INTERNATIONAL JOURNAL OF INFORMATION AND COMMUNICATION TECHNOLOGY TRENDS

Vol. 06 No. 01 (2026)

e. ISSN: 2409-5222

https://ijictt.ictt.info

Consensus-Guided Redundant Memory Isolation for Poisoning Mitigation in Collaborative Agent Frameworks

Michael J. Smith1, David R. Thompson2*, Emily K. Brown3

School of Computer Science, McGill University, Montreal, QC H3A 0E9, Canada

Corresponding author: david.thompson@university.edu

Abstract : *Memory poisoning can destabilize collaborative decision-making when agents rely on shared state synchronization. This study introduces a consensus-guided redundant memory isolation mechanism to prevent cross-agent contamination. Agents maintain parallel memory replicas and compute consensus confidence scores using weighted majority voting across trusted peers. Memory segments with confidence below 0.65 are temporarily isolated and revalidated before reintegration. Evaluation was conducted on a distributed task allocation benchmark with 60 agents under varying poisoning rates (5%–25%). The isolation mechanism reduced collective performance degradation by 42.1% and limited contamination spread to fewer than 18% of agents, compared with 61% under standard synchronization. Communication overhead increased by 12.7%, remaining within acceptable system bounds. The approach strengthens robustness while maintaining scalability in collaborative environments.*

Keywords: *Multi-agent collaboration; memory isolation; consensus mechanisms; adversarial resilience; distributed systems security; poisoning mitigation*

1. INTRODUCTION

Large language model (LLM)-based multi-agent systems have recently become an important paradigm for collaborative reasoning, planning, and task execution, especially in settings that require iterative interaction, role specialization, and distributed problem solving. In these systems, agents exchange messages, share intermediate outputs, and access common memory structures during the reasoning process. Shared memory typically stores prior messages, partial solutions, external retrieval results, and task context, allowing agents to preserve continuity across multiple interaction steps and coordinate their decisions more effectively. This mechanism improves reasoning consistency and often leads to higher task efficiency in complex collaborative workflows [1]. At the same time, the increasing reliance on shared memory has introduced a new security concern. Once incorrect, manipulated, or malicious information enters the memory store, it can be retrieved again in later reasoning cycles and reused by multiple agents. As a result, corrupted content may not remain local to one agent, but instead propagate through interaction and continue to influence system behavior over time [2,3]. Recent studies have shown that poisoned memory can remain active across multiple rounds of collaboration and may continue to affect downstream reasoning even after the original attack has ended. This risk becomes more serious in collaborative agent environments because memory contamination is closely coupled with message passing and collective decision making. Existing work has shown that adversarial content can be inserted into long-term memory, external knowledge bases, or agent-accessible memory stores, where it later reappears as seemingly valid evidence during reasoning [4,5]. Malicious records may also be

introduced indirectly through user interaction and then persist in memory as misleading context for future tasks [6]. These findings suggest that memory poisoning in LLM-based agent systems is not merely an input-level security issue. Instead, it should be understood as a dynamic propagation process in which harmful information can survive beyond the initial attack surface and spread through subsequent retrieval and communication. Recent evidence further indicates that the propagation and repair of poisoned memory in collaborative environments depend strongly on how contaminated information moves through agent interaction pathways and how quickly the system can identify and contain unreliable memory segments [7]. This makes memory integrity a central requirement for the reliable deployment of multi-agent LLM systems. Communication structure further amplifies this vulnerability. In collaborative reasoning systems, agents continuously exchange messages, partial judgments, and intermediate conclusions. These exchanges are designed to improve collective performance, but they also provide efficient channels for spreading corrupted information. Once one agent retrieves poisoned memory and incorporates it into its reasoning output, that output may influence the responses of other agents in later rounds. In this way, a local memory error can gradually become a system-level disturbance. Recent studies have shown that communication manipulation in multi-agent settings can alter global behavior even when only a limited portion of the interaction network is compromised [8,9]. Related surveys on multi-agent LLM frameworks have also emphasized that communication topology, dependency patterns, and information-sharing mechanisms strongly affect robustness and reliability [10]. Therefore, the security challenge in such systems is not limited to whether malicious content enters memory, but also includes how that content is redistributed, reinforced, and normalized through collaborative reasoning. Current defense methods mainly concentrate on attack prevention at the input stage. Common strategies include malicious prompt detection, semantic similarity screening, embedding-based anomaly identification, and other filtering mechanisms that attempt to stop harmful content before it affects agent behavior [11,12]. These methods are useful for reducing prompt injection risk and blocking some suspicious inputs before they are stored. However, their protective value becomes limited once corrupted information has already entered the memory layer. After contamination is written into shared memory, later retrieval may repeatedly reactivate the same harmful content, making the attack persistent rather than transient. Under such conditions, simple input filtering is no longer sufficient, because the main problem is no longer the arrival of malicious information but its continued reuse and circulation inside the collaborative system. This reveals a clear gap between attack prevention and post-contamination recovery in existing research [13,14]. A related line of research has explored how multi-agent collaboration itself can improve reasoning reliability. Recent studies on collective reasoning suggest that agreement among multiple agents can help reduce individual errors and improve output stability, particularly when several agents independently evaluate the same evidence or reasoning path [15]. Consensus-based interaction can therefore serve not only as a coordination mechanism, but also as a signal of information credibility. When one memory segment produces conflicting interpretations across agents, such disagreement may indicate hidden contamination, outdated content, or unreliable contextual associations. From this perspective, consensus is not only useful for answer selection, but also potentially valuable for memory diagnosis and repair. Despite this potential, current studies have rarely investigated how consensus signals can be used to detect, isolate, and correct poisoned content that has already entered shared memory in LLM-based multi-agent systems. The above observations indicate that memory poisoning defense in collaborative agent systems requires a shift in focus. Rather than relying only on front-end filtering, it is necessary to develop mechanisms that can evaluate memory reliability during interaction, prevent suspicious content from being repeatedly reused, and reduce the probability that local contamination becomes system-wide corruption. This is particularly important for multi-agent LLM systems, where reasoning quality depends heavily on persistent context sharing and inter-agent influence [16,17]. A defense mechanism at the memory level should therefore satisfy two requirements: it should be able to identify unreliable memory segments after storage, and

it should do so without severely weakening collaborative efficiency. Motivated by this need, this study proposes a consensus-guided redundant memory isolation mechanism for collaborative agent systems. The core idea is to combine redundancy and peer evaluation to improve the reliability of shared memory under adversarial conditions. Agents maintain redundant memory replicas and assess memory credibility through weighted consensus scores derived from cross-agent agreement. Memory segments with low confidence are not immediately discarded, but are temporarily isolated and rechecked before being returned to the shared memory pool. In this way, the proposed framework aims to interrupt the repeated reuse of corrupted information while preserving the benefits of collaborative reasoning. The significance of this study lies in extending the role of consensus from response correction to memory-level protection, thereby providing a practical approach for limiting poisoning propagation in shared-memory multi-agent environments. This perspective contributes to the development of more robust LLM-based collaborative systems and offers a useful foundation for secure reasoning in tasks where long-context interaction, shared state, and multi-agent cooperation are essential.

2.Materials and Methods

2.1 System Samples and Experimental Environment

The experiments were conducted in a distributed multi-agent collaboration environment designed for task allocation. The system consisted of 60 agents that communicated through a shared memory synchronization layer. Each agent stored a local memory copy and exchanged memory updates with other agents during each interaction round. The benchmark tasks required agents to coordinate decisions and assign subtasks within the group. Memory poisoning events were simulated by inserting incorrect memory segments into randomly selected agents. The poisoning rate ranged from 5% to 25% of the agents to represent different attack conditions. Each experiment ran for 40 interaction rounds to observe the spread and control of corrupted memory.

2.2 Experimental Design and Control Conditions

Two system settings were evaluated. The baseline setting used a standard memory synchronization method where agents accepted updates from peers without verification. The second setting used the proposed consensus-guided memory isolation mechanism. In this setting, agents kept redundant memory copies and calculated confidence scores before accepting external memory updates. Memory segments with low confidence values were isolated and checked before being added to the shared memory. Each poisoning condition was tested under both settings. Every configuration was repeated 20 times using different random seeds to reduce the influence of stochastic system behavior. Performance loss, contamination rate, and communication overhead were recorded as the main evaluation indicators.

2.3 Measurement Method and Quality Control

Several indicators were used to evaluate system performance and security. Collective task performance was measured as the proportion of tasks completed correctly by the agent group. Contamination rate was defined as the percentage of agents whose memory contained poisoned information. Communication overhead was calculated as the increase in message exchanges compared with the baseline system. All simulations used identical task settings and communication rules to ensure fair comparison between the two system settings. Random seeds were recorded for reproducibility, and each experiment was repeated multiple times to confirm result stability.

2.4 Data Processing and Model Formulation

Memory segments exchanged between agents were represented as embedding vectors generated from task context representations. Let m_i^t denote the memory state of agent i at interaction round t. A consensus confidence score was computed from the similarity between memory replicas provided by neighboring agents. The confidence score was defined as

$$C_i^t=\frac{1}{k}\sum_{j=1}^{k} s\,(m_i^t,m_j^t)$$

where k represents the number of trusted peers and s(·) denotes cosine similarity between two memory embeddings. When the confidence score fell below the threshold θ, the memory segment was isolated for further verification. The system contamination level at time ttt was calculated as

$$R_t = \frac{N_{poison}(t)}{N}$$

where $N_{poison}(t)$ is the number of agents containing corrupted memory at round t, and N is the total number of agents in the system. These measures were used to evaluate the spread of poisoning and the effectiveness of the isolation mechanism.

3.Results and Discussion

3.1 Impact of memory isolation on task performance

The proposed isolation mechanism improved system performance under poisoning conditions. In the baseline setting, direct memory synchronization allowed corrupted updates to enter the shared state and affect later coordination. This process reduced overall task quality. After consensus-guided isolation was applied, the decline in performance decreased by 42.1%. This result indicates that memory updates with low confidence should not be merged into shared memory immediately. Temporary isolation allows the system to verify whether the update is supported by other agents. Previous studies show that poisoned records can remain active in agent memory systems and influence later outputs once they are stored and retrieved. The result obtained here supports the need for a memory-level protection mechanism in collaborative systems [18,19]. A typical architecture of multi-agent collaboration is illustrated in Fig. 1.

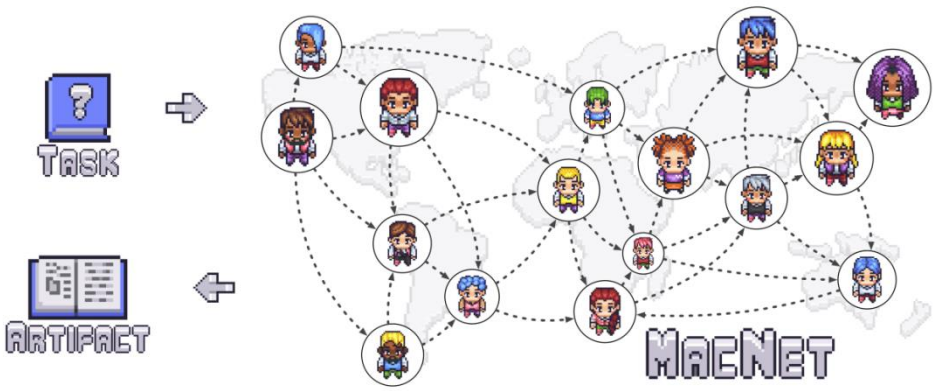


Figure 1:Structure of a collaborative agent system with shared memory and coordinated task execution.

3.2 Containment of contamination across agents

The proposed method also limited the spread of corrupted memory across the agent network. In the baseline setting, contamination reached 61% of agents. With redundant memory isolation, contamination remained below 18%. This difference is important because the major risk in collaborative systems is not only local corruption but also repeated transfer through synchronization. The results indicate that redundant memory replicas and peer-based confidence scores can slow the propagation of corrupted information. Earlier studies on agent communication show that manipulated information can spread through interaction channels and influence multiple downstream agents once synchronization occurs [20]. The present findings confirm that early containment is critical for preventing large-scale contamination. A typical interaction structure of collaborative agents is shown in Fig. 2 .

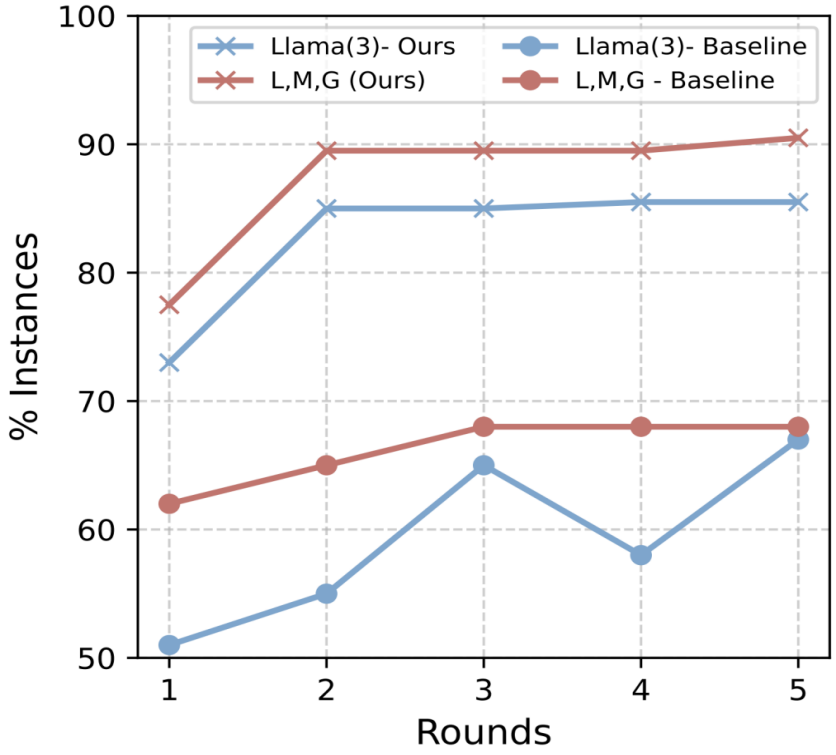


Figure 2:. Communication network showing information exchange among agents during distributed tasks.

3.3 Effect of consensus confidence on repair stability

Consensus confidence played a key role in the repair process. When the confidence score dropped below the threshold, the related memory segment was isolated and excluded from immediate reuse. This step reduced repeated contamination and improved later revalidation. The method works because it does not treat all memory updates as equally reliable. Instead, each update must pass a verification stage before synchronization. Recent studies on multi-agent consensus show that agreement among agents can improve robustness when systems operate under uncertain or adversarial conditions [21,22]. In this framework, consensus is used not only for decision selection but also for memory-state verification. This design helps maintain the reliability of the shared memory layer during continuous interaction.

3.4 Comparison with previous studies and practical implications

Compared with earlier research, the proposed framework focuses on the memory synchronization stage. Many existing defense methods concentrate on prompt filtering, anomaly detection, or final-stage voting. These methods are useful but usually operate before corruption enters memory or after the final decision is produced. The present method addresses the intermediate stage where poisoned memory may already exist and continue to spread during synchronization. Although the isolation mechanism increases communication overhead by 12.7%, the increase remains moderate relative to the gain in system robustness. This trade-off is acceptable for practical deployment because completely blocking contamination at a high coordination cost may reduce system scalability. The results suggest that redundant memory isolation provides a balanced strategy that improves resilience while maintaining reasonable efficiency in collaborative agent systems.

4.Conclusion

This study presents a consensus-based redundant memory isolation method to reduce memory poisoning in collaborative multi-agent systems. The results show that the proposed method limits the spread of corrupted memory and improves system stability during distributed decision tasks. Under simulated poisoning conditions, the isolation mechanism reduced the drop in task performance and greatly lowered the number of affected agents. The framework combines memory replication, peer confidence evaluation, and temporary isolation to protect shared memory during synchronization. This design helps maintain reliable cooperation among

agents while keeping communication cost at a reasonable level. The proposed method provides a practical way to improve the security of large language model–based agent systems. By focusing on the memory synchronization stage, the approach complements existing defenses that mainly address prompt filtering or output verification. The framework may support safer deployment of collaborative agents in applications such as automated research assistance, distributed planning, and cooperative decision support. However, the current evaluation used a limited benchmark with controlled poisoning scenarios. Future work should examine the method in larger agent networks, more complex task settings, and stronger adversarial environments. Additional research may also explore adaptive confidence thresholds and topology-aware isolation strategies to further improve scalability and system robustness.

References

Hasan, R. (2025). A Systematic Review Of Human-AI Collaboration In It Support Services: Enhancing User Experience And Workflow Automation. American Journal of Interdisciplinary Studies, 6(3), 01-37.

Qiu, D., Xu, D., & Yue, L. (2025, December). Reinforcement Learning-Augmented LLM Agents for Collaborative Decision Making and Performance Optimization. In 2025 7th International Conference on Frontier Technologies of Information and Computer (ICFTIC) (pp. 1337-1342). IEEE.

Enßlin, T. A., Kainz, V., & Boehm, C. (2023). Simulating reputation dynamics and their manipulation: An agent based model framework. Computational Communication Research, 5(1), 1.

Qiu, Y. (2024). Estimation of tail risk measures in finance: Approaches to extreme value mixture modeling. arXiv preprint arXiv:2407.05933.

Youvan, D. C. (2025). Pure Information from Quantum Randomness: QRNGs as Interfaces Between Substrate, Qubits, and the It-From-Bit Cosmos.

Chen, H., Li, J., Ma, X., & Mao, Y. (2025, June). Real-time response optimization in speech interaction: A mixed-signal processing solution incorporating C++ and DSPs. In 2025 7th International Conference on Artificial Intelligence Technologies and Applications (ICAITA) (pp. 110-114). IEEE.

Liu, H., Xu, D., Ma, Q., Xu, S., & Qiu, D. (2026). Memory Poisoning Propagation and Repair Mechanism in Multi-Agent Collaborative Environments.

Ponniah, J., & Dantsker, O. D. (2022). Strategies for scaleable communication and coordination in multi-agent (uav) systems. Aerospace, 9(9), 488.

Tran, K. T., Dao, D., Nguyen, M. D., Pham, Q. V., O'Sullivan, B., & Nguyen, H. D. (2025). Multi-agent collaboration mechanisms: A survey of llms. arXiv preprint arXiv:2501.06322.

Ma, Q., Yue, L., Xu, S., Shi, Y., & Liu, H. (2026). Web Agent Agentic Reinforcement Learning Decision Model Under Multi-Cost and Failure Risk Constraints.

Alamsabi, M., Tchuindjang, M., & Brohi, S. (2026). Embedding-Based Detection of Indirect Prompt Injection Attacks in Large Language Models Using Semantic Context Analysis. Algorithms, 19(1), 92.

Li, T., Liu, S., Hong, E., & Xia, J. (2025). Human Resource Optimization in the Hospitality Industry Big Data Forecasting and Cross-Cultural Engagement.

Małuszyński, M. J., Pawluśkiewicz, B., Gnatowski, T., Małuszyńska, I., Wołejko, E., Wydro, U., ... & Dąbrowski, P. (2026). The Effect of Petroleum-Derived Compounds in Soil on Microbiological Activity and the Physiological State of Plants. Applied Sciences, 16(4), 2076.

Bai, W., Wu, Q., Wu, K., & Lu, K. (2024). Exploring the Influence of Prompts in LLMs for Security-Related Tasks. In Workshop on Artificial Intelligence System with Confidential Computing (AISCC 2024)(San Diego, CA). USA. https://dx. doi. org/10.14722/aiscc.

Amirkhani, A., & Barshooi, A. H. (2022). Consensus in multi-agent systems: a review. Artificial Intelligence Review, 55(5), 3897-3935.

Du, Y. (2025). Research on Deep Learning Models for Forecasting Cross-Border Trade Demand Driven by Multi-Source Time-Series Data. Journal of Science, Innovation & Social Impact, 1(2), 63-70.

Maragheh, R. Y., & Deldjoo, Y. (2025). The future is agentic: Definitions, perspectives, and open challenges of multi-agent recommender systems. arXiv preprint arXiv:2507.02097.

Liu, S., Feng, H., & Liu, X. (2025). A Study on the Mechanism of Generative Design Tools' Impact on Visual Language Reconstruction: An Interactive Analysis of Semantic Mapping and User Cognition. Authorea Preprints.

Sunil, B. D., Sinha, I., Maheshwari, P., Todmal, S., Mallik, S., & Mishra, S. (2026). Memory Poisoning Attack and Defense on Memory Based LLM-Agents. arXiv preprint arXiv:2601.05504.

Gu, X., Yang, J., Tian, X., & Liu, M. (2025). Research on the Construction of a Human-Machine Collaborative Anti-Money Laundering System and Its Efficiency and Accuracy Enhancement in Suspicious Transaction Identification.

Amirkhani, A., & Barshooi, A. H. (2022). Consensus in multi-agent systems: a review. Artificial Intelligence Review, 55(5), 3897-3935.

Yang, Y., Leuze, C., Hargreaves, B., Daniel, B., & Baik, F. (2025). EasyREG: Easy Depth-Based Markerless Registration and Tracking using Augmented Reality Device for Surgical Guidance. arXiv preprint arXiv:2504.09498.