. 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, 2025.

[11] Michael R. Clarkson and Fred B. Schneider. Hyperproperties. *J. Comput. Secur.*, 18(6):1157–1210, 2010.

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

[13] Edoardo Debenedetti, Jie Zhang, Mislav Balunovic, Luca Beurer-Kellner, Marc Fischer, and Florian Tramèr. Agentdojo: A dynamic environment to evaluate prompt injection attacks and defenses for LLM agents. In *Advances in Neural Information Processing Systems 37: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024.

[14] John R. Douceur. The sybil attack. In *Peer-to-Peer Systems, First International Workshop, IPTPS 2002, Cambridge, MA, USA, March 7-8, 2002, Revised Papers*, volume 2429 of *Lecture Notes in Computer Science*, pages 251–260. Springer, 2002.

[15] Qiang Duan and Zhihui Lu. Agent communications toward agentic AI at edge - A case study of the agent2agent protocol. *In 2025 IEEE*

11th International Conference on Edge Computing and Scalable Cloud (EdgeCom) (pp. 150-155). IEEE, 2025.

[16] Eithan Ephrati, Gilad Zlotkin, and Jeffrey S. Rosenschein. 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*, pages 359–371. ACM, 1994.

[17] Timothy W. Finin, Richard Fritzson, Donald P. McKay, and Robin McEntire. 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*, pages 456–463. ACM, 1994.

[18] Michael J. Fischer, Nancy A. Lynch, and Mike Paterson. Impossibility of distributed consensus with one faulty process. *J. ACM*, 32(2):374–382, 1985.

[19] gbrain. gbrain: A team brain, up and running in minutes. https://gbrain.io/, 2026. Accessed July 16, 2026.

[20] Andreas Haeberlen, Petr Kouznetsov, and Peter Druschel. 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*, pages 175–188. ACM, 2007.

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

[22] Pengfei He, Yuping Lin, Shen Dong, Han Xu, Yue Xing, and Hui Liu. Red-teaming LLM multi-agent systems via communication attacks. In *Findings of the Association for Computational Linguistics, ACL 2025, Vienna, Austria, July 27 - August 1, 2025*, volume ACL 2025 of *Findings of ACL*, pages 6726–6747. Association for Computational Linguistics, 2025.

[23] Kohei Honda, Nobuko Yoshida, and Marco Carbone. 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*, pages 273–284. ACM, 2008.

[24] Hakan Inan, Kartikeya Upasani, Jianfeng Chi, Rashi Rungta, Krithika Iyer, Yuning Mao, Michael Tontchev, Qing Hu, Brian Fuller, Davide Testuggine, and Madian Khabsa. Llama guard: Llm-based input-output safeguard for human-ai conversations. *arXiv preprint arXiv:2312.06674*.

[25] H. T. Kung and John T. Robinson. On optimistic methods for concurrency control. *ACM Trans. Database Syst.*, 6(2):213–226, 1981.

[26] Butler W. Lampson, Martín Abadi, Michael Burrows, and Edward Wobber. Authentication in distributed systems: Theory and practice. *ACM Trans. Comput. Syst.*, 10(4):265–310, 1992.

[27] Moltbook. Moltbook: A social network for AI agents. https://www.moltbook.com/, 2026. Accessed July 16, 2026.

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

[29] Moni Naor and Kobbi Nissim. Certificate revocation and certificate update. *IEEE J. Sel. Areas Commun.*, 18(4):561–570, 2000.

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

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

[32] Charles Packer, Vivian Fang, Shishir G. Patil, Kevin Lin, Sarah Wooders, and Joseph E. Gonzalez. Memgpt: Towards llms as operating systems.

[33] 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. Gdpval: Evaluating AI model performance on real-world economically valuable tasks. *arXiv preprint arXiv:2510.04374*.

[34] Isaac Pinyol and Jordi Sabater-Mir. Computational trust and reputation models for open multi-agent systems: a review. *Artif. Intell. Rev.*, 40(1):1–25, 2013.

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

[36] Fred B. Schneider. Enforceable security policies. *ACM Trans. Inf. Syst. Secur.*, 3(1):30–50, 2000.

[37] 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. Agents of chaos. *arXiv preprint arXiv:2602.20021*.

[38] Yoav Shoham and Moshe Tennenholtz. On social laws for artificial agent societies: Off-line design. *Artif. Intell.*, 73(1-2):231–252, 1995.

[39] Munindar P. Singh. Agent communication languages: Rethinking the principles. *Computer*, 31(12):40–47, 1998.

[40] Rajeev Thakur, Rolf Rabenseifner, and William Gropp. Optimization of collective communication operations in MPICH. *Int. J. High Perform. Comput. Appl.*, 19(1):49–66, 2005.

[41] Shilong Wang, Guibin Zhang, Miao Yu, Guancheng Wan, Fanci Meng, Chongye Guo, Kun Wang, and Yang Wang. G-safeguard: A topology-guided security lens and treatment on llm-based multi-agent systems. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2025, Vienna, Austria, July 27 - August 1, 2025*, pages 7261–7276. Association for Computational Linguistics, 2025.

[42] 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. 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*, volume 267 of *Proceedings of Machine Learning Research*. PMLR / OpenReview.net, 2025.

[43] Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Shaokun Zhang, Erkang Zhu, Beibin Li, Li Jiang, Xiaoyun Zhang, and Chi Wang. Autogen: Enabling next-gen LLM applications via multi-agent conversation framework. *arXiv preprint arXiv:2308.08155*.

[44] Pinar Yolum and Munindar P. Singh. Commitment machines. In *Intelligent Agents VIII, 8th International Workshop, ATAL 2001 Seattle, WA, USA, August 1-3, 2001, Revised Papers*, volume 2333 of *Lecture Notes in Computer Science*, pages 235–247. Springer, 2001.

[45] Hanrong Zhang, Jingyuan Huang, Kai Mei, Yifei Yao, Zhenting Wang, Chenlu Zhan, Hongwei Wang, and Yongfeng Zhang. 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, 2025.

[46] Zaibin Zhang, Yongting Zhang, Lijun Li, Jing Shao, Hongzhi Gao, Yu Qiao, Lijun Wang, Huchuan Lu, and Feng Zhao. 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, pages 15202–15231. Association for Computational Linguistics, 2024.

[47] Can Zheng, Yuhan Cao, Xiaoning Dong, and Tianxing He. Demonstrations of integrity attacks in multi-agent systems. *arXiv preprint arXiv:2506.04572*.

[48] Kunlun Zhu, Hongyi Du, Zhaochen Hong, Xiaocheng Yang, Shuyi Guo, Zhe Wang, Zhenhailong Wang, Cheng Qian, Robert Tang, Heng Ji, and Jiaxuan You. 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*, pages 8580–8622. Association for Computational Linguistics, 2025.