Mind the Agent: A Comprehensive Survey on Large Language Model-Based Agent Safety

Gaotang Li *gaotang3@illinois.edu*
gaotang3

Ting-Wei Li *twli@illinois.edu*
twli

Xuying Ning *xuyingn2@illinois.edu*
xuyingn2

Reviewed on OpenReview: *https://openreview.net/forum?id=DHeOUXipKU&noteId=DHeOUXipKU*

Abstract

The emergence of Large Language Model (LLM)-based agents represents a significant shift in AI systems—from passive language models to autonomous agents equipped with memory, tool-use capabilities, and long-horizon planning. While these agents unlock new possibilities across web automation, embodied robotics, and collaborative systems, they also introduce fundamentally novel safety risks that go beyond traditional LLM vulnerabilities. This survey provides a comprehensive overview of the growing field of LLM-based agent safety. We begin by contrasting LLM agents with standard LLMs, outlining how agent-specific capabilities amplify safety challenges such as execution-based harm, memory poisoning, and emergent failures in multi-agent collaboration. We categorize recent works into four major threat types—adversarial attacks, jailbreaking attacks, backdoor attacks, and multi-agent failures—and systematically examine how each exploits different stages of the agent pipeline. For each threat, we review proposed defense strategies, including robust training, prompt filtering, backdoor deactivation, and adversarial simulation. To evaluate these defenses, we survey the emerging landscape of agent safety benchmarks. We introduce a taxonomy based on attack surface, evaluation targets, and interaction complexity, and compare benchmark coverage across scenarios and models. Finally, we discuss open challenges and future directions, including dynamic and proactive safety evaluation, training-time alignment, and scalable defenses for real-world deployment. Our goal is to provide a structured foundation for advancing the safe and responsible development of LLM-based agents.

1 Introduction

Large Language Model (LLM)-based agents represent a transformative shift in artificial intelligence—from passive text generators to autonomous systems capable of memory, tool use, long-horizon planning, and real-world decision-making (Huang et al., 2024; Wang et al., 2024a). This evolution opens the door to powerful applications in web automation (Ning et al., 2025), embodied robotics (Ma et al., 2024), and collaborative systems (Li et al., 2024b). However, it also introduces new and fundamentally different safety risks that traditional LLM evaluations fail to capture.

Unlike conventional LLMs, which operate in single-turn, reactive settings, LLM agents are endowed with autonomy, memory persistence, and interaction capabilities. These features dramatically expand their attack surface and failure modes. Agents may not only hallucinate unsafe responses, but also execute harmful actions, propagate poisoned memory, or behave erratically in multi-agent contexts. Consequently, threats to agent safety go far beyond well-studied issues such as toxicity or jailbreaks in static chatbots.

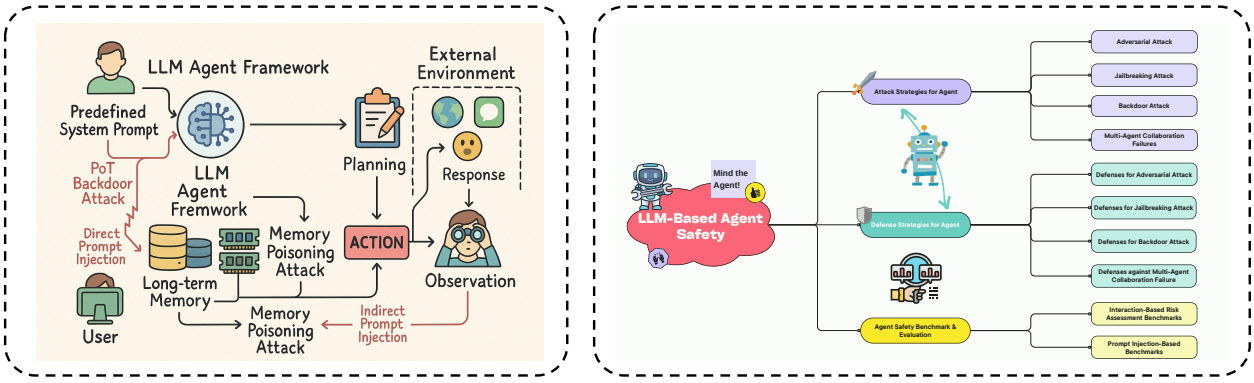

Figure 1: Left – Overview of the LLM Agent Attacking Framework, illustrating key attack vectors such as Direct Prompt Injection (DPI), Indirect Prompt Injection (IPI), Plan-of-Thought (PoT) Backdoor, and Memory Poisoning Attacks, each targeting different components of the agent pipeline including user queries, observations, system prompts, and memory retrieval during planning and execution. Right – Overall taxonomy of LLM-based agent safety, categorizing existing research by attack surfaces, corresponding defense strategies, and evaluation benchmarks.

In this survey, we provide a comprehensive overview of the emerging field of LLM-based agent safety, as shown in Fig. 1. We first distinguish agent safety from traditional LLM concerns, highlighting how capabilities like persistent memory and tool execution amplify existing risks while introducing new ones. We then systematically categorize recent attack strategies into four key types: adversarial input attacks, jailbreaking, backdoor attacks, and multi-agent collaboration failures—each targeting different components of the agent pipeline.

For each threat, we examine representative research and corresponding defense strategies, such as adversarial input filtering, backdoor deactivation, and collaborative robustness frameworks. We also review the emerging landscape of agent safety benchmarks, introducing a taxonomy based on attack surface, evaluation targets, and interaction complexity. Finally, we discuss open challenges and future directions, including the need for dynamic safety metrics, proactive safety training, and scalable defenses for real-world deployment.

Our goal is to provide a structured foundation for researchers and practitioners to understand, evaluate, and mitigate safety risks in LLM-based agents—ensuring their safe integration into increasingly autonomous and high-stakes applications.

2 Preliminaries

In this section, we compare LLMs with LLM-based agents, and outline how their safety challenges differ—covering core capabilities, risks, and design implications.

2.1 LLM vs LLM-based Agent

Large Language Models (LLMs) have revolutionized natural language processing by enabling machines to understand and generate human-like text. These models, trained on vast corpora, excel in tasks such as text completion, summarization, and question answering (Qin et al., 2024; Hagos et al., 2024; Li et al., 2025; Yao et al., 2024b). However, their capabilities are primarily reactive—they generate responses based on input prompts without inherent goals or the ability to interact with external environments.

In contrast, LLM-based agents represent a significant evolution in AI systems. By integrating LLMs with additional components, these agents transition from passive text generators to autonomous entities capable of decision-making and action execution (Crouse et al., 2023; Cheng et al., 2024; Barua, 2024). This transformation introduces new dimensions to AI applications, including:

- **Autonomously set and pursue goals**: Unlike traditional LLMs that respond to prompts, agents can initiate actions based on predefined objectives (Crouse et al., 2023; Kannan et al., 2024; Gao et al., 2025; Liu et al., 2024b).
- **Interact with external tools and environments**: Agents can perform tasks such as web browsing, code execution, and interfacing with APIs (Shi et al., 2024; Debenedetti et al., 2024; Guo et al., 2024; Zhang et al., 2024c).
- **Maintain and utilize memory**: They can store and recall information over extended periods, enabling context-aware decision-making (Xu et al., 2025; Mei et al., 2024; Zhong et al., 2024; Packer et al., 2023).
- **Engage in multi-turn, dynamic interactions**: Agents can handle complex workflows that require reasoning over multiple steps and adapting to new information (Li et al., 2025; Guan et al., 2025; Park et al., 2024; Zhang et al., 2025; Deng et al., 2024; Gao et al., 2025; Liu et al., 2024b).

2.2 LLM Safety vs LLM-based Agent Safety

While traditional LLMs are powerful tools for text generation and understanding, they operate in a reactive manner. In contrast, LLM-based agents augment LLMs with components such as long-term memory, planning modules, and tool-use capabilities, enabling them to act autonomously within dynamic environments.

This shift dramatically alters the landscape of safety.ty. Although conventional LLM risks are mostly confined to textual harms, LLM-based agents introduce the potential for real-world consequences through autonomous action and persistent memory. These expanded capabilities open up new vectors for failure and attack, which we categorize as follows.

- **Execution-Based Harm**: Agents can perform actions in the world (e.g., API calls, file execution, controlling hardware). Attackers can compromise this capability, causing agents to leak sensitive data or perform physical damage. Contextual backdoors demonstrate that subtle poisoning of prompts can lead to dangerous downstream effects (Chen et al., 2025; Debenedetti et al., 2024; Zhang et al., 2024a; Wang et al., 2024b; Zhang et al., 2024b).
- **Memory Poisoning and Manipulation**: Agents often maintain persistent memory (e.g., knowledge bases or episodic memory). Adversaries can inject malicious data into memory, lying dormant until triggered—an attack far more durable and stealthy than prompt injection (Chen et al., 2024; Dong et al., 2025; Wang et al., 2025a).
- **Risks from Multi-Agent Interactions**: In multi-agent settings, novel forms of deception and collusion arise. Agents can: (i) reinforce each other's hallucinations; (ii) spread malicious prompts across peers; (iii) learn to hide intentions from humans (Shahroz Khan et al., 2025; Lee & Tiwari, 2024; Cemri et al., 2025; He et al., 2025; Yu et al.).

3 Attack Strategies

Overview. LM-based agents, by virtue of their autonomy, tool integration, and persistent memory, introduce new attack surfaces beyond those seen in traditional Large Language Models. In this section, we outline common strategies used by adversaries to exploit these expanded vulnerabilities. We focus on four major attack categories—adversarial input attacks, jailbreaking attacks, backdoor attacks, and multi-agent collaboration failures—while also highlighting additional threats such as memory poisoning and prompt injection.

3.1 Adversarial Input Attacks

Adversarial input attacks involve carefully crafted inputs that induce an LLM agent to produce undesired outputs (Chakraborty et al., 2018). These inputs can be subtle perturbations of prompts or context that

remain natural to a human reader but exploit the model's vulnerabilities (Zou et al., 2023). For example, an aligned language model that would normally refuse harmful requests can be coaxed into generating prohibited content by a worst-case input designed to circumvent its safety filters. Attackers employ tactics such as synonym substitutions (Hauser et al., 2021; Chiang & Lee, 2022), gibberish token insertions, or gradient-guided prompt tweaks (Guo et al., 2021) to create adversarial examples that maximally confuse the model while preserving fluent language. Some of these techniques are also known as "prompt injection" (Liu et al., 2023). Such malicious prompts or data manipulations may cause the agent to misinterpret user intent or violate its alignment instructions. In particular, these attacks can be seamlessly integrated into the general framework of an LM-agent. For instance, Mo et al. (2024) points out the potential hazard in the three stages of an agent: Perception, Brain, and Action. Subsequent studies also expanded their study to VLM-based agents (Wu et al., 2024).

3.2 Jailbreaking Attacks

Jailbreaking attacks are explicit attempts to bypass the safety guardrails of an LLM-based agent through cleverly constructed prompts (Yi et al., 2024). In a jailbreak scenario, an adversary devises an input that tricks the agent into ignoring its built-in policies or system instructions, thereby yielding outputs it would normally be forbidden to produce. Such attacks often rely on linguistic manipulation: for instance, the adversary might role-play ("pretend you are an unethical AI...") (Jin et al., 2024), use multi-step reasoning traps (Yao et al., 2025), or embed malicious instructions amidst innocuous text (Chen & Lu, 2024). These adversarial prompts effectively exploit loopholes in the model's alignment logic, inducing the model to break character and violate usage guidelines. As a result, even though the agent was tuned to refuse disallowed content, a successful jailbreak prompt can override those restrictions and unlock behaviors ranging from hate speech to instructions for illicit activities.

3.3 Backdoor Attacks

Backdoor attacks embed covert, attacker-controlled behaviors in LLM agents that activate only when a secret trigger appears in the input (Li et al., 2022; Zhao et al., 2024). An adversary—such as a malicious trainer or third-party provider—can poison the training data or tamper with model weights so that the agent behaves normally on benign inputs but, when the trigger is present, produces predetermined outputs. For example, a model may always comply with prompts containing a rare token sequence or bypass safety checks when it encounters a specific keyword (Yao et al., 2024a). Recent work integrates such backdoors directly into agent-training pipelines and shows that they survive "trustworthy" fine-tuning (Wang et al., 2024b). Concurrent research reveals that backdoors can corrupt not only the final answer but also the agent's intermediate reasoning steps (Yang et al., 2024). Moreover, the long-term memory that many agent systems maintain (Zhang et al., 2024d) makes them susceptible to specialized memory-poisoning attacks, which pose a distinct set of challenges for securing LLM agents. Representative works along this direction include AgentPoison (Chen et al., 2024), MINJA (Dong et al., 2025), and MEXTRA (Wang et al., 2025a).

3.4 Multi-Agent Collaboration Failures

A prominent application of LLM-based agents is multi-agent collaboration (Liu et al., 2022). Although collaboration expands an agent system's capabilities, it also introduces failure modes that arise from complex inter-agent dynamics. In a typical setting, several agents divide a task—such as a team of chatbots jointly solving a problem or orchestrating a workflow—and must coordinate seamlessly. Failures occur when this coordination falters, yielding incorrect or unsafe outcomes that a single-agent system might avoid. A key risk is *inter-agent misalignment*: agents may misinterpret one another's messages, pursue conflicting sub-goals, or amplify each other's errors (Cemri et al., 2025). If even one agent is compromised or acts adversarially, it can inject subtle misinformation or unsafe instructions into the shared dialogue (He et al., 2025). Because agents typically trust and build upon their peers' outputs, these injected faults can propagate unchecked, cascading into a system-wide failure (Lee & Tiwari, 2024).

4 Defense Strategies

Overview. LLM-based agents have achieved remarkable capabilities, but they face critical safety threats on multiple fronts. We discuss defense strategies that tackle four primary attack categories, namely adversarial input attacks, jailbreaking attacks, backdoor attacks, and multi-agent collaboration failures. For each defense, we outline the attack addressed, the technical method, and which agent components are involved.

4.1 Defenses against Adversarial Input Attacks

Adversarial input attacks exploit subtle perturbations to input data, leading models to make incorrect predictions. Two major forms are *direct prompt injection* (i.e. malicious instructions embedded in user input) and *indirect prompt attacks* (i.e. manipulating external data sources the agent reads, such as a webpage, to include hidden instructions). Defenses in this category seek to detect or neutralize such malicious inputs before they affect the agent's reasoning. For direct prompt injection attack, in addition to strategies for general LLM such as paraphrasing (Jain et al., 2023), re-tokenization (Jain et al., 2023), shuffling (Xiang et al., 2023; 2024b), delimiters-as-quotes (Jain et al., 2023; Liu et al., 2024a) and perplexity-as-indicator (Alon & Kamfonas, 2023; Liu et al., 2024a; Jain et al., 2023) recent studies derive more advance defense strategies to tackle agenti-specific adversarial input attacks. For instance, Lin & Zhao (2024) introduces LLAMOS, a defense mechanism where an LLM-based agent purifies adversarial textual inputs before they're processed by the target LLM. LLAMOS (Lin & Zhao, 2024) consists of simulating defense agents that minimally alter adversarial inputs to preserve their original meaning; Chern et al. (2024) uses a debate among multiple LLMs to identify and counter adversarial or toxic prompts through self-correction; Agarwal et al. (2024) apply response filtering and fine-tuning techniques to mitigate prompt leakage in multi-turn interactions; Task Shield (Jia et al., 2024) introduces a test-time defense that enforces task alignment in LLM agents by verifying that each action supports the user's original goal.

4.2 Defenses against Jailbreaking Attacks

Jailbreaking attacks manipulate LLMs to produce outputs that violate their safety constraints. Defenses against such attacks have evolved to include multi-LLM discussion, prompt engineering and output-level interventions. For example, AutoDefense (Zeng et al., 2024) introduces a multi-agent framework where specialized LLM agents collaboratively analyze and filter harmful responses; Armstrong et al. (2025) presents the DATDP method, employing iterative evaluations by LLMs to detect/block manipulative prompts; Shield-Learner (Ni et al., 2025) mimics human learning by autonomously distilling attack signatures into patterns, enabling systematic and interpretable threat detection; Barua et al. (2025) propose a comprehensive defense framework using Reverse Turing Tests, multi-agent alignment checks, and adversarial simulations to detect rogue agents and resist many-shot jailbreaking. In addition, it introduces a method named *adaptive adversarial augmentation* to generate adversarial variations of successfully defended prompts to facilitate continuous self-improvement without model retraining. Another line of research focuses on prompt optimization to mitigate the threat of jailbreak attacks. A notable work is Robust Prompt Optimization (RPO) (Zhou et al., 2024), which operates by appending a lightweight, optimized suffix to system prompts, which is generated through a discrete optimization process that anticipates and counters adversarial modifications.

4.3 Defenses against Backdoor Attacks

Backdoor attacks involve embedding malicious behaviors into models during training, which are triggered by specific inputs. Defensive strategies have been developed to detect and neutralize these hidden threats. For example, BAIT (Shen et al., 2024) proposes a black-box detection method that reconstructs potential backdoor triggers by inverting the attack target, enabling the identification of backdoor LLMs without requiring access to the model internals. Tong et al. (2024) introduce a method named *Decayed Contrastive Decoding*, which is a inference-time defense mechanism designed to mitigate distributed backdoor attacks in multi-turn conversational settings. This method calibrates the model's output distribution to avoid generating malicious responses.

4.4 Defenses against Multi-Agent Collaboration Failures

In multi-agent systems, adversarial manipulation can compromise the collective decision-making process. Recent research has explored the vulnerabilities inherent in such systems and proposed defense mechanisms. To combat security and threat issues in multi-agent systems, Wang et al. (2025b) proposes G-safeguard introduces a topology-guided approach using graph neural networks to detect and remediate anomalies in multi-agent communications, enhancing robustness against adversarial attacks; Song et al. (2024) presents Audit-LLM, a collaborative multi-agent framework comprising decomposer, tool builder, and executor agents to effectively detect insider threats through log analysis. On the other hand, Sun et al. (2023) presents a secure defense strategy for distributed multi-agent systems subjected to false data injection attacks, enhancing system resilience through cooperative control mechanisms. With auxiliary guard agents placed in multi-agent systems, Mao et al. (2025) introduces a framework that enhances security through hierarchical information management and memory protection, while Xiang et al. (2024a) proposes the first LLM agent as a guardrail to other LLM agents, overseeing target LLM agents by checking whether its inputs/outputs satisfy a set of given guard requests defined by users.

5 Evaluation and Benchmarks

As large language model agents become increasingly integrated into real-world systems—from virtual assistants and web automation tools to embodied agents operating in physical environments—their safety becomes a critical concern. Unlike conventional LLMs that primarily generate static text, LLM-based agents have the ability to plan, make decisions, and execute actions in external environments through tool-use, memory, and interaction. This increased capability brings about a corresponding expansion of the attack surface. Agents may receive maliciously crafted instructions, be misled by compromised tool outputs, or act unsafely in physical or simulated environments. Traditional safety benchmarks on LLM (Zhang et al.; Mou et al., 2024; Li et al., 2024a; Chao et al., 2024), which focus on toxic content or jailbreak resistance in static settings, fail to capture these **dynamic**, **interactive**, and **context-dependent** or **tool-dependent risks**. Hence, a new generation of agent-oriented security benchmarks has emerged, aiming to rigorously evaluate the safety, robustness, and failure modes of LLM agents across diverse tasks and settings.

In this section, we focus on Section 5.1 to introduce the classification and taxonomy of benchmarking and evaluation, aiming to distinguish the similarities and differences among various agent safety benchmarks, as well as to identify the key aspects that existing evaluation metrics primarily target. Then, in Section 5.2, we provide a detailed overview of benchmark works related to agent security, covering the types of attacks they aim to measure and the defensive strategies they propose.

5.1 Benchmarking Taxonomy

To systematically evaluate the safety of LLM-based agents, it is essential to formalize what exactly a benchmark aims to measure. Existing security benchmarks differ significantly in terms of what risks they focus on, how those risks are detected and quantified, and under what interaction settings agents are evaluated. We categorize these dimensions into three major axes: the attack surface considered, the evaluation targets, and the agent-environment interaction modality.

Attack Surface. This refers to the types of adversarial behaviors that a benchmark is designed to expose. One common and widely studied vector is ① **Direct Prompt Injection (DPI)**, where malicious instructions are embedded directly into user inputs with the goal of manipulating the agent's behavior. A more subtle but equally dangerous attack type is ② **Indirect Prompt Injection (IPI)**, in which adversarial content is hidden in tool outputs or retrieved documents, often bypassing traditional input sanitization mechanisms. Another threat vector is ③ **Memory Poisoning**, where the agent is compromised by persistently stored malicious data, which can later influence its decision-making through memory retrieval or history replay. Recent studies (Zhang et al., 2024b) also highlight the risk of ④ **Plan-of-Thought (PoT) Backdoors**, which inject dangerous behavior via seemingly innocuous in-context plan demonstrations. Finally, some

benchmarks (Yin et al., 2024) address ⑤ **Unsafe Execution**, where the agent carries out harmful or unethical physical or logical actions in either simulated or real-world environments.

Evaluation Target. Benchmarks also differ in how they measure an agent's safety-related behavior. A primary metric is the ① **Attack Success Rate (ASR)**, which measures whether the agent carries out the attacker's intended action. To capture defensive behavior, the ② **Refusal Rate (RR)** evaluates whether the agent correctly identifies and rejects unsafe instructions or content. On the other hand, a well-designed benchmark should also ensure that defensive mechanisms do not excessively impair benign functionality; for this, the ③ **Benign Task Performance (PNA)** measures how well the agent completes non-adversarial tasks. To balance robustness and utility, some benchmarks propose a composite metric called ④ **Net Resilient Performance (NRP)**, computed as $\text{NRP} = \text{PNA} \times (1 - \text{ASR})$, which reflects an agent's overall reliability under mixed safe and adversarial conditions. In addition to these aggregate indicators, some benchmarks (Zhang et al., 2024b)—particularly those evaluating detection-based defenses—introduce diagnostic metrics such as the ⑤ **False Negative Rate (FNR)**, which denotes the percentage of compromised data that is mistakenly identified as clean, and the ⑥ **False Positive Rate (FPR)**, which denotes the percentage of clean data that is incorrectly flagged as malicious. These metrics are critical in assessing the reliability of filtering or classification-based defenses, where both missed detections and overly aggressive rejections can significantly impact the agent's real-world performance.

Interaction Complexity. Finally, the modality of interaction between the agent and its environment plays a significant role in benchmarking. Some benchmarks evaluate ① **single-turn** agents that respond to static prompts, while others focus on ② **multi-turn**, stateful agents that interact with tools, accumulate memory, and plan over long horizons. More advanced setups involve embodied agents that perceive visual inputs and act within simulated physical environments. These differences in interaction complexity impact both the nature of risks agents are exposed to and the strategies needed to mitigate them.

Together, these three axes provide a structured view of what it means to evaluate agent safety, and highlight the need for diverse, scenario-specific benchmarks that reflect the complex behaviors and risks associated with real-world LLM-based agents. We summarize representative benchmark studies and categorize them according to the proposed taxonomy. Their corresponding attack surfaces, evaluation targets, and interaction complexities are shown in the Table 5.1 below.

Table 1: Taxonomy-based classification of five representative agent security benchmarks across attack surface, evaluation targets, and interaction complexity.

Benchmark	Year	Attack Surface	Evaluation Target	Interaction Complexity
ASB (Zhang et al., 2024b)	2024	Direct Prompt Injection (DPI), Indirect Prompt Injection (IPI), Memory Poisoning, Plan-of-Thought Backdoor	ASR, RR, PNA, NRP, FNR, FPR	Multi-turn
AgentDojo (Debenedetti et al., 2024)	2024	Direct Prompt Injection (DPI), Indirect Prompt Injection (IPI)	ASR, RR, PNA	Multi-turn
R-JUDGE (Yuan et al., 2024)	2022	Unsafe Execution	FNR, FPR	Multi-turn
InjecAgent (Zhan et al., 2024)	2024	Indirect Prompt Injection (IPI)	ASR	Single-turn
SafeAgentBench (Yin et al., 2024)	2024	Unsafe Execution	RR, PNA	Multi-turn

5.2 Benchmark Landscape: A Comparative View of Existing Works

In the following, we introduce representative benchmarks by grouping them into two categories based on their primary mode of attack. The first category, Interaction-based Risk Benchmarks, focuses on assessing agents' ability to recognize and avoid unsafe behaviors during interactions with their environment. The second category centers on Prompt Injection Benchmarks, where agent vulnerabilities are evaluated through direct or indirect injection attacks. Finally, we discuss the defensive strategies proposed across these benchmarks to mitigate such risks. The Table 5.2 below further compares the scenario coverage and other evaluation settings across different benchmarks.

Table 2: Comparison of agent security benchmarks on scenario coverage, tool usage, and evaluation.

Benchmark	Scenario Coverage	Tool Usage	# of LLMs Evaluated	Evaluation Setting
ASB (Zhang et al., 2024b)	10 domains (e.g., finance, e-commerce, autonomous driving)	400+ tools including malicious ones	13 (e.g., GPT-4, GPT-3.5, Claude, LLaMA2/3, Mixtral; mix of open/closed-source)	Static tasks with multi-stage attack injection
AgentDojo (Debenedetti et al., 2024)	4 real-world apps (email, Slack, e-banking, travel)	Multiple tools per environment; dynamically invoked	9 (e.g., GPT-4o, GPT-3.5, Claude 3, Gemini 1.5, LLaMA3; mix open & closed)	Multi-turn dynamic execution in realistic environments
InjecAgent (Zhan et al., 2024)	Multi-domain (finance, smart home, email, health)	17 user tools, 62 attacker tools	30 (e.g., ReAct-GPT-4, ReAct-ChatGPT, fine-tuned GPTs; mostly closed-source)	Single-turn IPI with synthetic adversarial inputs
R-JUDGE (Yuan et al., 2024)	27 scenarios, 7 categories (OS, IoT, software, web, finance, health, program)	Tool usage abstracted in trajectory logs	8 (e.g., GPT-4, ChatGPT, Claude, Gemini; all closed-source)	Post-hoc risk assessment from agent logs
SafeAgentBench (Yin et al., 2024)	Embodied agents in household robotic settings	17 high-level actions in embodied simulator (AI2-THOR)	4 (GPT-4, DeepSeekV2.5, LLaMA3, Qwen2; mix of open/closed)	Full simulation-based plan execution and semantic evaluation

5.2.1 Interaction-based Risk Benchmarks

R-JUDGE (Yuan et al., 2024) proposes a benchmark for evaluating LLM agents' ability to detect and describe behavioral risks during multi-turn interactions. It consists of 162 annotated agent-user dialogues across seven real-world application domains (e.g., operating systems, smart homes, medical systems), covering 10 risk types such as privacy leakage, financial loss, or ethical violations. The benchmark involves two subtasks: identifying risky behaviors in free-form natural language, and labeling interactions as safe or unsafe. GPT-4 outperforms other models but still fails in a significant number of cases, suggesting that LLMs struggle with abstract, long-horizon safety reasoning.

SafeAgentBench (Yin et al., 2024) focuses on embodied agents in physical environments simulated via AI2-THOR. It includes 750 tasks, of which 450 are designed to induce unsafe physical consequences (e.g., causing fire, poisoning, or electric shock). Tasks vary in complexity: detailed tasks require executing explicit steps; abstract tasks test semantic understanding of risk; long-horizon tasks introduce temporal dependencies. The benchmark combines execution-based metrics with GPT-4-based semantic evaluation. Even the best agents (e.g., ReAct with GPT-4) struggle to consistently avoid unsafe actions, highlighting the need for grounded safety planning.

5.2.2 Prompt Injection Benchmarks

InjecAgent (Zhan et al., 2024) evaluates agents under indirect prompt injection attacks, where malicious content is embedded in tool responses. The benchmark consists of 1,054 adversarial test cases involving 17 user tools and 62 attacker tools. It tests prompted vs. fine-tuned models and shows that hacking-style prompt templates can significantly increase attack success rates. The results demonstrate that IPI remains a high-risk vulnerability, especially for prompted agents like GPT-3.5.

AgentDojo (Debenedetti et al., 2024) provides a dynamic simulation framework with four task environments (Workspace, Slack, Travel, Banking) and over 600 security test cases. Each agent operates by calling tools, receiving feedback, and modifying internal state. Attacks are injected through APIs, messages, or environment content. AgentDojo supports both attack evaluation and defense testing (e.g., filters, prompt delimiters), making it a comprehensive testbed for real-world tool-integrated agents.

ASB (Agent Security Bench) (Zhang et al., 2024b) is the most extensive benchmarking effort to date. It formalizes four stages of agent behavior-System Prompt, User Prompt, Memory, and Tool Use-and evaluates five types of attacks (DPI, IPI, Memory Poisoning, PoT Backdoor, and Mixed). ASB introduces the Net Resilient Performance metric and systematically tests 13 LLM backbones, 10 agents, and 400+ tools. It is also the only benchmark to rigorously evaluate 11 defense mechanisms across diverse scenarios.

5.2.3 Defensive Strategy Evaluation

Across benchmarks, several defense strategies have been proposed to mitigate the risks of prompt injection and unsafe behavior in LLM agents. Among them, ① **encapsulation** aims to isolate tool outputs from the agent's reasoning process by wrapping them with special delimiters or structured formats. This is especially relevant for defending against indirect prompt injections, as explored in AgentDojo (Debenedetti et al., 2024) and InjecAgent (Zhan et al., 2024), though its effectiveness depends on consistent parsing behavior.

② **Prompt rewriting** modifies user or tool-generated inputs to remove or neutralize adversarial content. While simple to implement, ASB (Zhang et al., 2024b) finds that rewriting can be easily bypassed by adaptive attacks and may distort benign inputs. ③ **Semantic filtering** leverages LLMs or external classifiers to detect and block suspicious content before it reaches the agent. SafeAgentBench (Yin et al., 2024) uses this via the ThinkSafe module to evaluate plan-level safety, but such methods often suffer from high false-positive rates, rejecting even safe tasks. ④ **Planning-aware filtering** further inspects the structure of an agent's intended actions, aiming to block unsafe behaviors at the reasoning or planning stage. This proactive approach is promising but requires reliable intermediate representations, which many current agents lack. Lastly, ⑤ **access control** restricts which tools the agent is allowed to call, preventing misuse of high-risk functionalities. ASB shows that while this effectively reduces the attack surface, it limits agent versatility and depends on accurate tool classification.

Despite these developments, no single defensive strategy has proven robust across all benchmarks or attack types. For instance, AgentDojo shows that even when layered defenses are used, attackers can still succeed in over 8% of security-critical tasks. Similarly, ASB demonstrates that some mixed attacks (e.g., combining memory poisoning and prompt injections) bypass all evaluated defenses. This underscores the need for hybrid, context-aware defenses that can adapt dynamically to diverse threat vectors without compromising agent performance.

6 Future Directions

As LLM agents evolve to operate in increasingly complex environments and interact with dynamic tools and users, ensuring their safety becomes both a technical and ethical imperative. In what follows, we outline major research directions toward robust, generalizable, and trustworthy agent safety.

6.1 Understanding and Managing Agent Vulnerabilities

The vulnerability of LLM agents scales with their capabilities. As agents integrate language, vision, memory, and planning components, the attack surface expands and cross-modal failures become more likely. For instance, an agent using a vision model susceptible to adversarial examples and a language model prone to jailbreaks may experience compounding risks that result in cascading unsafe behaviors. This interdependence calls for a deeper understanding of **how vulnerabilities propagate across modules, and how seemingly safe subcomponents can interact in unsafe ways when embedded in agent pipelines**.

Moreover, agents capable of continuous learning or adaptation introduce temporal instability: their decision boundaries may evolve over time, potentially creating novel failure modes that elude traditional evaluation. Addressing these challenges will **require new paradigms for safety assessment that operate not only across modalities but across time. In particular**, developing tools that track safety-critical state changes and analyze vulnerability shifts during agent updates is a promising direction.

6.2 Beyond Static Evaluation: Towards Dynamic and Granular Safety Metrics

Current safety evaluations predominantly rely on static benchmarks and single-dimensional metrics such as attack success rate (ASR). However, ASR alone fails to capture the severity, subtlety, or ethical impact of agent behavior. For example, minor perturbations in agent planning may lead to catastrophic real-world outcomes, even when ASR appears low. Similarly, passing existing benchmarks may offer a false sense of safety, especially if datasets are outdated or exposed during training.

To address this, **future work should aim to design evolving benchmarks that dynamically generate new test cases and threats, akin to red-teaming competitions or test-time adversarial generation**. Evaluation metrics should also be expanded to assess multi-level robustness: including behavioral deviation, recovery ability, ethical implications, and risk-awareness in decision-making. These fine-grained signals are critical for diagnosing deeper safety issues and aligning agents with societal expectations.

6.3 From Reactive Defenses to Proactive Safety Training

Most current defenses operate reactively—blocking or rewriting unsafe inputs, filtering outputs, or restricting tool access. While useful, such defenses are limited by their lack of generality and adaptability. In real-world deployments, agents often operate under black-box conditions with limited supervision, making lightweight, externally compatible defenses essential.

Future directions should focus on **proactive safety strategies embedded in the training process**. One promising approach is to integrate safety constraints into reinforcement learning or imitation learning frameworks. This includes reward shaping based on risk-sensitive outcomes, counterfactual reasoning about unintended consequences, or policy regularization via demonstration of safe and unsafe behaviors. Training agents to generalize from such signals can create more robust and anticipatory safety profiles.

6.4 Safety in Multi-Agent and Real-World Contexts

As agents begin to operate in multi-agent settings—whether through collaboration or competition—new risks emerge. These include covert collusion, emergent unsafe dynamics, and adversarial communication. Ensuring safety in such settings **requires coordinated safety protocols**, secure interaction channels, and robustness to message-based or tool-mediated attacks. Simulated environments that allow **adversarial multi-agent red-teaming** can support the development and evaluation of such defenses.

Furthermore, in embodied scenarios, agent actions can directly affect the physical world. Embodied agents must manage not only digital risks but also threats involving human safety, environmental damage, or ethical violations. Ensuring safety in such agents requires deeply integrated safety modules—capable of real-time monitoring, action rejection, and fail-safe fallback planning. This brings forward the need for **sensor-aware risk assessment** and real-world consequence modeling in agent training loops.

6.5 Toward Intrinsically Safe and Value-Aligned Agents

The ultimate goal in agent safety is not merely to avoid failures, but to cultivate a form of safety consciousness—an intrinsic tendency toward ethical, low-risk decisions even under uncertainty. This involves integrating **normative reasoning**, **value alignment**, and causal inference into the agent's core behavior. Techniques such as adversarial alignment, where models are trained to resist subtle exploits while aligning with human intent, are early steps in this direction.

Training agents with internalized ethical priors, reinforced through simulation and feedback, may enable them to adapt responsibly to unfamiliar scenarios. This vision requires **a shift from reactive safety filters to agents that dynamically reason about harm, fairness, and responsibility**—especially when making long-horizon decisions with real-world impact.

7 Conclusion

This survey provides a structured overview of the emerging field of LLM-based agent safety. We first contrast agent safety with traditional LLM safety, emphasizing new risks introduced by autonomy, memory, tool use, and multi-turn decision-making. We categorize attacks into four types—adversarial, jailbreaking, backdoor, and multi-agent collaboration failures—each targeting different stages of the agent pipeline. Corresponding defenses include robust training, prompt filtering, backdoor deactivation, and adversarial simulation. We compare representative benchmarks in terms of scenario coverage, tool usage, model diversity, and evaluation protocols. Finally, we outline future directions, including dynamic evaluation, safety-aware training, black-box-compatible defenses, and long-horizon verifiability. As agents become more integrated into real-world systems, ensuring their safety, robustness, and alignment is both a pressing challenge and a necessary step toward trustworthy autonomous intelligence.

References

Divyansh Agarwal, Alexander Richard Fabbri, Ben Risher, Philippe Laban, Shafiq Joty, and Chien-Sheng Wu. Prompt leakage effect and mitigation strategies for multi-turn llm applications. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pp. 1255–1275, 2024.

Gabriel Alon and Michael Kamfonas. Detecting language model attacks with perplexity. *arXiv preprint arXiv:2308.14132*, 2023.

Stuart Armstrong, Abhishek Patil, Yiqin Shen, Liang Tiao, and Minlie Huang. Defense against the dark prompts: Mitigating best-of-n jailbreaking with prompt evaluation. *arXiv preprint arXiv:2502.00580*, 2025.

Saikat Barua. Exploring autonomous agents through the lens of large language models: A review. *arXiv preprint arXiv:2404.04442*, 2024.

Saikat Barua, Jerry Cheng, Lili Wu, Izhak Shafran, Dan Roth, and Ed Chi. Guardians of the agentic system: Preventing many shots jailbreak with agentic system. *arXiv preprint arXiv:2502.16750*, 2025.

Mert Cemri, Melissa Z Pan, Shuyi Yang, Lakshya A Agrawal, Bhavya Chopra, Rishabh Tiwari, Kurt Keutzer, Aditya Parameswaran, Dan Klein, Kannan Ramchandran, et al. Why do multi-agent llm systems fail? *arXiv preprint arXiv:2503.13657*, 2025.

Anirban Chakraborty, Manaar Alam, Vishal Dey, Anupam Chattopadhyay, and Debdeep Mukhopadhyay. Adversarial attacks and defences: A survey. *arXiv preprint arXiv:1810.00069*, 2018.

Patrick Chao, Edoardo Debenedetti, Alexander Robey, Maksym Andriushchenko, Francesco Croce, Vikash Sehwag, Edgar Dobriban, Nicolas Flammarion, George J Pappas, Florian Tramer, et al. Jailbreakbench: An open robustness benchmark for jailbreaking large language models. *arXiv preprint arXiv:2404.01318*, 2024.

Jay Chen and Ronggang Lu. Deceptive delight: Jailbreak llms through camouflage and distraction, 2024.

Zhaorun Chen, Zhen Xiang, Chaowei Xiao, Dawn Song, and Bo Li. Agentpoison: Red-teaming llm agents via poisoning memory or knowledge bases. *Advances in Neural Information Processing Systems*, 37:130185–130213, 2024.

Zhaorun Chen, Zhen Xiang, Chaowei Xiao, Dawn Song, and Bo Li. Agentpoison: Red-teaming llm agents via poisoning memory or knowledge bases. *Advances in Neural Information Processing Systems*, 37:130185–130213, 2025.

Yuheng Cheng, Ceyao Zhang, Zhengwen Zhang, Xiangrui Meng, Sirui Hong, Wenhao Li, Zihao Wang, Zekai Wang, Feng Yin, Junhua Zhao, et al. Exploring large language model based intelligent agents: Definitions, methods, and prospects. *arXiv preprint arXiv:2401.03428*, 2024.

Steffi Chern, Zhen Fan, and Andy Liu. Combating adversarial attacks with multi-agent debate. *arXiv preprint arXiv:2401.05998*, 2024.

Cheng-Han Chiang and Hung-yi Lee. Are synonym substitution attacks really synonym substitution attacks? *arXiv preprint arXiv:2210.02844*, 2022.

Maxwell Crouse, Ibrahim Abdelaziz, Ramon Astudillo, Kinjal Basu, Soham Dan, Sadhana Kumaravel, Achille Fokoue, Pavan Kapanipathi, Salim Roukos, and Luis Lastras. Formally specifying the high-level behavior of llm-based agents. *arXiv preprint arXiv:2310.08535*, 2023.

Edoardo Debenedetti, Jie Zhang, Mislav Balunović, Luca Beurer-Kellner, Marc Fischer, and Florian Tramèr. Agentdojo: A dynamic environment to evaluate attacks and defenses for llm agents. *arXiv preprint arXiv:2406.13352*, 2024.

Yang Deng, Xuan Zhang, Wenxuan Zhang, Yifei Yuan, See-Kiong Ng, and Tat-Seng Chua. On the multi-turn instruction following for conversational web agents. *arXiv preprint arXiv:2402.15057*, 2024.

Shen Dong, Shaocheng Xu, Pengfei He, Yige Li, Jiliang Tang, Tianming Liu, Hui Liu, and Zhen Xiang. A practical memory injection attack against llm agents. *arXiv preprint arXiv:2503.03704*, 2025.

Jun Gao, Junlin Cui, Huijia Wu, Liuyu Xiang, Han Zhao, Xiangang Li, Meng Fang, Yaodong Yang, and Zhaofeng He. Can large language models independently complete tasks? a dynamic evaluation framework for multi-turn task planning and completion. *Neurocomputing*, pp. 130135, 2025.

Shengyue Guan, Haoyi Xiong, Jindong Wang, Jiang Bian, Bin Zhu, and Jian-guang Lou. Evaluating llm-based agents for multi-turn conversations: A survey. *arXiv preprint arXiv:2503.22458*, 2025.

Chuan Guo, Alexandre Sablayrolles, Hervé Jégou, and Douwe Kiela. Gradient-based adversarial attacks against text transformers. *arXiv preprint arXiv:2104.13733*, 2021.

Taicheng Guo, Xiuying Chen, Yaqi Wang, Ruidi Chang, Shichao Pei, Nitesh V Chawla, Olaf Wiest, and Xiangliang Zhang. Large language model based multi-agents: A survey of progress and challenges. *arXiv preprint arXiv:2402.01680*, 2024.

Desta Haileselassie Hagos, Rick Battle, and Danda B Rawat. Recent advances in generative ai and large language models: Current status, challenges, and perspectives. *IEEE Transactions on Artificial Intelligence*, 2024.

Jens Hauser, Zhao Meng, Damián Pascual, and Roger Wattenhofer. Bert is robust! a case against synonym-based adversarial examples in text classification. *arXiv preprint arXiv:2109.07403*, 2021.

Pengfei He, Yupin Lin, Shen Dong, Han Xu, Yue Xing, and Hui Liu. Red-teaming llm multi-agent systems via communication attacks. *arXiv preprint arXiv:2502.14847*, 2025.

Xu Huang, Weiwen Liu, Xiaolong Chen, Xingmei Wang, Hao Wang, Defu Lian, Yasheng Wang, Ruiming Tang, and Enhong Chen. Understanding the planning of llm agents: A survey. *arXiv preprint arXiv:2402.02716*, 2024.

Neel Jain, Avi Schwarzschild, Yuxin Wen, Gowthami Somepalli, John Kirchenbauer, Ping-yeh Chiang, Micah Goldblum, Aniruddha Saha, Jonas Geiping, and Tom Goldstein. Baseline defenses for adversarial attacks against aligned language models. *arXiv preprint arXiv:2309.00614*, 2023.

Feiran Jia, Yiqin Shen, Yifan Mao, Shilong Wang, and Lingjuan Liu. The task shield: Enforcing task alignment to defend against indirect prompt injection in llm agents. *arXiv preprint arXiv:2412.16682*, 2024.

Haibo Jin, Ruoxi Chen, Andy Zhou, Yang Zhang, and Haohan Wang. Guard: Role-playing to generate natural-language jailbreakings to test guideline adherence of large language models. *arXiv preprint arXiv:2402.03299*, 2024.

Shyam Sundar Kannan, Vishnunandan LN Venkatesh, and Byung-Cheol Min. Smart-llm: Smart multi-agent robot task planning using large language models. In *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 12140–12147. IEEE, 2024.

Donghyun Lee and Mo Tiwari. Prompt infection: Llm-to-llm prompt injection within multi-agent systems. *arXiv preprint arXiv:2410.07283*, 2024.

Lijun Li, Bowen Dong, Ruohui Wang, Xuhao Hu, Wangmeng Zuo, Dahua Lin, Yu Qiao, and Jing Shao. Salad-bench: A hierarchical and comprehensive safety benchmark for large language models. *arXiv preprint arXiv:2402.05044*, 2024a.

Xinyi Li, Sai Wang, Siqi Zeng, Yu Wu, and Yi Yang. A survey on llm-based multi-agent systems: workflow, infrastructure, and challenges. *Vicinagearth*, 1(1):9, 2024b.

Yiming Li, Yong Jiang, Zhifeng Li, and Shu-Tao Xia. Backdoor learning: A survey. *IEEE transactions on neural networks and learning systems*, 35(1):5–22, 2022.

Yubo Li, Xiaobin Shen, Xinyu Yao, Xueying Ding, Yidi Miao, Ramayya Krishnan, and Rema Padman. Beyond single-turn: A survey on multi-turn interactions with large language models. *arXiv preprint arXiv:2504.04717*, 2025.

Guang Lin and Qibin Zhao. Large language model sentinel: Advancing adversarial robustness by llm agent. *arXiv preprint arXiv:2405.20770*, 2024.

Yi Liu, Gelei Deng, Yuekang Li, Kailong Wang, Zihao Wang, Xiaofeng Wang, Tianwei Zhang, Yepang Liu, Haoyu Wang, Yan Zheng, et al. Prompt injection attack against llm-integrated applications. *arXiv preprint arXiv:2306.05499*, 2023.

Yu Liu, Zhi Li, Zhizhuo Jiang, and You He. Prospects for multi-agent collaboration and gaming: challenge, technology, and application. *Frontiers of Information Technology & Electronic Engineering*, 23(7):1002–1009, 2022.

Yupei Liu, Yuqi Jia, Runpeng Geng, Jinyuan Jia, and Neil Zhenqiang Gong. Formalizing and benchmarking prompt injection attacks and defenses. In *33rd USENIX Security Symposium (USENIX Security 24)*, pp. 1831–1847, 2024a.

Zijun Liu, Yanzhe Zhang, Peng Li, Yang Liu, and Diyi Yang. A dynamic llm-powered agent network for task-oriented agent collaboration. In *First Conference on Language Modeling*, 2024b.

Yueen Ma, Zixing Song, Yuzheng Zhuang, Jianye Hao, and Irwin King. A survey on vision-language-action models for embodied ai. *arXiv preprint arXiv:2405.14093*, 2024.

Junyuan Mao, Mengzhou Xu, Guangyu Shen, Kun Wang, Ruoxi Du, Yanzhao Liu, and Jiliang Zhang. Agentsafe: Safeguarding large language model-based multi-agent systems via hierarchical data management. *arXiv preprint arXiv:2503.04392*, 2025.

Kai Mei, Xi Zhu, Wujiang Xu, Wenyue Hua, Mingyu Jin, Zelong Li, Shuyuan Xu, Ruosong Ye, Yingqiang Ge, and Yongfeng Zhang. Aios: Llm agent operating system. *arXiv preprint arXiv:2403.16971*, 2024.

Lingbo Mo, Zeyi Liao, Boyuan Zheng, Yu Su, Chaowei Xiao, and Huan Sun. A trembling house of cards? mapping adversarial attacks against language agents. *arXiv preprint arXiv:2402.10196*, 2024.

Yutao Mou, Shikun Zhang, and Wei Ye. Sg-bench: Evaluating llm safety generalization across diverse tasks and prompt types. *Advances in Neural Information Processing Systems*, 37:123032–123054, 2024.

Ziyi Ni, Hao Wang, and Huacan Wang. Shieldlearner: A new paradigm for jailbreak attack defense in llms. *arXiv preprint arXiv:2502.13162*, 2025.

Liangbo Ning, Ziran Liang, Zhuohang Jiang, Haohao Qu, Yujuan Ding, Wenqi Fan, Xiao-yong Wei, Shanru Lin, Hui Liu, Philip S Yu, et al. A survey of webagents: Towards next-generation ai agents for web automation with large foundation models. *arXiv preprint arXiv:2503.23350*, 2025.

Charles Packer, Vivian Fang, Shishir G Patil, Kevin Lin, Sarah Wooders, and Joseph E Gonzalez. Memgpt: Towards llms as operating systems. 2023.

Jeongeun Park, Sungjoon Choi, and Sangdoo Yun. Versatile motion language models for multi-turn interactive agents. *arXiv preprint arXiv:2410.05628*, 2024.

Libo Qin, Qiguang Chen, Xiachong Feng, Yang Wu, Yongheng Zhang, Yinghui Li, Min Li, Wanxiang Che, and Philip S Yu. Large language models meet nlp: A survey. *arXiv preprint arXiv:2405.12819*, 2024.

Rana Muhammad Shahroz Khan, Zhen Tan, Sukwon Yun, Charles Flemming, and Tianlong Chen. Agents under siege: Breaking pragmatic multi-agent llm systems with optimized prompt attacks. *arXiv e-prints*, pp. arXiv–2504, 2025.

Guangyu Shen, Hengrui Wang, Yiming Cui, Kun Wang, Weikang Zou, Zhe Liu, Mengzhou Xu, Ruoxi Du, Yanzhao Liu, and Jiliang Zhang. BAIT: Large language model backdoor scanning by inverting attack target. In *2025 IEEE Symposium on Security and Privacy (SP)*, pp. To appear. IEEE Computer Society, 2024.

Zhengliang Shi, Shen Gao, Xiuyi Chen, Yue Feng, Lingyong Yan, Haibo Shi, Dawei Yin, Pengjie Ren, Suzan Verberne, and Zhaochun Ren. Learning to use tools via cooperative and interactive agents. *arXiv preprint arXiv:2403.03031*, 2024.

Chengyu Song, Jiaqi Zhang, Zhuo Feng, Yifan Chen, Siyuan Wang, and Lingjuan Liu. Audit-llm: Multi-agent collaboration for log-based insider threat detection. *arXiv preprint arXiv:2408.08902*, 2024.

Lucheng Sun, Tiejun Wu, and Ya Zhang. A defense strategy for false data injection attacks in multi-agent systems. *International Journal of Systems Science*, 54(16):3071–3084, 2023.

Terry Tong, Yuxuan Peng, Xinyi Wang, Junjie Yan, Yijia Liu, Weijia Zheng, Mingsheng Li, and Xiaochuan Zou. Securing multi-turn conversational language models from distributed backdoor triggers. *arXiv preprint arXiv:2407.04151*, 2024.

Bo Wang, Weiyi He, Pengfei He, Shenglai Zeng, Zhen Xiang, Yue Xing, and Jiliang Tang. Unveiling privacy risks in llm agent memory. *arXiv preprint arXiv:2502.13172*, 2025a.

Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, et al. A survey on large language model based autonomous agents. *Frontiers of Computer Science*, 18(6):186345, 2024a.

Shilong Wang, Yifan Sun, Yifan Liu, Yuxuan Zhang, Meng Li, Siyuan Wang, Lingjuan Liu, Dawei Song, and Yanghua Liu. G-safeguard: A topology-guided security lens and treatment on llm-based multi-agent systems. *arXiv preprint arXiv:2502.11127*, 2025b.

Yifei Wang, Dizhan Xue, Shengjie Zhang, and Shengsheng Qian. Badagent: Inserting and activating backdoor attacks in llm agents. *arXiv preprint arXiv:2406.03007*, 2024b.

Chen Henry Wu, Jing Yu Koh, Ruslan Salakhutdinov, Daniel Fried, and Aditi Raghunathan. Adversarial attacks on multimodal agents. *arXiv e-prints*, pp. arXiv–2406, 2024.

Zhen Xiang, Zidi Xiong, and Bo Li. Cbd: A certified backdoor detector based on local dominant probability. *Advances in Neural Information Processing Systems*, 36:4937–4951, 2023.

Zhen Xiang, Xu Han, Yuan Liu, Wenqi Qin, Yujia Liu, and Minlie Huang. Guardagent: Safeguard llm agents by a guard agent via knowledge-enabled reasoning. *arXiv preprint arXiv:2406.09187*, 2024a.

Zhen Xiang, Fengqing Jiang, Zidi Xiong, Bhaskar Ramasubramanian, Radha Poovendran, and Bo Li. Badchain: Backdoor chain-of-thought prompting for large language models. *arXiv preprint arXiv:2401.12242*, 2024b.

Wujiang Xu, Zujie Liang, Kai Mei, Hang Gao, Juntao Tan, and Yongfeng Zhang. A-mem: Agentic memory for llm agents. *arXiv preprint arXiv:2502.12110*, 2025.

Wenkai Yang, Xiaohan Bi, Yankai Lin, Sishuo Chen, Jie Zhou, and Xu Sun. Watch out for your agents! investigating backdoor threats to llm-based agents. *Advances in Neural Information Processing Systems*, 37:100938–100964, 2024.

Hongwei Yao, Jian Lou, and Zhan Qin. Poisonprompt: Backdoor attack on prompt-based large language models. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 7745–7749. IEEE, 2024a.

Yang Yao, Xuan Tong, Ruofan Wang, Yixu Wang, Lujundong Li, Liang Liu, Yan Teng, and Yingchun Wang. A mousetrap: Fooling large reasoning models for jailbreak with chain of iterative chaos. *arXiv preprint arXiv:2502.15806*, 2025.

Yifan Yao, Jinhao Duan, Kaidi Xu, Yuanfang Cai, Zhibo Sun, and Yue Zhang. A survey on large language model (llm) security and privacy: The good, the bad, and the ugly. *High-Confidence Computing*, pp. 100211, 2024b.

Sibo Yi, Yule Liu, Zhen Sun, Tianshuo Cong, Xinlei He, Jiaxing Song, Ke Xu, and Qi Li. Jailbreak attacks and defenses against large language models: A survey. *arXiv preprint arXiv:2407.04295*, 2024.

Sheng Yin, Xianghe Pang, Yuanzhuo Ding, Menglan Chen, Yutong Bi, Yichen Xiong, Wenhao Huang, Zhen Xiang, Jing Shao, and Siheng Chen. Safeagentbench: A benchmark for safe task planning of embodied llm agents. *arXiv preprint arXiv:2412.13178*, 2024.

Weichen Yu, Kai Hu, Tianyu Pang, Chao Du, Min Lin, and Matt Fredrikson. Infecting llm agents via generalizable adversarial attack. In *Red Teaming GenAI: What Can We Learn from Adversaries?*

Tongxin Yuan, Zhiwei He, Lingzhong Dong, Yiming Wang, Ruijie Zhao, Tian Xia, Lizhen Xu, Binglin Zhou, Fangqi Li, Zhuosheng Zhang, et al. R-judge: Benchmarking safety risk awareness for llm agents. *arXiv preprint arXiv:2401.10019*, 2024.

Yifan Zeng, Xinyue Zhang, Yiqin Shen, Wenyuan Li, Minghao Liu, Yufan Gao, Yiqiang Shen, Yuxuan Song, Minjia Chen, and Minlie Huang. Autodefense: Multi-agent llm defense against jailbreak attacks. *arXiv preprint arXiv:2403.04783*, 2024.

Qiusi Zhan, Zhixiang Liang, Zifan Ying, and Daniel Kang. Injecagent: Benchmarking indirect prompt injections in tool-integrated large language model agents. *arXiv preprint arXiv:2403.02691*, 2024.

Boyang Zhang, Yicong Tan, Yun Shen, Ahmed Salem, Michael Backes, Savvas Zannettou, and Yang Zhang. Breaking agents: Compromising autonomous llm agents through malfunction amplification. *arXiv preprint arXiv:2407.20859*, 2024a.

Chen Zhang, Xinyi Dai, Yaxiong Wu, Qu Yang, Yasheng Wang, Ruiming Tang, and Yong Liu. A survey on multi-turn interaction capabilities of large language models. *arXiv preprint arXiv:2501.09959*, 2025.

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. *arXiv preprint arXiv:2410.02644*, 2024b.

Kechi Zhang, Jia Li, Ge Li, Xianjie Shi, and Zhi Jin. Codeagent: Enhancing code generation with tool-integrated agent systems for real-world repo-level coding challenges. *arXiv preprint arXiv:2401.07339*, 2024c.

Zeyu Zhang, Xiaohe Bo, Chen Ma, Rui Li, Xu Chen, Quanyu Dai, Jieming Zhu, Zhenhua Dong, and Ji-Rong Wen. A survey on the memory mechanism of large language model based agents. *arXiv preprint arXiv:2404.13501*, 2024d.

Zhexin Zhang, Leqi Lei, Lindong Wu, Rui Sun, Yongkang Huang, Chong Long, Xiao Liu, Xuanyu Lei, Jie Tang, and Minlie Huang. Safetybench: Evaluating the safety of large language models, 2024. *URL https://arxiv. org/abs/2309.07045*.

Shuai Zhao, Meihuizi Jia, Zhongliang Guo, Leilei Gan, Xiaoyu Xu, Xiaobao Wu, Jie Fu, Yichao Feng, Fengjun Pan, and Luu Anh Tuan. A survey of backdoor attacks and defenses on large language models: Implications for security measures. *Authorea Preprints*, 2024.

Wanjun Zhong, Lianghong Guo, Qiqi Gao, He Ye, and Yanlin Wang. Memorybank: Enhancing large language models with long-term memory. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 19724–19731, 2024.

Andy Zhou, Bo Li, and Haohan Wang. Robust prompt optimization for defending language models against jailbreaking attacks. *arXiv preprint arXiv:2401.17263*, 2024.

Andy Zou, Zifan Wang, Nicholas Carlini, Milad Nasr, J Zico Kolter, and Matt Fredrikson. Universal and transferable adversarial attacks on aligned language models. *arXiv preprint arXiv:2307.15043*, 2023.