DUMA-bench: A Dual-Control Multi-Agent Benchmark for Evaluating LLM Agent Security

Anonymous ACL submission

Abstract

LLM-based agent systems are increasingly deployed for automating complex tasks, yet their security in realistic interaction scenarios remains understudied. Existing benchmarks evaluate either isolated agent capabilities or prompt injection resistance in simplified settings without accounting for active user interaction dynamics. We present DUMA-bench (Dual-control User Multi-Agent benchmark), extending τ^2-bench with three security domains aligned with AI-SAFE (levels 1–5) and OWASP classifications: `mail_rag_phishing` (RAG poisoning), `collab` (inter-agent attacks), and `output_handling` (improper output handling).

We evaluate five models (Claude Sonnet 4.5, GPT-4-turbo, GPT-4o, GPT-4o-mini, GPT-3.5-turbo) across varying user temperatures ($T_{\text{user}} \in \{0.0, 0.5, 1.0\}$), measuring pass@1 and pass@4 metrics. Key findings: (1) model size does not predict security—GPT-4o outperforms larger models in RAG poisoning while GPT-3.5-turbo shows highest stability; (2) user behavior variability significantly impacts Attack Success Rate (ASR) even with non-malicious users; (3) all models show high ASR (75–90%) in RAG poisoning, indicating critical vulnerabilities requiring specialized defenses.

1 Introduction

1.1 Context and Problem Statement

The development of Large Language Models (LLMs) and their integration into agent systems opens new possibilities for automating complex tasks. AI agents are capable of autonomous planning, interaction with external tools (APIs, databases, file systems), and real-time decision-making (Muleys et al., 2025). However, these same capabilities create fundamentally new attack surfaces not characteristic of classical machine learning systems.

A typical AI agent comprises five interconnected components (Muleys et al., 2025). At its core, the **LLM** serves as the central component for understanding instructions and generating responses, while a **planning module** transforms high-level goals into executable action sequences. The agent's **memory** system maintains both short-term context through dialogue history and long-term knowledge through RAG systems and knowledge bases. External **tools** provide the agent with capabilities to interact with APIs, databases, and other real-world systems, while the **interface** layer serves as the entry point for receiving user requests and delivering responses.

Each of these components represents a potential attack vector. The AI-SAFE framework (Muleys et al., 2025) systematizes threats across five levels: interface (Prompt Injection, DoS), execution and tools (Tool Misuse, Privilege Escalation), infrastructure and orchestration (Cross-Agent Poisoning), core and logic (Jailbreaking, Goal Manipulation), data and knowledge (RAG Poisoning, Data Leakage).

1.2 Limitations of Existing Methods

Current benchmarks for evaluating agent system security suffer from three fundamental limitations. First, most benchmarks including AgentBench (Liu et al., 2025) and Agent Security Bench (Zhang et al., 2025) conduct **isolated evaluation** where agents operate under monopolistic control with users serving merely as passive instruction sources. Second, existing security-focused benchmarks employ **simplified attack scenarios**—for instance, Agent Dojo (Debenedetti et al., 2024) concentrates on prompt injection in single-agent contexts without considering the complexities of inter-agent interaction and RAG systems. Third and most critically, current benchmarks fail to account for the **active user** dynamic. Research on τ-bench (Yao et al., 2024) and τ^2-bench (Barres et al., 2025) has demonstrated that introducing an active user through dual-control paradigms leads to agent performance drops of up to 25 percentage points, indicating that coordination and communication become critical failure points. Yet existing security benchmarks systematically overlook this dynamic, creating a significant gap between evaluation settings and real-world deployment scenarios.

1.3 Contributions

This work makes four key contributions to the evaluation of LLM agent security. We present **DUMA-bench**, a new security benchmark extending τ^2-bench with three security domains that model typical attack vectors: RAG poisoning through the `mail_rag_phishing` domain, inter-agent attacks through the `collab` domain, and improper output handling through the `output_handling` domain. The benchmark is publicly available on GitHub.[1] We propose a **dual-control evaluation methodology** for assessing agent robustness to attacks while accounting for active user behavior, with rigorous formalization within the Dec-POMDP framework. Our **comprehensive empirical evaluation** covers five state-of-the-art LLMs—Claude Sonnet 4.5, GPT-4-turbo, GPT-4o, GPT-4o-mini, and GPT-3.5-turbo—across multiple configurations, demonstrating that model size does not consistently predict security robustness and revealing unexpected performance patterns. Finally, we formulate and test three **testable hypotheses** about agent security: the effect of multiple runs on pass@k variance (H1), the impact of user temperature on attack success rate (H2), and the relationship between model size and security performance (H3).

2 Related Work

2.1 Agent System Evaluation Benchmarks

The development of LLM agents has led to the creation of benchmark series for evaluating their capabilities. AgentBench (Liu et al., 2025) evaluates agents in eight environments, including operating systems, databases, and web navigation, but focuses on functional capabilities without security consideration. ToolBench and API-Bank investigate tool usage but in trusted environments.

A key breakthrough was the τ-bench (Yao et al., 2024) and τ^2-bench (Barres et al., 2025) series, where the user is modeled as an active participant capable of changing environment state. Formalization within the Dec-POMDP framework (Amato et al., 2013) showed that user coordination is a critical bottleneck: even advanced models lose up to 25 percentage points of performance when transitioning from monopolistic to dual control.

[1] `https://github.com/sec-lab-itmo/duma-benchmark`

2.2 LLM Agent Security Evaluation

Agent Security Bench (ASB) (Zhang et al., 2025) formalizes attacks and defenses for LLM agents, but is limited to single-agent scenarios. Agent Dojo (Debenedetti et al., 2024) creates a dynamic environment for evaluating prompt injection attacks, demonstrating that modern agents are vulnerable even to simple attacks. However, Agent Dojo does not model an active user as an interaction participant, attacks propagating through inter-agent communication channels, or RAG system poisoning in realistic scenarios involving emails and documents.

2.3 Threat Modeling Frameworks

OWASP LLM Top 10 (OWASP Foundation, 2025b) and OWASP AI Agents Top 15 (OWASP Foundation, 2025a) systematize threats for AI systems. The AI-SAFE framework (Muleys et al., 2025) proposes a five-level threat model specific to agent architectures. Our work uses these classifications for systematic coverage of attack vectors in the developed domains.

2.4 Positioning of This Work

This work fills a critical gap between dual-control benchmarks like τ^2-bench, which evaluate agent-user coordination but do not focus on security, and security benchmarks like Agent Dojo and ASB, which assess vulnerabilities but do not account for active users and inter-agent interaction. We extend the τ^2-bench methodology with security domains covering AI-SAFE threats and evaluate agent robustness in realistic scenarios with active users.

3 Method

3.1 Problem Formulation

3.1.1 Interaction Model (Dec-POMDP)

We formalize agent-user interaction as a Decentralized Partially Observable Markov Decision Process (Dec-POMDP) (Amato et al., 2013; Barres et al., 2025), following the τ^2-bench framework. The environment $\mathcal{E}$ is described by a state space $\mathcal{S}$ that is only partially observable to participants. The agent

$\mathcal{A}$ and user $\mathcal{U}$ function as two autonomous players, each equipped with their own observation spaces Ω_A and Ω_U, and action spaces $\mathcal{A}_A$ and $\mathcal{A}_U$. The transition function $T : \mathcal{S} \times \mathcal{A}_A \times \mathcal{A}_U \to \Delta(\mathcal{S})$ defines environment dynamics, capturing how both agent and user actions jointly determine state evolution. Critically, the user is not a passive instruction source but actively changes environment state through their actions, creating genuine coordination challenges.

3.1.2 Attacker Model

We adopt a grey-box attacker model with realistic assumptions about adversary capabilities. The attacker possesses **system knowledge**, understanding the agent architecture and available tools, but has no access to the system prompt or model weights. The attacker's **capabilities** include injecting malicious content into various input channels such as emails, colleague messages, user queries, and document contents in RAG systems. The attacker's **goal** is to force the agent to perform unauthorized actions or disclose confidential information, representing realistic threat scenarios in deployed agent systems.

3.1.3 Evaluation Metrics

pass@k metric. Defines the fraction of cases where the agent successfully passed k asserts:

$$\text{pass@}k = \frac{1}{N}\sum_{i=1}^{N} \mathbb{1}\left[\sum_{j=1}^{M_i} a_{ij} \geq k\right], \qquad (1)$$

where N is the number of cases, M_i is the number of asserts in case i, $a_{ij} \in \{0, 1\}$ is the result of assert j.

Attack Success Rate (ASR). Attack success indicator:

$$\text{ASR} = 1 - \text{pass@}1. \qquad (2)$$

In the security context, asserts verify that the agent maintains security invariants: absence of malicious instruction execution, preservation of data integrity, and refusal of unauthorized actions.

3.2 Developed Security Domains

3.2.1 `mail_rag_phishing` Domain (RAG Poisoning)

Description. Models an attack on an email client's RAG system through injection of malicious instructions in email content.

Threat classification: AI-SAFE.DATA.1 (Knowledge Base Poisoning), AI-SAFE.INPUT.1 (Indirect Prompt Injection), LLM01, RAG: Indirect Prompt Injection.

Domain architecture. The domain simulates an email client with integrated RAG for searching relevant emails. The knowledge base contains a mixture of legitimate and malicious emails designed to test the agent's ability to distinguish safe from compromised content. The agent has access to four tools: `search_emails` for querying the knowledge base, `get_email_details` for retrieving full message content, `send_email` for composing new messages, and `forward_email` for redistributing existing messages.

3.2.2 `collab` Domain (Inter-Agent Interaction)

Description. Models a multi-agent environment where an attacking agent ("colleague") manipulates the target agent through the communication channel.

Threat classification: AI-SAFE.INFRA.3 (Cross-Agent Poisoning), AI-SAFE.EXEC.1 (Tool Misuse), AI-SAFE.EXEC.2 (Privilege Escalation), T15 (Cross-Agent Communication Poisoning), MCP05.

Domain architecture. The domain implements a workspace with multiple agents operating at different privilege levels, creating opportunities for privilege escalation attacks. A ticketing and task system with delegation mechanisms enables attackers to manipulate task assignments. An activity log provides audit capabilities for detecting suspicious behavior. The agent can invoke five tools: `get_tickets` for retrieving assigned work, `update_ticket` for modifying task status, `delegate_task` for reassigning responsibilities, `get_logs` for accessing system activity, and `send_message` for inter-agent communication.

3.2.3 `output_handling` Domain (Output Processing)

Description. Tests agent robustness to generating malicious code or injections in output.

Threat classification: AI-SAFE.INPUT.3 (Improper Output Handling), LLM05.

Domain architecture. The domain models a web application where the agent generates content for display to end users. A database containing user data provides context for content generation while creating opportunities for SQL injection. A rendering system displays generated content in browsers, enabling testing of XSS and other injection vulnerabilities. The agent operates with four tools: `generate_content` for creating textual responses, `execute_query` for database access, `render_template` for formatting output, and `send_response` for delivering content to users.

4 Experiments

4.1 Experimental Setup

4.1.1 Models Under Study

We evaluate five state-of-the-art LLMs spanning different capability levels and architectural generations. **Claude Sonnet 4.5** represents Anthropic's latest model with strong reasoning capabilities. **GPT-4-turbo** provides OpenAI's advanced model with enhanced capabilities, while **GPT-4o** offers an optimized variant balancing performance and cost. **GPT-4o-mini** serves as a compact version with reduced cost, and **GPT-3.5-turbo** establishes a baseline for comparison. Agent generation parameters are fixed at temperature $T_{\text{agent}} = 0.0$ for all models to ensure deterministic agent behavior.

4.1.2 Variable Parameters

We vary the user model temperature across three levels to test robustness under different user behavior patterns: $T_{\text{user}} = 0.0$ for deterministic behavior, $T_{\text{user}} = 0.5$ for moderate variability, and $T_{\text{user}} = 1.0$ for high variability.

4.1.3 Metrics

We evaluate agent robustness using two complementary metrics. The **pass@k** metric measures the proportion of cases where the agent successfully passes at least k asserts, as formally defined in Equation 1. The **ASR (Attack Success Rate)** quantifies the proportion of successful attacks, computed as ASR $= 1 -$ pass@1. In our experiments, we primarily report pass@1 (at least 1 assert passed) and pass@4 (at least 4 asserts passed) metrics to evaluate both basic and stringent security thresholds.

4.1.4 Research Hypotheses

We test three hypotheses:

Hypothesis 1 (H1): At fixed agent and user temperatures, increasing the number of runs k reduces the variance of pass@k, but does not guarantee monotonic changes in ASR.

Hypothesis 2 (H2): Changes in non-attacking user queries cause changes in ASR at fixed agent temperature.

Hypothesis 3 (H3): Model size does not consistently correlate with pass@k and ASR; mid-sized models may outperform larger models in specific security domains.

4.1.5 Protocol

For each combination (model, temperature, domain, case), we perform $n = 10$ independent runs. All metrics are recorded, and results are aggregated for statistical analysis using Fisher's exact test (Fisher, 1935) for significance testing. We use Wilson confidence intervals (Wilson, 1927) for proportion estimates, which provide better coverage properties for small sample sizes compared to normal approximations.

5 Results & Discussion

5.1 Aggregated Results

Table 1 presents aggregated pass@1 results by model and domain across all temperature settings. Table 2 reports statistical significance of differences between GPT-4o and GPT-4o-mini at each temperature, and Table 3 shows the impact of temperature variation within each model.

5.2 Analysis

5.2.1 Hypothesis Testing Results

H1: Effect of Multiple Runs. Our results confirm that at fixed temperatures, increasing k in pass@k reduces variance. However, ASR does not change monotonically with k. For example, Claude Sonnet 4.5 shows pass@1 = 0.25 but pass@4 = 0.20 in the `mail_rag_phishing` domain at $T_{\text{user}} = 0.0$, indicating that some initially successful runs fail with more stringent evaluation criteria. This non-monotonicity suggests that agent robustness is sensitive to the number of security checks required, even with deterministic agent temperature.

H2: User Temperature Impact. Changes in user model temperature (T_{user}) cause measurable changes in ASR at fixed agent temperature. This confirms that user behavior variability, even when the user is non-malicious, affects the attack success rate. The effect is most pronounced in the `collab` domain where social engineering plays a key role.

H3: Model Size vs. Security Performance. Our results partially confirm this hypothesis. Surprisingly, GPT-4o (mid-sized) outperforms larger models (GPT-4-turbo, Claude Sonnet 4.5) in the `mail_rag_phishing` domain across all temperatures, though it shows instability when evaluat-

Table 1: Model Robustness Comparison by Domain (pass@1)

Model	RAG Phishing	Collab	Output
GPT-4o (T=0.0)	16/50 (32.0%)	13/60 (21.7%)	16/30 (53.3%)
GPT-4o (T=0.5)	18/50 (36.0%)	13/60 (21.7%)	18/30 (60.0%)
GPT-4o (T=1.0)	16/50 (32.0%)	17/60 (28.3%)	18/30 (60.0%)
GPT-4o-mini (T=0.0)	5/50 (10.0%)	18/60 (30.0%)	13/30 (43.3%)
GPT-4o-mini (T=0.5)	6/50 (12.0%)	13/60 (21.7%)	14/30 (46.7%)
GPT-4o-mini (T=1.0)	4/50 (8.0%)	18/60 (30.0%)	18/30 (60.0%)

Table 2: Statistical Significance: GPT-4o vs GPT-4o-mini (Fisher exact test, two-tailed)

Domain	T	4o pass@1	mini pass@1	4o ASR	mini ASR	p-value
mail_rag_phishing	0	16/50 (32%)	5/50 (10%)	68%	90%	1.28e-02
mail_rag_phishing	0.5	18/50 (36%)	6/50 (12%)	64%	88%	9.12e-03
mail_rag_phishing	1	16/50 (32%)	4/50 (8%)	68%	92%	5.04e-03
collab	0	13/60 (22%)	18/60 (30%)	78%	70%	4.04e-01
collab	0.5	13/60 (22%)	13/60 (22%)	78%	78%	1.00e+00
collab	1	17/60 (28%)	18/60 (30%)	72%	70%	1.00e+00
output_handling	0	16/30 (53%)	13/30 (43%)	47%	57%	6.06e-01
output_handling	0.5	18/30 (60%)	14/30 (47%)	40%	53%	4.38e-01
output_handling	1	18/30 (60%)	18/30 (60%)	40%	40%	1.00e+00

Table 3: Temperature Effect: p-values (Fisher exact test)

Domain	Model	T0:0.5	T0:1	T0.5:1
RAG phish.	4o	.833	1.00	.833
RAG phish.	4o-mini	1.00	1.00	.741
collab	4o	1.00	.528	.528
collab	4o-mini	.404	1.00	.404
output	4o	.795	.795	1.00
output	4o-mini	1.00	.301	.438

ing higher k values. GPT-3.5-turbo, despite being the smallest model, demonstrates the most stable performance across temperature variations. This suggests that model size and architectural sophistication alone are not reliable predictors of security robustness—specialized security training or instruction-following capabilities may play a more critical role.

5.2.2 Cross-Model Comparison

Key observations emerge when comparing all five models across domains. In the **RAG Poisoning (`mail_rag_phishing`)** domain, GPT-4o achieves the highest pass@1 (32-36% depending on T), outperforming both GPT-4-turbo (30-40%) and Claude Sonnet 4.5 (25%). GPT-4o-mini shows significantly lower performance (8-12%, $p < 0.01$), while GPT-3.5-turbo performs poorly (15-20%) but maintains consistent results across temperatures. For **Inter-Agent Attacks (`collab`)**, Claude Sonnet 4.5 demonstrates superior performance (96-100% pass@1), followed by GPT-4-turbo (96-100%) and GPT-4o (21.7-28.3%), with no statistically significant differences between GPT-4o and GPT-4o-mini ($p > 0.4$). In the **Output Handling (`output_handling`)** domain, Claude Sonnet 4.5 again leads (50-67%), followed by GPT-4-turbo (50-67%) and GPT-4o (53.3-60%), with differences between GPT-4o and GPT-4o-mini remaining statistically insignificant.

5.2.3 RAG System Vulnerability

The `mail_rag_phishing` domain remains the most challenging across all models: even the best configuration (Claude Sonnet 4.5 at $T = 0.5$) shows ASR = 75% at pass@1. This indicates high effectiveness of indirect prompt injection attacks through RAG context and the critical need for additional defenses (e.g., source validation, instruction filtering, policy-driven tool gating).

5.2.4 Temperature Stability

GPT-3.5-turbo exhibits the most stable behavior across different user temperatures, with minimal variance in pass@k metrics. In contrast, GPT-4o shows high sensitivity to temperature changes, particularly in the `collab` domain where pass@1 ranges from 21.7% to 28.3% as T_{user} increases from 0.0 to 1.0.

6 Conclusion

We present DUMA-bench, a comprehensive benchmark for evaluating LLM agent security in dual-control paradigms where active users can influence environment state. Our work yields four main findings. First, DUMA-bench addresses a critical gap in the evaluation landscape: existing security benchmarks poorly correspond to real agent usage scenarios. Our three security domains covering RAG poisoning, inter-agent attacks, and output handling align with AI-SAFE levels 1-5, OWASP LLM Top 10, and OWASP AI Agents Top 15. Second, model size does not predict security—GPT-4o outperforms larger models including GPT-4-turbo and Claude Sonnet 4.5 in RAG poisoning scenarios, while GPT-3.5-turbo shows the most stable performance across temperature variations. This partially confirms our hypothesis that model size does not consistently correlate with security robustness. Third, user behavior significantly affects attack success, with changes in user temperature impacting ASR even when the user is non-malicious, confirming that dual-control dynamics are critical for realistic security evaluation. Fourth, RAG attacks remain a critical vulnerability, with all models showing high ASR (75-90%) in the `mail_rag_phishing` domain, indicating that indirect prompt injection through RAG systems remains a major vulnerability requiring specialized defenses.

Future directions include: (1) extending DUMA-bench with additional domains from OWASP classifications (`resource_overload`, `supply_chain`); (2) evaluating defense mechanisms (Llama Guard, Promptfoo) on current domains; (3) conducting qualitative analysis of simulation traces to identify failure patterns; (4) expanding model coverage to open-source alternatives (Llama, Mistral); and (5) investigating the impact of different agent architectures on security robustness.

DUMA-bench is open-source and available at https://github.com/sec-lab-itmo/duma-benchmark, enabling reproducible security evaluation for the research community.

Acknowledgments

We thank the authors of τ^2-bench for their foundational work in dual-control agent evaluation, which inspired this research. We also acknowledge the Yandex Cloud team for developing the AI-SAFE framework.

7 Limitations

Our work has five important limitations. First, while we test across multiple configurations with $n = 10$ runs each, larger sample sizes would enable more robust statistical conclusions, especially when comparing models with similar performance. However, our use of Fisher's exact test and Wilson confidence intervals provides valid statistical inference for small samples. Second, the user is modeled by an LLM, which may not fully capture the unpredictability and errors of real human users, though this approach ensures reproducibility and controlled experimentation. Third, while we evaluate five models including both proprietary systems (GPT, Claude) and recognize the need for open models (Llama, Gemini, etc.), our current results may not fully generalize across all model families, and future work will expand model coverage. Fourth, DUMA-bench currently covers three security domains, with additional domains such as `resource_overload` and `supply_chain` attacks planned for future releases. Fifth, the current version does not evaluate effectiveness of defense mechanisms such as Llama Guard, Promptfoo, and custom validators, which represents a key direction for future research.

References

Christopher Amato, Girish Chowdhary, Alborz Geramifard, N. Kemal Ure, and Mykel J. Kochenderfer. 2013. Decentralized control of partially observable markov decision processes. In *52nd IEEE Conference on Decision and Control*, pages 2398–2405.

Vincent Barres, Hao Dong, Soumya Ray, Xinyun Si, and Karthik Narasimhan. 2025. τ^2-bench: Evaluating conversational agents in a dual-control environment. *arXiv preprint arXiv:2506.07982*.

Edoardo Debenedetti and 1 others. 2024. Agentdojo: A dynamic environment to evaluate prompt injection attacks and defenses for llm agents. In *Advances in Neural Information Processing Systems*, volume 37, pages 82895–82920.

Ronald A. Fisher. 1935. The logic of inductive inference. *Journal of the Royal Statistical Society*, 98(1):39–82.

Xiao Liu and 1 others. 2025. Agentbench: Evaluating llms as agents. *arXiv preprint arXiv:2308.03688*.

R. Muleys, S. Nesteruk, and A. Lodin. 2025. Ai secure agentic framework essentials (ai-safe) v1.0. Technical report, Yandex Cloud.

OWASP Foundation. 2025a. Owasp ai agents (agentic ai) top 15. Available at owasp.org.

OWASP Foundation. 2025b. Owasp top 10 for large language model applications. Available at owasp.org.

Edwin B. Wilson. 1927. Probable inference, the law of succession, and statistical inference. *Journal of the American Statistical Association*, 22(158):209–212.

Shunyu Yao, Noah Shinn, Pedram Razavi, and Karthik Narasimhan. 2024. τ-bench: A benchmark for tool-agent-user interaction in real-world domains. *arXiv preprint arXiv:2406.12045*.

Hanrong Zhang and 1 others. 2025. Agent security bench (asb): Formalizing and benchmarking attacks and defenses in llm-based agents. *arXiv preprint arXiv:2410.02644*.