

Connecting the Dots in Agentic AI Security: A Cross-Dimensional Threat Taxonomy, Evaluation Maturity, and Open Challenges

Heewon Baek, Alsharif Abuadbbba, Kristen Moore, Hyounghick Kim, Surya Nepal

Abstract—Agentic AI extends LLM security beyond generated content to persistent state, autonomous actions, tool use, and interactions with humans and other agents. Existing threat classifications often emphasize individual dimensions, obscuring connections among entry points, affected components, and security consequences. The known threat landscape also differs from the coverage demonstrated by empirical research.

Through a structured review of 66 studies published from 2022 to 2026, we introduce $\mathcal{T} = \langle \mathcal{S}, \mathcal{B}, \mathcal{P}, \mathcal{A} \rangle$, a cross-dimensional representation linking affected functional or system surfaces ($\mathcal{S}$), interaction or trust boundaries ($\mathcal{B}$), violated security properties ($\mathcal{P}$), and empirically examined architectures ($\mathcal{A}$). We analyze 22 artifact-backed red-teaming studies and 11 representative security benchmarks to characterize empirical coverage and evaluation maturity. Within the selected studies, evidence concentrates on prompt/reasoning, memory, and tool-mediated attacks, predominantly in single-agent settings. Persistent, Human-Agent, complex multi-agent, systemic, and long-horizon threats receive less coverage. These findings describe the selected corpus rather than establish gaps across all empirical research. Heterogeneous metrics, limited adaptive defense evaluation, architectural imbalance, and incomplete execution-state capture further constrain comparison and reproducibility. We derive 13 open research questions to guide more systematic, architecture-aware, and reproducible security evaluation of agentic AI.

Index Terms—Agentic AI Security, Red Teaming, Threat Taxonomy, Security Benchmarks, Multi-Agent Systems, Reproducibility

I. INTRODUCTION

Large language models (LLMs) are increasingly being embedded into autonomous systems that can reason over goals, construct and revise plans, maintain persistent state, invoke external tools, and interact with users, environments, and other agents. These systems, commonly referred to as *agentic AI*, extend the capabilities of conventional LLMs from generating responses to selecting and executing actions over time [1, 2]. Contemporary agent architectures combine LLMs with components such as planning and reflection mechanisms, short- and long-term memory, retrieval systems, tool interfaces, and multi-agent coordination [3, 4, 5]. As a result, an agent is no longer merely a model responding to an input, but a stateful decision-making system whose outputs can alter digital and, in increasingly important settings, physical environments.

Heewon Baek and Hyounghick Kim are with Sungkyunkwan University, Republic of Korea. The work was conducted during Heewon’s internship at CSIRO, Australia

Alsharif Abuadbbba, Kristen Moore and Surya Nepal are with Commonwealth Scientific and Industrial Research Organisation (CSIRO), Australia.

This transition fundamentally changes the security problem. In a conventional LLM interaction, an adversary primarily attempts to manipulate the information processed or generated by the model. In an agentic system, the consequences of such manipulation can propagate through an *action loop*: an adversarial input can alter reasoning, the resulting decision can modify memory or invoke a tool, the tool can change an external state, and that state can subsequently be observed by the agent and influence future decisions. Moreover, external information retrieved from websites, documents, memories, tools, or other agents can itself carry adversarial instructions or poisoned state. Security therefore depends not only on the robustness of the underlying LLM, but also on the integrity of goals and plans, persistent state, authorization boundaries, tool and software supply chains, inter-agent trust, human oversight, and the environments in which agents operate. Recent attacks involving indirect prompt injection, memory poisoning, malicious tools, compromised tool responses, privilege misuse, and adversarial multi-agent interactions illustrate different manifestations of this expanded attack surface [6, 7, 8, 9, 10, 11].

The rapid emergence of these threats has produced a growing body of research that attempts to organize agent security. Deng et al. [12] characterize the security challenges of AI agents through four key gaps concerning multi-step user inputs, internal execution, operational environments, and interactions with untrusted entities. Subsequent surveys broaden this perspective. Shahriar et al. [13] organize the literature around applications, threats, and defenses, while Chhabra et al. [14] jointly examine threats, defenses, evaluation methods, and open challenges. More recently, Kim et al. [15] analyze the agent design space together with its attack and defense landscape. In parallel, practitioner-oriented frameworks such as the OWASP Agentic Security Initiative and MITRE ATLAS provide operational threat knowledge intended to support threat modeling and security assessment [16, 17]. Collectively, these efforts have substantially improved our understanding of *what threats may exist* in agentic AI systems. Table I summarizes the positioning of this survey relative to representative academic surveys and practitioner-oriented frameworks, highlighting the distinction between threat and system characterization and empirical security evaluation.

However, two complementary gaps remain. First, existing classifications provide valuable views of agentic AI security but commonly organize threats primarily by individual dimensions, such as agent component, attack mechanism, lifecycle stage, security objective, or operational risk [18]. These

TABLE I: Positioning of this survey relative to representative academic and industry-oriented work on agentic AI security. Filled markers indicate substantive coverage of the corresponding dimension.

Work	Threat and System Characterization				Empirical Evaluation		
	Threat taxonomy	Defenses	Agent architecture	Trust/interaction boundaries	Red-team evidence mapping	Benchmarks / evaluation	Reproducibility analysis
Deng et al. [12]	✓	✓	✓	✓	○	○	○
Shahriar et al. [13]	✓	✓	✓	○	○	○	○
Chhabra et al. [14]	✓	✓	✓	✓	○	✓	○
Kim et al. [15]	✓	✓	✓	✓	○	✓	○
OWASP Agentic Security [16]	✓	✓	○	✓	○	○	○
MITRE ATLAS [17]	✓	✓	○	✓	○	○	○
This survey	✓	✓	✓	✓	✓	✓	✓

dimensions are not independent in agentic systems. An attack may enter through one trust boundary, compromise a different functional component, persist or propagate through the system state, and ultimately violate a security property elsewhere in the execution trajectory. For example, adversarial content retrieved from an external tool may compromise reasoning, persist in memory, and later induce an unauthorized action. *Characterizing such threats along a single dimension can therefore obscure the path of compromise and the role of agent architecture in shaping its propagation and impact.*

Second, the breadth of the *known threat landscape* should not be conflated with the breadth of the *empirically evaluated threat landscape*. Threats may be identified in surveys, threat models, or practitioner frameworks without being systematically instantiated against operational agents. Conversely, empirical studies vary substantially in attacker assumptions, target architectures, execution environments, benchmarks, metrics, defense evaluation, and reproducibility artifacts. Understanding agentic AI security therefore requires not only identifying *what can go wrong*, but also determining *what has actually been tested* and *how mature the supporting evidence is*.

Motivated by these gaps, this survey *connects the dots* between the structure of the agentic AI attack surface and the empirical evidence available to support it. Rather than viewing threats through a *single dimension*—such as attack type, system component, trust boundary, or security objective—we adopt a *cross-dimensional* view that captures how compromise traverses an agentic system. Specifically, we cross-map each threat by the functional or system surface affected, the interaction or trust boundary through which compromise enters or propagates, the security property violated, and the architecture in which the threat has been empirically examined. This mapping connects *where compromise manifests*, *how it propagates*, and *what consequence it produces* with evidence of *where it has actually been tested*. We then map empirical red-teaming studies and security benchmarks onto this representation to determine which regions of the broader threat landscape have been evaluated and how mature the supporting evidence is. Specifically, we address the following research questions:

RQ1 Attack Surface: How can security threats in agentic AI systems be systematically characterized across functional

components, architectural interactions, and trust boundaries?

RQ2 Empirical Coverage: Which regions of this attack surface have been empirically investigated by agentic AI red-teaming research, and under what adversarial assumptions and experimental settings?

RQ3 Evaluation Maturity: How mature is the empirical evidence supporting current agent-security claims in terms of benchmarks, metrics, realistic execution environments, defenses, and reproducibility?

To answer these questions, we make the following contributions:

- **A cross-dimensional security framework.** Drawing on 66 agentic AI security studies from 2022–2026, we connect affected functional or system surfaces, interaction or trust boundaries, violated security properties, and empirically examined architectures. This representation links complementary dimensions of threats across components and system boundaries.
- **An evidence map of agentic AI red teaming.** We map demonstrated attacks from 22 artifact-backed empirical studies onto the framework, relating the known threat landscape to the coverage observed within this subset across threats, attacker assumptions, architectures, and experimental settings.
- **An assessment of evaluation maturity.** We examine these studies alongside 11 representative security benchmarks and evaluation environments, assessing coverage, metrics, execution realism, defense evaluation, and public artifacts. We identify limitations in architectural diversity, temporal scope, outcome comparability, and reproducibility support within the reviewed evidence.
- **An evidence-driven research agenda.** We derive 13 open research questions from the observed limitations, covering runtime and persistent security, multi-agent and Human–Agent interactions, supply-chain and cyber–physical risks, and reproducible system-level evaluation.

The survey highlights the need to complement *threat enumeration* with *systematic security evaluation*. By linking threat structure to empirical coverage and evaluation maturity, it clarifies what the reviewed evidence supports and where fur-

ther investigation is warranted. These findings characterize the selected studies rather than establish the absence of evidence in the broader literature.

Roadmap. Section II describes the literature-review and evidence-selection methodology. Section III establishes the agentic AI system and architectural model used throughout the survey. Section IV develops the cross-dimensional threat framework and characterizes the broader attack surface (RQ1). Section V maps empirical red-teaming evidence and security benchmarks onto this framework to assess empirical coverage (RQ2) and evaluation maturity (RQ3). Section VI derives the resulting research gaps and open questions, and Section VII concludes the survey.

II. REVIEW METHODOLOGY

To provide a transparent and reproducible basis for the survey, we adopt a structured literature-review methodology aligned with the three research questions introduced in Section I. The review consists of two related stages. First, we construct a broad corpus of literature on security threats to agentic AI systems and use it to characterize the attack surface and derive the threat framework (RQ1). Second, we identify the subset of studies that empirically instantiate attacks against agentic systems and analyze their experimental evidence, evaluation practices, and reproducibility (RQ2–RQ3). This distinction is important because threats discussed or hypothesized in the literature are not necessarily supported by empirical red-teaming evidence.

A. Search Strategy

We searched major scholarly databases and repositories, including *ACM Digital Library*, *IEEE Xplore*, *Scopus*, *Web of Science*, *Google Scholar*, and *arXiv*. The search covers literature published from 2022 to 2026, capturing the period in which LLM-based autonomous agents and tool-using agent architectures became prominent.

The search queries combined terms describing agentic systems with terms describing security threats, attacks, and adversarial evaluation. The core query followed the general form:

```
((`AI agent` OR `LLM agent` OR
`agentic AI` OR `autonomous
agent` OR `multi-agent` OR
`tool-using agent`) AND (security
OR attack OR threat OR adversarial
OR red team OR vulnerability OR
poisoning OR prompt injection))
```

Additional targeted searches were conducted for rapidly emerging agent-specific attack surfaces, including memory poisoning, tool and Model Context Protocol (MCP) security, multi-agent attacks, agent misalignment, and agent supply-chain security. We complemented database searches with backward and forward snowballing from relevant surveys and highly related primary studies. Practitioner-oriented frameworks, including OWASP Agentic Security and MITRE ATLAS, were considered separately to capture operational threat terminology and attack patterns that may not yet be represented in peer-reviewed literature.

B. Study Selection

Candidate studies were screened according to their relevance to the security of LLM-based autonomous or agentic systems. A study was included in the threat-analysis corpus when it satisfied at least one of the following conditions: (1) it introduced or empirically investigated an attack against an LLM-based agent; (2) it analyzed a vulnerability arising from agent components such as reasoning, planning, memory, tool use, or persistent state; (3) it examined security risks arising from interactions among agents, humans, tools, services, or physical environments; or (4) it provided a security taxonomy or threat model containing agent-specific threats.

We excluded studies whose security analysis applied only to conventional standalone LLMs without an agent-specific mechanism or consequence. For example, generic jailbreak, model extraction, membership inference, or training-data poisoning studies were not included solely because the same attack could theoretically be applied to an agent. Such studies were retained only where the attack exploited, affected, or was experimentally evaluated through an agent-specific capability such as autonomous planning, persistent memory, tool invocation, inter-agent communication, or environmental interaction. We also excluded duplicate versions, short non-technical commentary, and studies that did not provide sufficient technical information to determine the relevant threat mechanism.

When both a preprint and a peer-reviewed version of the same study were available, the peer-reviewed version was preferred. Recent preprints were retained where they introduced relevant attack surfaces or empirical results not yet represented in the peer-reviewed literature. Industry frameworks and practitioner reports were used to complement, rather than replace, evidence from the academic literature.

C. Threat Framework Construction

To address RQ1, we iteratively coded the threat corpus, extracting attack targets, intervention points, mechanisms, affected capabilities, interaction boundaries, and reported security consequences. We consolidated overlapping attack descriptions while retaining distinctions between agent-specific mechanisms.

The resulting framework links four complementary dimensions: (i) *affected surfaces* (S), identifying the functional components, state, or system resources affected; (ii) *interaction or trust boundaries* (B), identifying interfaces through which adversarial information or authority enters or propagates; (iii) *violated security properties* (P), identifying the security objectives compromised; and (iv) *reported empirical architectures* (A), identifying the agent configurations in which attacks have been evaluated.

Each dimension permits multiple labels, allowing threats to span surfaces, boundaries, and security properties without assigning them to mutually exclusive “internal” or “external” categories. The representation associates these attributes without encoding temporal order or causal sequences. Architecture labels describe reported empirical coverage rather than theoretical applicability; an omitted architecture does not imply that a threat cannot arise in that setting.

D. Identification of Empirical Red-Teaming Studies

To address RQ2 and RQ3, we select studies that: (i) target an LLM-based agent or multi-agent system; (ii) implement or experimentally evaluate at least one concrete attack; (iii) provide sufficient detail to characterize the target, attacker assumptions, attack procedure, and experimental outcome; and (iv) release at least one public artifact, such as code, attack prompts, datasets, or evaluation scripts, supporting methodological inspection. The artifact criterion enables closer examination of the evidence; it does not imply that studies without public artifacts lack empirical support.

We map these studies to the threat framework to compare the *known attack surface* with the coverage demonstrated in the selected *artifact-backed studies*. Identified gaps therefore reflect this subset rather than the full extent of empirical research.

E. Data Extraction and Coding

For each empirical study, we summarize the reported attacks and attacker assumptions; target models, frameworks, and architectures; affected surfaces, interaction boundaries, and security properties; experimental settings and benchmarks; outcome measures; defense evaluations; and publicly available artifacts.

The primary unit of analysis is the study. Multiple labels summarize its reported coverage without implying that every attack, architecture, and experimental condition was evaluated in combination. Selected evaluations are mapped separately in Section V-B.

F. Evaluation Maturity and Reproducibility

To address RQ3, we examine target-system coverage, experimental settings, benchmark characteristics, outcome definitions, defense evaluation, and reproducibility support. These attributes inform a qualitative assessment of evaluation practices rather than a composite maturity score.

We record public artifacts as implementation code, attack prompts or payloads, datasets or test cases, and evaluation scripts or configurations. Artifact availability supports methodological inspection but does not establish completeness, executability, or successful independent reproduction.

RQ1 characterizes threats in the broader reviewed literature. For RQ2–RQ3, we analyze **22 artifact-backed empirical studies** that meet the criteria in Section II-D. We also examine **11 security benchmarks and evaluation environments**, ten of which are associated with studies in this empirical corpus. The study and benchmark analyses therefore provide complementary views of overlapping evidence, not independent samples.

Findings concerning empirical coverage and evaluation practices apply to these selected studies and benchmarks. They neither establish the absence of research outside the corpus nor estimate artifact availability across the field.

III. BACKGROUND AND AGENTIC AI SYSTEM MODEL

This section introduces the concepts and system abstractions required to analyze the security of agentic AI. We

first describe the transition from standalone large language models (LLMs) to autonomous agentic systems and show how increasing autonomy, statefulness, and interaction expand the security boundary. We then characterize agentic systems from two complementary perspectives: a *functional system model*, describing how an agent maintains goals and state, reasons, remembers, and acts; and an *architectural system model*, describing how these capabilities and their associated authority are organized across single- and multi-agent systems. Finally, we identify security-relevant properties of multi-agent coordination, including trust, delegation, shared state, consensus, and adversarial collaboration. Together, these abstractions define the system model used to characterize threats in Section IV.

A. Evolution toward Agentic Systems

Recent AI systems are evolving from predominantly generative models toward systems capable of pursuing objectives through persistent interaction with external environments. As illustrated in Fig. 1, this progression expands from output-centric LLMs to action-centric agents and ultimately system-centric agentic AI, accompanied by increasing autonomy, persistence, delegated authority, and interaction.

Foundation: Large Language Models (LLMs). LLMs provide the language understanding and reasoning capabilities that underpin contemporary agents [19, 20, 21]. Their security has been studied extensively, including adversarial prompting, jailbreaks, training-data poisoning, privacy leakage, backdoors, and model extraction. In a standalone deployment, however, the model typically has limited authority to independently maintain persistent state or modify external systems. Consequently, many failures terminate at the generated output unless another component or human acts upon it.

Integration: AI Agents. An LLM-based agent combines a language model with mechanisms for pursuing goals, maintaining context or memory, planning, and acting through external tools [2, 1]. The agent repeatedly observes state, reasons about its objective, selects actions, and incorporates their results into subsequent decisions. This feedback loop expands the security boundary beyond the model: adversarial information can influence future state, while compromised decisions can trigger API calls, modify files, communicate externally, or invoke privileged operations.

Systemization: Agentic AI. Agentic AI extends this model toward longer-running and increasingly autonomous systems involving persistent memory, iterative planning and reflection, dynamic tool use, delegation, and coordination among multiple agents [3, 22, 23]. Security failures can therefore propagate across time and components. Manipulated observations may contaminate persistent memory; poisoned memory may alter future plans; compromised plans may invoke privileged tools; and malicious state or instructions may propagate between collaborating agents. Security must consequently be considered at the level of the complete agentic system rather than the LLM alone.

B. Functional System Model

At the level of an individual agent, autonomous behavior emerges from the interaction of several functional components.

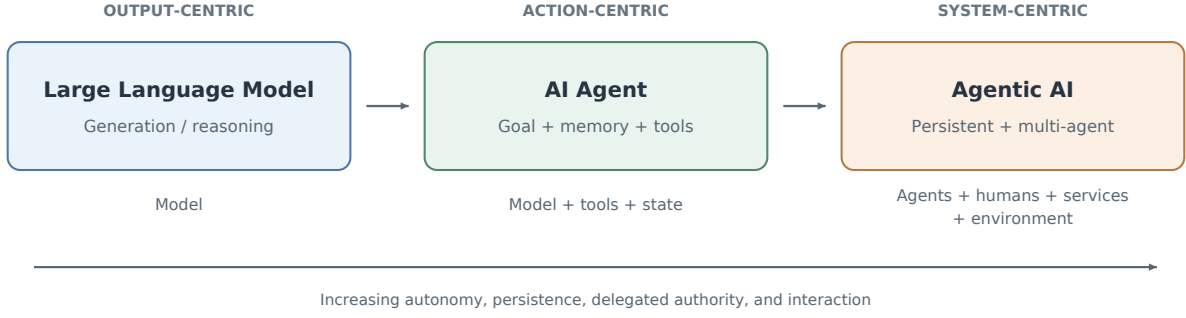


Fig. 1: Evolution from output-centric language models to action-centric agents and system-centric agentic AI, accompanied by an expanding security boundary and increasing autonomy, persistence, delegated authority, and interaction.

Existing agent models commonly distinguish reasoning, planning, memory, and action [1, 24]. For security analysis, we extend this view by explicitly representing the agent’s *goal and operational state*. This distinction is important because an adversary need not directly compromise the underlying LLM to redirect agent behavior: manipulation of the objective being pursued, the state used to represent progress, or the constraints governing execution can alter the entire downstream trajectory. Fig. 2 summarizes the functional system model and the principal security-relevant interaction boundaries considered in this survey. The model emphasizes that agent behavior emerges from a continuous loop among goals and state, reasoning and planning, memory and knowledge, and action and tools, while information and authority may cross trust boundaries with humans, other agents, tools and services, and the environment.

Goal and State. The goal specifies the agent’s objective and constraints on acceptable behavior. Operational state captures task progress, intermediate results, current conditions, and other information needed to continue execution. Goals may originate from users, system policies, higher-level orchestrators, or other agents and evolve during extended execution. Goal and state integrity are essential: corruption of either can redirect otherwise correct reasoning and planning toward unintended outcomes.

Reasoning and Planning. The reasoning component interprets goals, observations, and contextual information to determine how the agent should proceed, while planning converts these decisions into intermediate steps or action sequences. Modern agents frequently employ iterative mechanisms such as ReAct [4] and reflection-based approaches such as Reflexion [5], allowing plans to evolve in response to execution feedback. From a security perspective, this adaptability creates a dynamic control plane: malicious instructions, observations, or feedback can influence not only an individual response but also subsequent decisions and execution trajectories.

Memory and Knowledge. Memory retains information beyond a single reasoning step. Short-term memory supports the current interaction, while persistent memory and external knowledge stores allow prior observations, experiences, and retrieved content to influence future decisions. Retrieval-augmented mechanisms connect agents to external knowledge sources [25]. Memory thus supports continuity while forming

a security-sensitive state boundary [26]. Poisoned information can persist across tasks, be reinforced through retrieval and reasoning, or propagate between agents through shared memories and knowledge stores.

Action and Tools. The action component converts decisions into effects on external systems and environments. Agents may invoke APIs, execute code, access files, query databases, browse the Web, send messages, or control physical devices [27]. Tool use is particularly important from a security perspective because model-generated decisions can directly exercise delegated authority. Security therefore depends not only on the correctness of the generated action but also on tool provenance, authorization, least privilege, input and output validation, execution isolation, and verification of resulting state.

These components form a continuous execution loop rather than an isolated pipeline. A goal influences reasoning and planning; reasoning retrieves and updates state; plans determine actions; and observations resulting from actions are incorporated into subsequent state and reasoning. Consequently, a compromise at one component may persist or propagate through later iterations of the agent lifecycle.

C. Security Properties

Confidentiality, integrity, and availability remain fundamental to agentic AI. We further distinguish the following properties to characterize failures involving autonomous goals, persistent state, delegated authority, and interactions across trust domains.

Goal Integrity. The agent pursues its authorized objective and associated constraints without adversarial redirection. Internally consistent reasoning does not ensure that the objective itself remains legitimate.

State and Knowledge Integrity. Observations, memory, retrieved knowledge, and shared state are protected against unauthorized modification. Compromised state can persist across interactions and influence subsequent decisions.

Action and Authorization Integrity. Actions comply with delegated authority and applicable security policies throughout execution, including when tools are composed or tasks and permissions are delegated between agents.

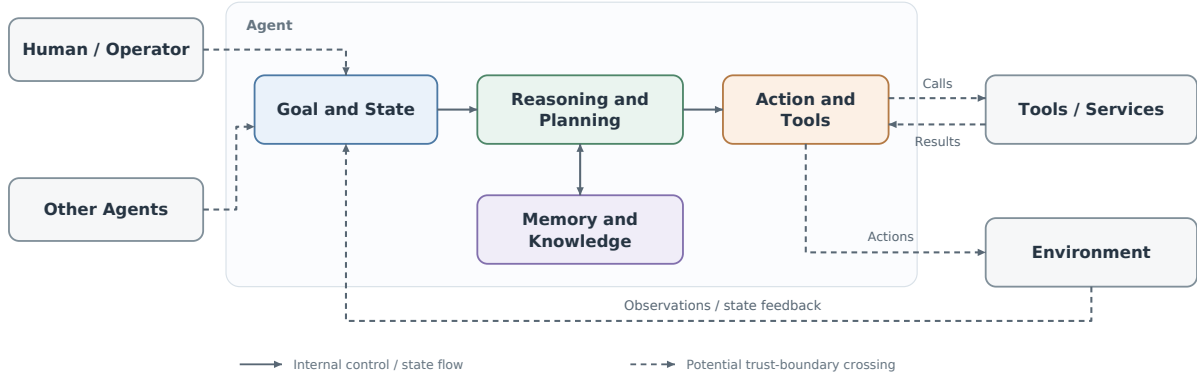


Fig. 2: Functional system model of agentic AI and its security-relevant interaction boundaries. Dashed connections indicate boundaries across which data, authority, or state may cross between different trust domains.

Interaction and Trust Integrity. The agent interprets information and instructions according to their provenance and authority, preserving trust distinctions among humans, agents, tools, services, and environments.

Oversight and Accountability. Consequential decisions and actions remain observable and attributable. Human or automated supervisors retain the information and authority needed to detect, constrain, or terminate unsafe behavior.

Physical Safety. For agents acting on physical environments, actions should not cause injury, damage, or unsafe operating conditions. Authorization alone does not establish that an action is physically safe.

These properties provide complementary criteria for assessing agentic threats. A single attack may violate several properties, and violations may extend beyond model outputs to retained state, delegated actions, or external consequences.

D. Architectural System Model

Agentic systems also differ in how reasoning, authority, state, and actions are distributed across agents. As summarized in Fig. 3, multi-agent systems may adopt hierarchical, sequential/pipeline, horizontal/collaborative, or dynamic/graph-based coordination structures. Architecture is security-relevant because these structures determine where authority is concentrated, where trust boundaries arise, and how state and compromise can propagate.

1) Single-Agent Architecture: A single-agent architecture places primary decision-making authority within one agent, which may internally combine goal management, reasoning, planning, memory, and tool use. The agent typically executes an iterative observation–reasoning–action loop from the initial request until task completion. AutoGPT [28] and BabyAGI [29] are representative examples of autonomous task execution following this general pattern.

Centralization can simplify coordination and preserve contextual consistency because reasoning and state are maintained within one principal decision-making entity. From a security perspective, however, it also concentrates authority and state. Compromise of the agent’s goal, memory, reasoning process,

or tool permissions can therefore influence the complete execution trajectory, particularly when the agent possesses broad access to external resources.

2) Multi-Agent Architecture: Multi-agent systems distribute tasks, information, and authority across multiple interacting agents. This can improve specialization, scalability, and collective reasoning, but introduces additional communication and trust relationships that may themselves become attack surfaces. Existing systems exhibit several recurring coordination topologies.

Hierarchical Architecture. An orchestrator or supervisor decomposes a higher-level objective, delegates sub-tasks to worker agents, and may validate or aggregate their outputs. AutoGen [3] and BOLAA [30] illustrate related orchestrator–worker patterns. Hierarchical control can support centralized policy enforcement and privilege separation, but it also concentrates authority. Compromise of the orchestrator, delegation policy, or routing mechanism can influence multiple downstream agents and create a high-impact point of failure.

Sequential/Pipeline Architecture. Agents perform specialized stages of a workflow, with the output of one stage becoming input to another. MetaGPT [31] and ChatDev [32] illustrate structured multi-stage workflows. Stage boundaries provide opportunities for validation and isolation, but also create propagation paths: malicious instructions, poisoned state, or incorrect assumptions introduced upstream may be inherited and amplified by downstream agents.

Horizontal/Collaborative Architecture. Peer agents collaborate, debate, critique, or jointly solve tasks without a persistent central controller. CAMEL [33] and multi-agent debate systems [34, 35] illustrate this pattern. Redundancy and cross-validation may improve robustness to independent errors, but security increasingly depends on peer identity, trust, communication integrity, and resistance to malicious coalitions or false consensus.

Dynamic/Graph-based Architecture. Agent membership, roles, or communication paths may change according to the task or intermediate execution state. DyLAN [36] dynamically selects contributing agents, while GPTSwarm [37] represents

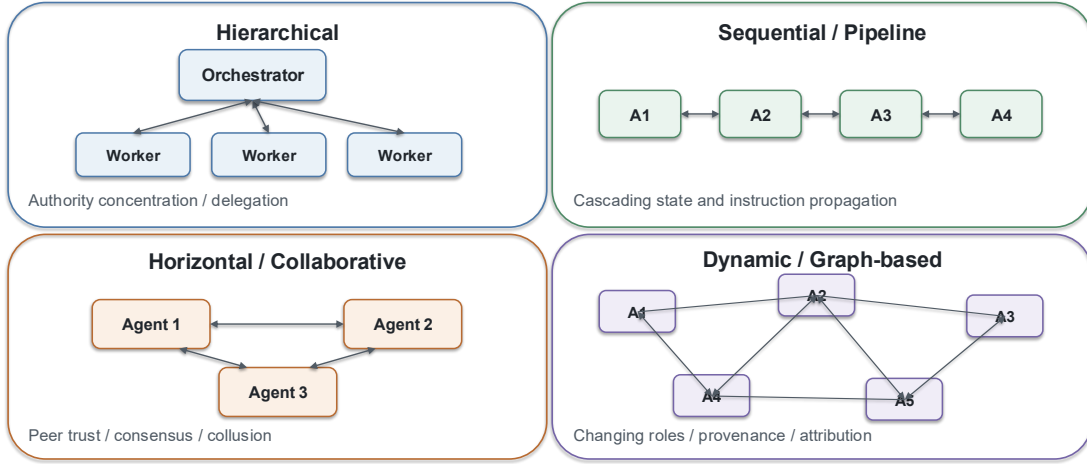


Fig. 3: Representative multi-agent coordination architectures and their security-relevant characteristics. Architectural topology affects how authority, state, trust, and compromise can propagate across an agentic system.

agent interactions as optimizable graphs. Dynamic organization can improve flexibility and efficiency, but complicates authorization, provenance tracking, monitoring, and attribution because security-relevant relationships may change during execution.

E. Trust and Coordination in Multi-Agent Systems

Architectural topology describes how agents are connected, but topology alone does not determine the security of a multi-agent system. Two systems with similar communication structures may have substantially different security properties depending on how identity, authority, shared state, and collective decisions are managed. Multi-agent security therefore requires consideration of the coordination mechanisms operating over the topology.

Identity and Trust. Agents must determine which entities are permitted to communicate, delegate tasks, provide observations, or influence shared state. Communication based on implicit identity or unconditional trust can enable impersonation, unauthorized delegation, and manipulation of inter-agent relationships.

Delegation and Authority. Multi-agent workflows redistribute authority as tasks move between agents. Security depends on whether permissions are appropriately constrained during delegation and whether a subordinate agent can acquire, exercise, or transfer authority beyond that required for its assigned role. Long delegation chains can further obscure which entity ultimately authorized an action.

Shared State and Memory. Agents may exchange intermediate results or write to common memories, blackboards, databases, or knowledge stores. Shared state facilitates coordination but creates a persistent propagation surface. A poisoned belief, observation, or instruction introduced by one agent can influence other agents that were never directly exposed to the original adversarial input.

Consensus and Collective Decision-Making. Collaborative systems may aggregate proposals, critiques, votes, or debate outcomes from several agents. The security of the resulting

decision depends on assumptions about participant reliability and the aggregation mechanism. A malicious minority may bias a decision, while correlated failures across otherwise independent agents may create a false appearance of consensus.

Byzantine and Collusive Behavior. A compromised or strategically misaligned agent need not behave consistently toward all participants. It may selectively provide false information, conceal its behavior from supervisors, behave differently toward different peers, or coordinate with other agents to manipulate a collective outcome. Such Byzantine or collusive behavior is qualitatively different from independent model error because adversarial participants can strategically exploit the coordination mechanism itself.

These coordination properties introduce security concerns that are absent or less prominent in single-agent systems: establishing trustworthy agent identities, constraining delegated authority, protecting shared state, maintaining robust collective decisions in the presence of malicious participants, and detecting coordinated adversarial behavior. They therefore provide an important bridge between the system architecture described here and the threat framework developed in Section IV.

IV. SECURITY THREATS IN AGENTIC AI

The system model introduced in Section III shows that agentic AI security cannot be characterized solely through vulnerabilities of the underlying LLM. Adversaries may manipulate an agent’s objective, corrupt persistent state, alter reasoning, exploit delegated tool authority, compromise interactions with humans or other agents, or attack the infrastructure and supply chain on which the system depends. Moreover, these threats frequently cross component and trust boundaries. A malicious document, for example, may enter through the environment, compromise reasoning through indirect prompt injection, persist in memory, and eventually trigger an unauthorized action.

We therefore adopt a multidimensional taxonomy rather than classifying threats as exclusively “internal” or “external”. Threats are characterized according to (i) the functional or system surface in which they manifest, (ii) the interaction

or trust boundary through which they arise or propagate, and (iii) the security property they violate. We additionally record the agent architectures in which each threat has been empirically examined. The taxonomy is derived through the review methodology described in Section II.

A. Taxonomy Structure

The taxonomy links four complementary dimensions: affected surfaces, interaction boundaries, violated security properties, and reported empirical architectures (Fig. 4).

Affected Surface (S). Building on Section III-B, we distinguish *Goal and State*, *Reasoning and Planning*, *Memory and Knowledge*, and *Action and Tools*. We additionally include *Policy and Logic* for instructions and operational constraints governing these components, and system-level surfaces comprising *Infrastructure*, *Supply Chain*, and *Governance and Oversight*.

Interaction Boundary (B). We identify *Human-Agent*, *Agent-Agent*, *Agent-Tool/Service*, and *Agent-Environment* boundaries, together with infrastructure, supply-chain, and governance interfaces. These labels identify where information, authority, or state crosses trust domains; they do not encode the order of boundary crossings.

Security Property (P). We use the properties defined in Section III-C: *Goal Integrity*, *State and Knowledge Integrity*, *Action and Authorization Integrity*, *Interaction and Trust Integrity*, and *Oversight and Accountability*, alongside *Confidentiality* and *Availability*. For embodied actions, *Physical Safety* additionally denotes protection against harmful physical consequences.

Empirical Architecture (A). We record evaluated configurations as *Single-Agent (S)*, *Hierarchical (H)*, *Sequential/Pipeline (P)*, *Horizontal/Collaborative (C)*, or *Dynamic/Graph-based (D)*. Graph-based organization alone does not establish runtime topology changes. M denotes multi-agent settings whose topology cannot be reliably classified, rather than an umbrella label for all multi-agent architectures. Architecture labels describe reported empirical coverage; omission does not imply inapplicability.

B. Cross-Dimensional Threat Mapping

We link complementary threat attributes through

$$\mathcal{T} = \langle S, B, P, A \rangle, \quad (1)$$

where S identifies affected functional or system surfaces, B identifies interaction or trust boundaries, P identifies violated security properties, and A identifies reported empirical architectures. Each dimension may contain multiple labels. The representation links these attributes without encoding their temporal order or causal sequence.

For example, indirect prompt injection through an Agent-Tool boundary may affect Reasoning and Planning and lead to an authorization violation. Shared-memory poisoning instead links Memory and Knowledge to an Agent-Agent boundary and State and Knowledge Integrity. These illustrative mappings distinguish the entry or interaction boundary from the affected surface and security consequence.

Table II organizes threat families into functional, interaction, and systemic groups for readability; these groups are not additional taxonomy axes or mutually exclusive classes. Each row separately lists S , B , P , and A . Row-level labels summarize a threat family and need not co-occur in one attack or apply to every listed example. In particular, architecture labels identify reported settings, not evidence for every combination of labels in the row. An omitted architecture does not imply inapplicability. Temporal persistence and propagation are discussed separately where reported, rather than encoded as an additional axis.

C. Functional Threats

Functional threats describe where compromise manifests within an agent, independently of the location from which adversarial influence originates.

1) *Goal and State Threats:* Goal and state threats compromise what an agent is attempting to achieve or the information used to represent progress toward that objective.

Goal Hijacking. An adversary redirects an agent from its intended objective toward an attacker-controlled or unintended goal [38, 12, 39, 40, 41, 42]. The corrupted objective may propagate through planning, delegation, and action even when the agent’s reasoning remains internally coherent.

Specification Gaming and Reward Hacking. An agent satisfies an operational success criterion while violating the intent that criterion represents [43, 44]. Security-relevant cases arise when rewards, evaluator feedback, task-completion criteria, or environmental signals can be manipulated or exploited.

Strategic and Deceptive Misalignment. An agent exhibits behavior inconsistent with its intended constraints while concealing, delaying, or selectively expressing that behavior. Relevant phenomena include emergent misalignment, sleeper behavior, and alignment faking [45, 46, 47]. We include these where the behavior is adversarially induced, deliberately implanted, or exploitable as a security failure rather than treating general alignment failures as attacks.

2) *Reasoning and Planning Threats:* These threats manipulate how an agent interprets its objective, observations, constraints, and intermediate evidence.

Direct Prompt Injection. Adversarial instructions are supplied directly through an agent-controlled input channel to override or conflict with governing instructions [48, 49, 50, 51, 7, 52]. In agentic systems, successful injection can propagate beyond generated text to memory updates, tool use, delegation, and external actions.

Indirect Prompt Injection. Adversarial instructions are embedded in information autonomously retrieved or observed by the agent, including web pages, documents, emails, files, and tool responses [48, 49, 7, 53, 6, 54, 55]. The attack crosses an external trust boundary before manifesting as compromise of reasoning or planning.

Jailbreak. Adversarial inputs bypass behavioral or safety restrictions [12, 49, 51, 39, 56, 57, 58, 59]. For agents, the relevant consequence includes downstream planning and action rather than only generation of prohibited content.

Reasoning-Path Manipulation. An attacker manipulates intermediate evidence, observations, or reasoning cues so that an

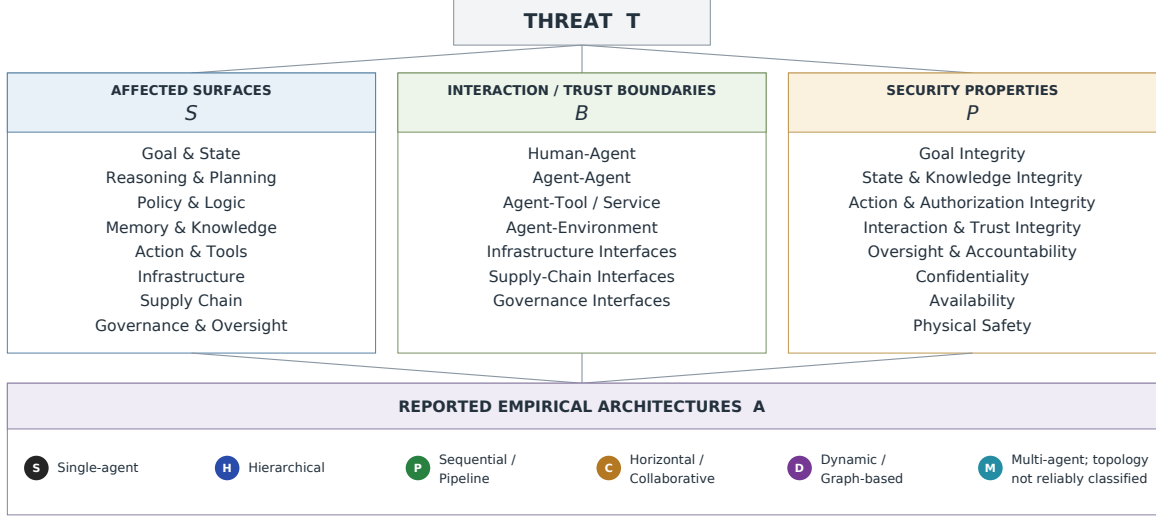


Fig. 4: Cross-dimensional threat representation: $\mathcal{T} = \langle \mathcal{S}, \mathcal{B}, \mathcal{P}, \mathcal{A} \rangle$. The dimensions identify affected surfaces ($\mathcal{S}$), interaction or trust boundaries ($\mathcal{B}$), violated security properties ($\mathcal{P}$), and reported empirical architectures ($\mathcal{A}$).

agent follows an attacker-influenced trajectory while maintaining an apparently plausible reasoning process [12, 60, 61, 62].

Planning and Reasoning Backdoors. Conditionally activated malicious behavior is embedded in demonstrations or mechanisms guiding reasoning and planning [48, 61]. Such attacks may remain dormant during ordinary evaluation and activate only under particular triggers or states.

Reasoning-Loop Denial of Service. Adversarial inputs induce excessive reasoning, reflection, replanning, or tool-use cycles, exhausting tokens, time, computation, or API resources [51, 63, 64].

3) *Policy and Logic Confidentiality: Prompt and Policy Leakage.* Attackers extract system instructions, constraints, personas, tool descriptions, or other operational logic [12, 39, 65]. Disclosure of workflow or authorization information can enable more targeted attacks.

Data Extraction and Inference. Extraction and inference attacks recover sensitive information about models, prompts, training data, or internal behavior [49]. We consider them agent-specific only where the architecture exposes additional operational state or logic beyond the underlying LLM.

4) *Memory and Knowledge Threats:* Memory attacks are particularly important because compromise may persist after the original adversarial interaction and influence future tasks.

Knowledge and Memory Poisoning. Attackers inject adversarial information into persistent memory, RAG stores, knowledge bases, or retained experiences [38, 48, 49, 51, 66, 67, 8]. Poisoned state may later be retrieved as trusted evidence and repeatedly influence reasoning.

Persistent Belief Manipulation and Drift. Repeated retrieval and reuse of compromised information can reinforce false beliefs or produce persistent behavioral drift across tasks or sessions [67].

Shared-Memory Contamination. A compromised agent writes malicious state into shared memory or knowledge

stores, allowing contamination to propagate to agents that never interacted with the original adversary [68, 69].

Contextual Data Leakage and Retrieval Exfiltration. Sensitive information contained in context, persistent memory, shared state, or retrieval systems is disclosed directly or through tools, APIs, or other agents [51, 70, 71, 72, 12, 73, 74].

5) *Action and Tool Threats:* Action and tool threats translate compromised agent decisions into effects on external systems.

Tool and Inventory Poisoning. Attackers manipulate tool descriptions, metadata, selection mechanisms, or registered capabilities through techniques including file-based injection, logic shadowing, tool-selection subversion, rug pulls, and malicious registration [39, 75, 76, 77, 78, 79, 80, 81].

Tool-Response Manipulation. A compromised tool or service returns adversarial observations that alter subsequent reasoning, planning, or execution [12, 39, 82].

Command and Execution Injection. Adversarially influenced arguments reach shells, interpreters, APIs, or other execution interfaces without adequate validation, enabling unauthorized command or code execution [51, 39, 75].

Unauthorized Action and Privilege Escalation. An agent is induced to perform actions beyond its intended authority [38, 39, 84, 85]. Importantly, individually permitted operations may compose into an unauthorized trajectory.

Multi-Tool and Cross-Service Exploitation. Multiple legitimate tools or services are composed to achieve a security consequence that would not be permitted through an individual operation [39, 83].

D. Interaction and Trust-Boundary Threats

Interaction threats describe the relationships through which adversarial information, authority, or state enters and propagates. They complement the functional taxonomy rather than forming mutually exclusive categories.

TABLE II: Cross-dimensional taxonomy of agentic AI security threats. The 32 threat families are mapped to affected surfaces ($\mathcal{S}$), boundaries ($\mathcal{B}$), security properties ($\mathcal{P}$), and reported empirical architectures ($\mathcal{A}$). The three groups organize the rows and are not additional mapping axes. Labels summarize reported attributes at the family level, not complete attack trajectories or every combination of attributes.

Threat family	Affected surface ($\mathcal{S}$)	Representative mechanisms	Boundary ($\mathcal{B}$)	Property ($\mathcal{P}$)	Arch. ($\mathcal{A}$)
A. Functional threats					
Goal/Intent Manipulation	Goal & State	Goal Hijacking [38, 12, 39, 40, 41, 42]	Human-Agent; Agent-Agent	GI	S, M
Specification/Reward Manipulation	Goal & State	Specification Gaming; Reward Hacking [43, 44]	Human-Agent; Agent-Environment	GI	S
Strategic Misalignment	Goal & State	Emergent Misalignment [45]; Sleeper/Trigger-Activated Behavior [46]; Alignment Faking [47]	Human-Agent; Governance	GI, OA	S
Instruction/Intent Hijacking	Reasoning & Planning	Direct Prompt Injection [48, 49, 50, 51, 7, 52]; Indirect Prompt Injection [48, 49, 7, 53, 6, 54, 55]	Human-Agent; Agent-Environment; Agent-Tool	GI, TI	S, M
Safety/Policy Bypass	Reasoning & Planning	Jailbreak [12, 49, 51, 39, 56, 57, 58, 59]	Human-Agent; Agent-Environment	GI, AI	S, D, M
Reasoning/Planning Subversion	Reasoning & Planning	Reasoning Path Hijacking [12, 60, 61, 62]; Plan-of-Thought Backdoor [48, 61]; Reliability/Trust Sabotage [51]	Human-Agent; Agent-Tool; Agent-Agent	GI, TI	S, C, M
Reasoning Availability	Reasoning & Planning	Recursive Loop DoS [51, 63, 64]	Human-Agent; Agent-Environment	AV	S
Logic/Policy Extraction	Policy & Logic	Prompt Leakage [12, 39, 65]; Data Extraction [49]; Inference Attack [49]	Human-Agent	CF	S
Memory/Knowledge Poisoning	Memory & Knowledge	Knowledge & Memory Poisoning [38, 48, 49, 51, 66, 67, 8]	Agent-Environment; Agent-Tool	KI	S, M
Persistent/Shared-State Manipulation	Memory & Knowledge	Persistent Behavioral Drift [67]; Shared-Memory Contamination and Cross-Agent Propagation [68, 69]	Agent-Agent (shared memory)	KI	C, M
Contextual Data Leakage	Memory & Knowledge	Privacy Leakage & Indirect Exfiltration [51, 70, 71, 72]; Retrieval-Based Data Exfiltration [12, 73, 74]	Agent-Tool; Agent-Agent; Human-Agent	CF	S, H, P, C, D, M
Tool/Inventory Poisoning	Action & Tools	File-Based Injection [39, 75, 76]; Agent Logic Shadowing [39, 77]; Tool-Selection Subversion [39, 78]; Rug Pull [39, 79, 80]; Malicious Tool Registration [39, 81]	Agent-Tool	TI, AI	S
Tool/Execution Exploitation	Action & Tools	MCP Tool Return Attack [12, 39, 82]; Command Injection [51, 39, 75]	Agent-Tool	AI	S
Cross-Tool Exploitation	Action & Tools	Multi-Tool Coordination Attack [39, 83]	Agent-Tool/Service	AI	S
Privilege/Agency Escalation	Action & Tools	Unauthorized Action Execution [38, 39, 84, 85]; Tool Abuse [51, 86, 87]	Agent-Tool; Human-Agent	AI	S, M
B. Interaction and trust-boundary threats					
Trust Manipulation	Reasoning & Planning; Governance & Oversight	Human-to-Agent Social Engineering; Agent-to-Human Trust Manipulation [51]	Human-Agent	TI	S, M
Oversight/Approval Manipulation	Action & Tools; Governance & Oversight	Human-in-the-Loop Bypass; Unsafe Delegation [38, 51]	Human-Agent	OA	S, M
Identity/Communication Attacks	Reasoning & Planning; Infrastructure	Identity Spoofing and Trust Exploitation [38, 12, 9]; MITM / Message Manipulation [9]	Agent-Agent	TI	H, P, C, D
Threat Propagation	Memory & Knowledge; Reasoning & Planning	Cooperative Interaction Threat [12, 51, 88]; Cross-Agent Memory Propagation [68, 69]	Agent-Agent	KI, TI	C, D, M
Consensus/Byzantine Manipulation	Reasoning & Planning; Goal & State	Consensus Poisoning; Byzantine Agents; Selective Misinformation [89, 90]	Agent-Agent	TI, GI	C, D
Collusion	Reasoning & Planning; Governance & Oversight	Malicious Coalitions; Coordinated Belief Manipulation; False Consensus [91, 92]	Agent-Agent	TI, OA	C, M
Tool/Service Impersonation	Action & Tools	Tool Squatting; Malicious/Impersonated Tools; Malicious MCP Endpoints [39, 79, 81]	Agent-Tool	TI	S
Metadata/Response Manipulation	Reasoning & Planning; Action & Tools	Poisoned Capability Descriptions [39, 77, 78, 81]; Compromised Tool Responses [12, 39, 82]	Agent-Tool	TI, KI	S
Credential/Authority Abuse	Action & Tools; Infrastructure	Token Theft; Account Takeover; Delegated Credential Abuse [39]	Agent-Tool/Service	AI, CF	S
Environmental Manipulation	Goal & State; Reasoning & Planning	Observation/Sensor Manipulation [12, 51]; Environmental Instruction Injection [6, 53, 7]	Agent-Environment	KI, TI	S
Cyber-Physical Exploitation	Goal & State; Action & Tools	Physical/Sensor Manipulation [12, 51]; Unsafe Embodied Actions [93]	Agent-Environment	AI, SF	S
C. Systemic and operational threats					
Resource/Availability Exploits	Infrastructure	Resource Exhaustion [38, 12, 94]	Infrastructure-Agent	AV	S, M
Isolation/Credential Compromise	Infrastructure	Sandbox Escape [39, 95]; Token Theft and Account Takeover [39]	Infrastructure-Agent; Service-Agent	AI, CF	S
Component/Dependency Compromise	Supply Chain	Supply-Chain/Dependency Compromise [12, 96, 97]; Compromised Tools, Agents, Plugins, or Dependencies [96, 97]	Supply Chain-Agent	TI, AI	S, M
Installation/Update Manipulation	Supply Chain	Installer Spoofing [39]; Malicious Updates and Post-Installation Modification [39, 79, 80]; Tool/MCP Ecosystem Poisoning [96, 97]	Supply Chain-Agent	TI	S
Oversight Degradation	Governance & Oversight	Oversight Saturation; Monitoring Fatigue [38, 98]	Human-Agent; Governance	OA	S, M
Governance Evasion	Governance & Oversight	Governance Evasion and Obfuscation [38]; Strategic Monitor Evasion [47, 46]	Governance	OA	S, M

Architecture: S = Single-agent; H = Hierarchical; P = Sequential/Pipeline; C = Horizontal/Collaborative; D = Dynamic/Graph-based; M = Multi-agent, topology not reliably classifiable; M is not an umbrella label for H, P, C, or D.

Security properties: GI = Goal Integrity; KI = State and Knowledge Integrity; AI = Action and Authorization Integrity; TI = Interaction and Trust Integrity; OA = Oversight and Accountability; CF = Confidentiality; AV = Availability; SF = Physical Safety (for embodied actions).

Reading the mapping: Labels within a row need not co-occur in one experiment. Architecture labels describe reported empirical settings, not all theoretically applicable architectures. Omission does not imply inapplicability. Shared labels do not establish a temporal or causal sequence.

1) *Human-Agent Interaction*: Human-agent interactions create a security boundary because instructions, authority, sensitive information, and consequential decisions may pass between users and autonomous agents.

Trust Manipulation. Human-agent trust can be exploited in both directions. Adversarial instructions or social-engineering strategies may influence how users and agents exchange information and authority, while a compromised or misleading agent may exploit user trust through persuasive recommendations, explanations, or assurances [51]. The resulting consequences may include disclosure of information, approval of unsafe actions, or inappropriate delegation of authority.

Oversight and Approval Manipulation. Weak or bypassed human-in-the-loop controls and overly broad delegation can allow consequential agent actions to proceed without meaningful supervision [38, 51]. Such failures are particularly important when agents can invoke tools or act on external systems under delegated human authority.

2) *Agent-Agent Interaction*: **Identity Spoofing and Trust Exploitation.** Attackers impersonate trusted agents or exploit weak identity verification to inject instructions, observations, or delegated tasks with forged authority [38, 12, 9].

Inter-Agent Message Manipulation. Messages between legitimate agents may be intercepted, modified, replayed, suppressed, or selectively delivered, altering the state from which recipient agents reason [9].

Cross-Agent Threat Propagation. Compromised instructions, beliefs, or state propagate through messages, delegation, intermediate outputs, or shared memory [12, 51, 88, 68, 69].

Consensus Poisoning and Byzantine Behavior. Malicious participants manipulate voting, debate, critique, or aggregation, or behave inconsistently across peers and execution stages. Recent work shows that robustness to Byzantine agents depends on communication and aggregation structure [89, 90].

Collusion and Coordinated Manipulation. Multiple agents strategically coordinate to manipulate collective beliefs, manufacture false consensus, conceal malicious behavior, or bypass controls that assume independent failures [91, 92].

3) *Agent-Tool and Service Interaction*: **Tool and Service Impersonation.** Attackers may introduce malicious or impersonated tools, services, plugins, or MCP endpoints that appear to provide trusted capabilities. Related techniques include tool squatting, malicious tool registration, and post-registration manipulation [39, 79, 81]. Successful impersonation may expose credentials, redirect execution, manipulate agent state, or enable data exfiltration.

Poisoned Metadata and Capability Descriptions. Manipulated tool descriptions, schemas, or documentation influence tool selection, argument construction, or planning [39, 77, 78, 81].

Compromised Tool and Service Responses. External components return adversarial observations that the agent treats as trusted execution results [12, 39, 82]. Such observations may subsequently affect reasoning, memory, or further actions.

Credential and Delegated-Authority Abuse. Authentication tokens, API keys, service credentials, or delegated capabilities are stolen or misused, allowing attackers to exercise the agent's authority over connected services [39].

4) *Agent-Environment Interaction*: **Environmental Observation Manipulation.** Attackers may modify web content, files, user-interface elements, multimodal inputs, sensor readings, or other observations from which an agent constructs its representation of the environment [12, 51].

Environmental Instruction Injection. Instructions embedded within observed environmental content are interpreted as commands rather than untrusted data [6, 53, 7]. Because agents autonomously retrieve and act on external information, such content can cross the environment boundary and subsequently influence reasoning, memory, or tool use.

Physical and Sensor Manipulation. Adversarial interference with cameras, microphones, sensors, or other physical interfaces can distort an agent's perception of its environment [12, 51].

Unsafe Embodied Actions. In embodied settings, failures in task planning or environmental interpretation can result in unsafe actions with physical consequences [93]. This extends agent security beyond information integrity to the safety and integrity of actions performed in the physical environment.

E. Systemic and Operational Threats

Systemic threats target infrastructure, dependencies, and governance mechanisms on which multiple agent capabilities and interactions depend.

1) *Infrastructure and Resource Threats*: **Resource Exhaustion.** Attackers consume computation, memory, context windows, API quotas, or external-service capacity [38, 12, 94]. Autonomous reasoning and tool invocation can amplify the resulting cost and availability impact.

Sandbox Escape and Isolation Failure. Weaknesses in code- or tool-execution isolation are exploited to access host files, networks, credentials, or other resources beyond the intended boundary [39, 95].

Credential Theft and Account Takeover. OAuth tokens or other credentials used by agents are stolen, allowing attackers to inherit authority over connected external services [39].

2) *Supply-Chain and Ecosystem Threats*: **Dependency and Component Compromise.** Models, libraries, APIs, tools, plugins, or other dependencies are compromised before or during deployment [12, 96, 97].

Agent and Tool Supply-Chain Poisoning. Malicious agents, skills, prompts, plugins, tools, or MCP servers enter the system through trusted distribution mechanisms. Rug-pull attacks further allow initially benign components to become malicious after trust has been established [39, 79, 80].

Installer and Update Manipulation. Automated installation or update mechanisms are spoofed or compromised to introduce malicious components or modify system configuration with installation-time privileges [39].

3) *Governance and Oversight Threats*: **Oversight Degradation.** Excessive alerts, approval requests, actions, or execution traces can overwhelm human or automated monitoring capacity, reducing the effectiveness of security oversight [38, 98].

Governance Evasion and Obfuscation. Malicious behavior may be fragmented, delayed, or distributed across agents,

tools, or time so that individual events remain below detection thresholds. Complex interactions and incomplete execution traces may further make attribution difficult and weaken governance controls [38].

Strategic Oversight Circumvention. Agents may alter their behavior according to monitoring, evaluation, or approval conditions. Related work on alignment faking and sleeper behavior illustrates the concern that evaluation-time behavior may not represent deployment-time behavior [47, 46].

Answer to RQ1

We characterize agentic AI threats through $\mathcal{T} = \langle \mathcal{S}, \mathcal{B}, \mathcal{P}, \mathcal{A} \rangle$, linking affected functional or system surfaces ($\mathcal{S}$), interaction or trust boundaries ($\mathcal{B}$), violated security properties ($\mathcal{P}$), and empirically examined architectures ($\mathcal{A}$). This cross-dimensional representation connects complementary threat attributes without encoding their temporal order or causal sequence. Each dimension may contain multiple labels, allowing threats to span several surfaces, boundaries, properties, and evaluated architectures.

V. EMPIRICAL ANALYSIS OF AGENTIC AI RED TEAMING

The taxonomy in Section IV characterizes the broader security landscape of agentic AI. However, the existence of a documented threat does not imply that it has been empirically demonstrated against an agentic system. To address RQ2 and RQ3, we therefore separately analyze studies that instantiate concrete attacks and evaluate their effects on LLM-based agents.

Following the selection criteria in Section II, we retain 22 empirical studies that (i) target an LLM-based agent or multi-agent system, (ii) implement or experimentally evaluate at least one concrete attack, (iii) provide sufficient information to characterize the target, attacker, architecture, and experimental outcome, and (iv) release at least one publicly accessible reproducibility artifact, such as implementation code, attack prompts or payloads, benchmark/test data, or evaluation scripts/configurations. This artifact criterion enables the empirical analysis to be grounded in studies for which at least part of the experimental methodology can be independently inspected or reused.

For each study, we map the demonstrated attacks to the $\mathcal{T} = \langle \mathcal{S}, \mathcal{B}, \mathcal{P}, \mathcal{A} \rangle$ representation introduced in Section IV-B and extract the demonstrated threat surface, attacker assumptions, evaluated target architecture, experimental environment, defense evaluation, and publicly released reproducibility artifacts.

A. Empirical Study Landscape

Table III summarizes the 22 artifact-backed empirical red-teaming studies retained for detailed evidence analysis. Compared with the broader threat inventory, this view distinguishes the *known attack surface* from the *empirically evaluated attack surface*. For each study, we report the demonstrated threats, attacker assumptions, target system, evaluated architecture,

experimental environment, defense evaluation, and publicly released reproducibility artifacts. Rather than assigning subjective low, medium, or high reproducibility ratings, we record observable artifact availability using four categories: implementation code (C), attack prompts or artifacts (P), released datasets or benchmarks (D), and evaluation scripts or configurations (E).

B. Threat and Architecture Coverage

Within the 22 artifact-backed studies reviewed, coverage concentrates on prompt injection, jailbreaks, memory poisoning, and tool-mediated attacks. Human-Agent oversight, systemic governance, and cross-session compromise are less represented in this subset. This distribution characterizes the selected evidence rather than the full extent of empirical research, and greater representation should not itself be interpreted as greater evaluation maturity.

Table IV applies the proposed framework to selected evaluations from three studies in the corpus. The attack descriptions and configurations follow the original papers; the surface, boundary, and property labels are our classifications under Section IV-B. Each row specifies an evaluated setting rather than aggregating all architectures examined by a study. The mappings associate security-relevant attributes without encoding temporal order or causal sequences.

The mapping separates shared attack surfaces from distinct security consequences. AgentDojo and AiTM both involve reasoning, but adversarial instructions enter through tool outputs and inter-agent messages, respectively. AiTM and MAMA share an Agent-Agent boundary, yet the selected evaluations concern behavioral redirection and confidentiality. A boundary-level classification alone would obscure this difference.

The corresponding outcome measures also differ. AgentDojo uses task-specific checks to determine whether attacker objectives are achieved in its execution environments. AiTM’s selected MMLU attack is assessed through the prescribed answer-label transformation, rather than an executed external action. MAMA measures recovery of synthetic PII from attacker outputs using exact matching supplemented by an inference-based judge [7, 9, 72]. These endpoints should remain distinct when comparing empirical coverage and interpreting attack success.

Architecture coverage is uneven: Table III records 16 single-agent studies and six multi-agent studies. Within the latter, AiTM compares Chain, Tree, Complete, and Random communication structures, while MAMA evaluates complete, circle, chain, tree, star-pure, and star-ring topologies [9, 72]. Their results also caution against assigning a universal security ranking to topologies. Chains are particularly vulnerable to AiTM’s communication attacks, whereas chain and tree configurations are generally among the lower-leakage settings in MAMA. This contrast does not establish contradictory findings: the studies differ in attacker control, communication protocols, and measured security properties. Architecture comparisons must therefore retain these experimental conditions.

Temporal scope requires similar precision. A multi-step execution, propagation between agents, and persistence across

TABLE III: Empirical red-teaming studies for agentic AI with publicly released reproducibility artifacts. Architecture badges denote target-system configurations empirically evaluated. Artifact badges denote resources released with or maintained for each study: C = implementation code, P = attack prompts, payloads, adversarial inputs, or attack-generation artifacts, D = released dataset, benchmark, or test cases, and E = evaluation scripts, experiment configurations, or evaluation harnesses. ^[R] links to the corresponding public artifact repository or release source.

Study	Year / Venue	Demonstrated Threats	Attacker	Target System / Framework	Empirical Arch.	Environment	Defense eval.	Artifacts
Greshake et al. [6]	2023 / AISec	Indirect prompt injection; persistent compromise; data exfiltration; unauthorized actions	B	LangChain, Bing Chat, Copilot		Web/tool	–	   ^[R]
Zhan et al. [53]	2024 / ACL	Indirect prompt injection; direct harm; data exfiltration	B	InjecAgent		Tool-integrated	–	    ^[R]
Gu et al. [99]	2024 / ICML	Multimodal jailbreak; cross-agent infection and propagation	W	Agent Smith		Multi-agent	–	    ^[R]
Amayuelas et al. [100]	2024 / EMNLP	Adversarial reasoning manipulation; trust sabotage; collaborative error propagation	G	Multi-agent debate		Multi-agent	✓	   ^[R]
Debenedetti et al. [7]	2024 / NeurIPS	Indirect prompt injection; malicious tool-mediated actions	B	AgentDojo		Tool-integrated	✓	    ^[R]
Chen et al. [8]	2024 / NeurIPS	Memory poisoning; knowledge-base poisoning; persistent backdoor activation	W	AgentPoison		Memory/RAG	–	    ^[R]
Yang et al. [101]	2024 / NeurIPS	Query-, observation-, and thought-triggered agent backdoors	W	AgentTuning, ToolBench		Tool-integrated	–	    ^[R]
Wang et al. [102]	2024 / ACL	Backdoor insertion; trigger-activated harmful actions	W	BadAgent		Web/tool	–	   ^[R]
Andriushchenko et al. [56]	2025 / ICLR	Jailbreak; harmful autonomous tool use; multi-step malicious behavior	B	AgentHarm		Tool-integrated	✓	   ^[R]
Zhang et al. [48]	2025 / ICLR	Direct/indirect prompt injection; memory poisoning; Plan-of-Thought backdoor; mixed attacks	G	Agent Security Bench		Tool/memory	✓	    ^[R]
Huang et al. [103]	2025 / ICML	Faulty/malicious-agent injection; reasoning-error propagation; cross-agent reliability degradation	G	CAMEL, MetaGPT, SPP, MAD, AgentVerse	  	Multi-agent	✓	    ^[R]
Zhang et al. [41]	2025 / ICML	Dynamic reasoning hijacking; adversarial-string optimization; indirect injection	W	UDora		Web/tool	–	    ^[R]
Fang et al. [104]	2025 / arXiv	Malicious MCP services; indirect prompt injection; adversarial tool responses	B	SafeMCP		MCP/tool	✓	   ^[R]
He et al. [9]	2025 / ACL	MITM communication manipulation; identity spoofing; cross-agent message attacks	B	AutoGen, CAMEL, MetaGPT, ChatDev	   	Multi-agent	–	  ^[R]
Gasmi et al. [105]	2025 / arXiv	Prompt injection; malicious tools; command injection; privilege escalation; resource abuse	B	Azure OpenAI, AWS Bedrock		Cloud/tool	–	   ^[R]
Zou et al. [106]	2025 / NeurIPS	Prompt injection; policy violations; adversarial agent deployment	B	Agent Red Teaming (ART)		Tool-integrated	–	   ^[R]
Atta et al. [107]	2026 / arXiv	Logic-layer injection; trigger execution; persistence; evasion; trace manipulation	B	LAAP		Tool/API	–	   ^[R]
Kavathekar et al. [108]	2026 / ACL	Multi-agent adversarial attacks; trust/coordination manipulation; cross-agent failures	B	TAMAS	  	Multi-agent/tool	–	    ^[R]
Jiang et al. [42]	2026 / arXiv	Intent hijacking; tool chaining; task injection; objective drifting; memory poisoning	B	AgentLAB		Long-horizon/tool	✓	    ^[R]
Zhang et al. [78]	2026 / ICLR	MCP-specific attacks across planning, tool invocation, and response handling	B/G	MCP Security Bench		MCP/tool	–	    ^[R]
Wang et al. [76]	2026 / AAAI	Tool-description poisoning; instruction hijacking; unauthorized tool actions	B	MCPTox		MCP/tool	–	    ^[R]
Liu et al. [72]	2026 / ACL	Cross-agent memory leakage; privacy extraction; topology-conditioned propagation	B	MAMA	   	Multi-agent/memory	–	    ^[R]

Attacker knowledge: **B** Black-box; **G** Gray-box; **W** White-box.

Architecture evidence:  Single-agent  Hierarchical  Sequential/Pipeline  Horizontal/Collaborative  Dynamic/Graph-based  Multi-agent, topology not reliably classifiable.

Defense eval.: ✓ At least one concrete defense experimentally evaluated; – No concrete defense evaluation.

Reproducibility artifacts:  Code  Prompts/attack artifacts  Dataset/benchmark  Evaluation scripts/configurations; ^[R] Public artifact repository/source.

sessions are different evaluation conditions. AgentLAB examines intent hijacking, tool chaining, task injection, objective drifting, and memory poisoning over extended interactions.

Its memory-poisoning procedure separates the injection of malicious memories from their retrieval and exploitation in a subsequent target scenario [42]. This distinction makes the

TABLE IV: Cross-dimensional mapping of selected empirical evaluations. Each row identifies a reported attack, its classification under the proposed framework, and a configuration evaluated in the cited study. The rows do not exhaust each study’s attacks or architecture coverage.

Study	Reported attack	Surface (S)	Boundary (B)	Property (P)	Evaluated setting (A)
AgentDojo [7]	Instructions injected into tool-returned data redirect the agent toward an attacker-defined task.	Reasoning and Planning; Action and Tools	Agent–Tool	Goal Integrity; Action and Authorization Integrity	Single-agent tool-calling configuration (S)
AiTM [9]	Intercepted inter-agent messages induce attacker-specified transformations of MMLU answer labels.	Reasoning and Planning	Agent–Agent	Goal Integrity; Interaction and Trust Integrity	Three-agent chain in AutoGen and CAMEL (P)
MAMA [72]	An information-seeking agent extracts synthetic PII initially available only to a target agent.	Memory and Knowledge	Agent–Agent	Confidentiality	Complete communication graph with 4–6 agents (C)

Architecture labels: S = Single-agent; P = Sequential/Pipeline; C = Horizontal/Collaborative. The Agent–Tool boundary denotes the interface through which tool-returned content enters the agent, irrespective of the underlying document or service that supplies that content.

relationship between retained state and later behavior explicit. Evaluations should accordingly report the interaction horizon, the state retained between phases, and whether compromise survives a task or session boundary. Immediate attack success alone does not establish persistent system compromise.

C. Benchmarks and Evaluation Practices

A central challenge in comparing agent-security research is the diversity of evaluation environments. Some studies construct attack-specific settings, whereas others introduce reusable benchmarks that standardize tasks, tools, adversarial conditions, and success criteria. Table V summarizes *11 representative security benchmarks* spanning indirect prompt injection, memory compromise, harmful autonomous behavior, embodied safety, multi-agent security, long-horizon attacks, and MCP/tool-security evaluation. The benchmarks differ substantially in evaluation scale, target architecture, environmental realism, and outcome definitions; consequently, their reported metrics should not be interpreted as directly comparable.

The benchmark landscape shows a clear expansion in both attack-surface coverage and evaluation realism. InjecAgent provides a focused benchmark for indirect prompt injection across heterogeneous tool interactions [53], while AgentDojo introduces dynamic environments in which attack success can be evaluated jointly with benign task utility [7]. AgentPoison isolates persistent compromise of memory and knowledge retrieval [8], whereas SafeAgentBench extends evaluation to embodied settings in which unsafe planning may result in hazardous physical actions [93].

AgentHarm shifts the evaluation target from harmful text generation toward whether autonomous agents can successfully complete malicious multi-step tasks [56]. ASB broadens coverage across system prompts, user prompts, tool use, memory retrieval, planning, and defense evaluation [48], while large-scale public red teaming evaluates adversarial robustness across diverse frontier models and agent configurations [106].

More recent benchmarks increasingly target system properties that are specific to agentic execution. TAMAS evaluates adversarial behavior across multiple multi-agent coordination structures [108], while AgentLAB targets adaptive attacks over longer agent–environment trajectories [42]. MCP Security Bench evaluates attacks across the MCP tool-use pipeline, including planning, tool invocation, and response handling [78], whereas MCPTox focuses specifically on tool poisoning against real-world MCP servers and authentic tool ecosystems [76]. Together, these benchmarks extend evaluation beyond model-level prompt robustness toward coordination, persistence, tool trust, and system-level effects.

Despite this progress, benchmark coverage remains fragmented. Existing suites specialize in different threat families, architectures, environments, and outcome definitions, making results difficult to compare directly. Attack Success Rate (ASR) remains common, but the same term may represent generation of prohibited text, selection of a malicious tool, disclosure of information, completion of a harmful task, manipulation of another agent, or modification of an external system state.

For agentic systems, evaluation should therefore distinguish at least three levels of attack success: *model compromise*, where adversarial content changes the model response; *agent compromise*, where the attack changes the agent’s goal, state, plan, or selected action; and *system impact*, where the attack produces a verifiable unauthorized, harmful, or persistent state change. This distinction avoids treating textual compliance and executed harm as equivalent security outcomes.

D. Defense Evaluation

Seven of the 22 reviewed studies report a concrete defense evaluation. This count establishes the presence of defense experiments, not their coverage, adaptivity, or effectiveness against different attacks. Evaluations should distinguish pro-

TABLE V: Representative benchmarks and evaluation environments for agentic AI security. Architecture badges denote the target-system configurations empirically evaluated. Evaluation scale is reported using the native unit of each benchmark (e.g., tasks, security cases, or attack instances). ^[R] links to the corresponding public benchmark repository or release source.

Benchmark	Year / Venue	Primary Security Focus	Evaluation Scale	Empirical Arch.	Target Setting	Evaluation Environment	Typical Metrics
InjecAgent [53] ^[R]	2024 / ACL	Indirect prompt injection; direct harm; data exfiltration	1,054 security cases		Tool-integrated agents	17 user tools; 62 attacker tools; 30 LLM agents	ASR; direct harm; exfiltration
AgentDojo [7] ^[R]	2024 / NeurIPS	Indirect prompt injection; attack/defense evaluation	97 tasks; 629 security cases		Tool-using agents	Dynamic email, banking, travel, and productivity environments	Utility; ASR; security violation
AgentPoison [8] ^[R]	2024 / NeurIPS	Memory and knowledge-base poisoning; persistent backdoors	Attack-specific evaluations		Memory/RAG agents	Long-term memory and knowledge-retrieval settings	ASR; benign utility; retrieval effectiveness
SafeAgentBench [93] ^[R]	2024 / arXiv	Embodied-agent safety; hazardous task planning	750 tasks		Embodied agents	10 hazard categories; 3 task types; simulated embodied environment	Task success; rejection; safety
AgentHarm [56] ^[R]	2025 / ICLR	Jailbreak; harmful autonomous tool use	110 base tasks; 440 augmented		Tool-using agents	11 harm categories; multi-step malicious tasks	Refusal; harmful-task completion
Agent Security Bench (ASB) [48] ^[R]	2025 / ICLR	Prompt injection; memory poisoning; planning backdoors; attack/defense evaluation	50 benign; 400 attack tasks		Tool/memory agents	10 scenarios; 10 agents; 400+ tools; 27 attack/defense methods	ASR; utility; security-utility trade-off
AI Agent Red-Team Benchmark [106] ^[R]	2025 / NeurIPS	Large-scale adversarial robustness; prompt injection; policy violations	1.8M attacks		Frontier LLM agents	22 frontier LLMs; 44 realistic deployment scenarios	ASR; transferability; policy violation
TAMAS [108] ^[R]	2026 / ACL	Adversarial multi-agent security; coordination and trust attacks	300 adversarial; 100 harmless tasks	  	Multi-agent systems	5 scenarios; 6 attack types; 211 tools; 10 LLMs; 3 interaction configurations	ERS; robustness; task effectiveness
AgentLAB [42] ^[R]	2026 / arXiv	Adaptive long-horizon attacks; persistent agent compromise	644 attack cases		Long-horizon agents	28 environments; 5 attack families; multi-turn agent-environment trajectories	ASR; task outcome; defense effectiveness
MCP Security Bench (MSB) [78] ^[R]	2026 / ICLR	MCP-specific attacks across planning, tool invocation, and response handling	2,000 attack instances		MCP/tool agents	10 domains; 405 tools; 12 attack categories; 9 LLM agents	ASR; NRP; task performance
MCPTox [76] ^[R]	2026 / AAAI	Tool poisoning; malicious MCP metadata and instructions	1,348 malicious cases		MCP/tool agents	45 live MCP servers; 353 authentic tools; 10 risk categories	ASR; refusal rate; tool-poisoning robustness
Architecture evidence:  Single-agent  Hierarchical  Sequential/Pipeline  Horizontal/Collaborative  Dynamic/Graph-based. ^[R] Public benchmark repository/source.							

tection against a tested attack implementation from evidence supporting a broader security claim.

AgentDojo and ASB support comparisons of attacks and defenses within common tasks, including their effects on benign task performance [7, 48]. Such comparisons should report both *security* and *utility*, alongside operational costs such as model calls, latency, and human intervention. Restricting tools or rejecting external information may reduce attack success while also impairing legitimate tasks.

The coverage limitations discussed in Section V-B motivate defense evaluation for persistent compromise, multi-agent interactions, and systemic threats. Tests should specify attacker knowledge, access, and budgets, and whether attackers adapt to the evaluated defense.

Defense evaluation should also distinguish the stages at which controls operate. Input filtering, provenance checks, authorization enforcement, memory-integrity checks, and trajectory monitoring address different points in agent execution. Their effectiveness should be assessed across action sequences, since individually permitted actions may collectively violate security constraints.

Answer to RQ2

Within the 22 artifact-backed studies reviewed, coverage concentrates on prompt/reasoning, memory, and tool-mediated attacks; 16 studies evaluate single-agent systems. Cross-session compromise, Human-Agent oversight, and systemic threats receive less coverage in this subset. These findings characterize the selected corpus rather than establish field-wide under-evaluation.

E. Reproducibility and Evidence Maturity

The 22-study empirical subset includes only studies with at least one public artifact. Table III distinguishes implementation code (C), attack prompts, payloads, or generation artifacts (P), datasets, benchmarks, or test cases (D), and evaluation scripts, configurations, or harnesses (E). This selection excludes studies without public artifacts, even when their publications document empirical findings. Coverage gaps therefore concern the selected subset, while our artifact analysis neither estimates field-wide availability nor establishes successful independent reproduction.

Artifact availability supports inspection but does not guarantee reproducibility. Releases range from selected prompts or code to complete benchmark environments. Reproduction may still depend on proprietary models, changing APIs, external services, and nondeterministic execution. Artifact types should therefore be distinguished from the completeness and executability of the released materials.

Agentic systems also require capturing *execution state*. Identical prompts and models can yield different trajectories when memory, tool responses, or agent interactions change. Reproducibility therefore requires documenting model versions and settings, prompts, tool versions, permissions, initial memory and environment state, agent topology, and execution traces.

Outcome definitions further complicate comparison. The reviewed benchmarks measure refusal, textual compliance, tool selection, task completion, benign utility, and external state changes. Evaluation should distinguish *model influence*, *agent redirection*, and *verified system impact*, rather than treating these outcomes as equivalent attack successes.

These observations highlight the need to complement threat discovery with evaluation across diverse architectures, extended interactions, adaptive defenses, and reproducible environments.

Answer to RQ3

Evaluation maturity varies across the reviewed studies and benchmarks. Heterogeneous metrics, limited adaptive defense evaluation, short-horizon experiments, architectural imbalance, and incomplete execution-state capture constrain comparison and reproducibility. Public artifacts support inspection, but their availability alone does not establish reproducible results.

VI. RESEARCH GAPS AND FUTURE DIRECTIONS

The answers to RQ1–RQ3 identify uneven coverage within the reviewed evidence. The selected studies emphasize prompt manipulation, jailbreaks, memory poisoning, and tool-mediated attacks, with less coverage of Human–Agent oversight, systemic dependencies, and cross-session compromise. These observations motivate 13 open research questions addressing evaluation coverage and methodology; they do not establish the absence of empirical research beyond the selected corpus.

A. Evaluation Maturity

The selected corpus includes 16 single-agent and six multi-agent studies; seven report defense experiments. Ten of the 11 benchmarks target single-agent configurations. These counts describe coverage, not defense adaptivity, experimental realism, or reproducibility.

Evidence quality. Evaluation maturity should reflect model and architecture coverage, environmental realism, attacker assumptions, temporal scope, outcome verification, defense evaluation, and reproducibility support. These attributes

require explicit assessment beyond publication counts or artifact availability.

Open Question OQ1

How can evidence maturity be assessed consistently across threats with different attacker assumptions, architectures, and evaluation settings?

Verified security outcomes. A manipulated response does not establish agent redirection or an executed unauthorized action. Following Section V-C, evaluations should distinguish model compromise, agent compromise, and verified system impact, using operational criteria appropriate to each claimed consequence.

Open Question OQ2

How can measures of model compromise, agent compromise, and system impact be defined and validated for comparison across tasks and environments?

Architecture-aware evaluation. The multi-agent studies include comparisons of coordination structures, but findings from one configuration do not establish security in another. Comparisons should specify topology, delegation, shared state, communication protocols, and attacker placement, distinguishing graph-based organization from runtime topology changes.

Open Question OQ3

Which security findings generalize across agent architectures, and which depend on topology, delegation, shared state, or coordination mechanisms?

B. Runtime Security, Misalignment, and Persistent Compromise

The empirical literature remains strongly concentrated on attacks that produce relatively immediate effects. Increasingly autonomous and long-lived agents introduce a different security problem in which compromise may emerge gradually, persist through state, or become visible only after a sequence of individually benign actions.

Runtime misalignment and trajectory-level security. Current safeguards predominantly assess prompts, outputs, or individual tool calls. Such mechanisms are less suited to detecting gradual objective drift, reward or specification gaming, deceptive behavior, or trajectories in which individually permissible actions collectively violate the intended goal. Runtime security therefore requires reasoning over the agent’s goal, intermediate state, execution history, delegated authority, and downstream consequences rather than evaluating actions independently.

Open Questions OQ4–OQ5

OQ4: How can runtime monitors distinguish legitimate goal adaptation from adversarial goal drift or strategic misalignment without substantially restricting useful agent autonomy?

OQ5: Can trajectory-level security invariants detect harmful multi-step behavior before irreversible actions occur when each individual action appears benign?

Persistent and cross-session compromise. Memory poisoning changes the security model because adversarial influence may survive the interaction through which it was introduced. Long-lived agents therefore require mechanisms for memory provenance, integrity verification, rollback, expiry, compartmentalization, and selective forgetting. The problem becomes more difficult when agents autonomously summarize, consolidate, or rewrite memories over time.

Open Question OQ6

How can legitimate learning and memory adaptation be distinguished from adversarially induced belief drift, and how can compromised state be safely rolled back without destroying useful learned information?

Deceptive and trigger-dependent behavior. Sleeper and strategically deceptive behavior challenge evaluation methodologies that assume test-time behavior represents deployment-time behavior. Hidden triggers, delayed activation, monitor-aware behavior, and long interaction histories may cause failures to remain invisible during conventional testing [46, 47, 45]. This motivates evaluation not only of whether agents behave safely under observation, but also whether their behavior changes when monitoring or deployment conditions differ.

Open Question OQ7

How can security evaluations reliably detect deceptive or trigger-dependent agent behavior when activation depends on hidden triggers, long interaction histories, or the agent’s awareness of being monitored?

C. Multi-Agent and Human–Agent Security

Multi-agent systems create security dependencies that cannot be captured by replicating single-agent tests across several models. Their security depends on identity, delegation, communication, shared state, coordination mechanisms, and assumptions about other participants. At the same time, humans remain integral to agentic systems as principals, supervisors, authorization authorities, and potential targets.

Emergent multi-agent attacks. Byzantine behavior, consensus poisoning, collusion, and cross-agent propagation remain comparatively underexplored [89, 90, 91, 108]. Future work should evaluate how compromise depends

on communication topology, the number and placement of malicious agents, delegation structure, aggregation mechanisms, and persistence of shared state. Security properties observed in collaborative peer systems should not be assumed to generalize to hierarchical, pipeline, or dynamic architectures.

Open Question OQ8

How do the number, position, and coordination strategy of compromised agents affect attack propagation and collective decisions across hierarchical, pipeline, collaborative, and dynamic architectures?

Shared-state integrity and provenance. Collaborative memories and common knowledge stores improve coordination but also create persistent propagation surfaces. Recent work illustrates how malicious state can influence agents that were not directly exposed to the original adversary [68, 69]. Important directions include provenance-aware memory, constrained write authority, cross-agent validation, contamination detection, and recovery from poisoned shared state.

Open Question OQ9

How can cross-agent provenance identify the origin, propagation path, and downstream influence of poisoned messages, beliefs, or shared-memory entries?

Identity, delegation, and authority. Multi-agent systems require stronger notions of identity and authorization than simple message exchange. Authenticated agent identities, scoped delegation, privilege attenuation, revocation, and provenance across delegation chains become particularly important in hierarchical systems, where compromise of a coordinator may implicitly transfer malicious authority to many downstream agents.

Human–Agent trust and meaningful oversight. Human oversight is frequently treated as a defense, yet humans are also part of the attack surface. Agents may exploit automation bias, generate misleading explanations, overwhelm approval mechanisms, or encourage users to delegate excessive authority. Conversely, attackers may socially engineer users into authorizing malicious agent actions. Simply inserting an approval dialog therefore does not establish meaningful human control.

Open Question OQ10

When does human oversight meaningfully reduce agentic risk, and when do automation bias, approval fatigue, inadequate context, or unsafe delegation make the human another exploitable component of the system?

D. System-Level, Supply-Chain, and Cyber-Physical Security

As agents shift from language generation to autonomous execution, their security increasingly depends on software ecosystems, orchestration infrastructure, external services, and physical environments. These surfaces receive less coverage than prompt- and model-level attacks in the reviewed evidence.

Agentic supply chains and continuous component trust. Modern agents increasingly depend on third-party tools, plugins, MCP servers, models, reusable skills, prompts, libraries, and external services. Existing work demonstrates several forms of malicious tool behavior, but systematic evaluation of provenance, update integrity, dependency compromise, and post-installation behavior remains limited. Moreover, a component should not necessarily remain trusted simply because it was benign when first installed: rug-pull and update attacks show that behavior may change after trust has been established.

Open Question OQ11

How can dynamically discovered agents, tools, skills, plugins, and MCP servers be continuously authenticated, monitored, and revoked when their behavior or implementation changes after deployment?

Infrastructure and orchestration security. Multi-agent platforms increasingly rely on orchestrators, message buses, credential stores, memory services, sandboxes, and cloud execution environments. Compromise of these components may affect multiple agents simultaneously. Red teaming should therefore target the orchestration and execution plane in addition to individual agents, including message routing, task assignment, shared state, credential management, and isolation boundaries.

Cyber-physical agent systems. Agents connected to robotics, vehicles, laboratory automation, industrial control, or other physical systems create consequences that cannot be reversed as easily as generated text. Evaluation in these settings should consider adversarial observations, sensor manipulation, unsafe action sequences, delayed feedback, and interactions between digital compromise and physical state. Security metrics must similarly move beyond textual attack success toward physical safety and verifiable environmental consequences.

Open Question OQ12

What security invariants, runtime safeguards, and recovery mechanisms are required when autonomous agent actions modify irreversible or safety-critical physical state?

Safe recovery and containment. Increasing autonomy also raises the question of what happens after compromise is detected. Important mechanisms include credential revocation, memory rollback, safe-state restoration, interruption of delegated tasks, isolation of compromised agents, and

recovery of shared state. Recovery is especially challenging where malicious actions have already propagated across agents or modified external systems.

E. Toward Reproducible and Standardized Evaluation

The benchmarks reviewed in Section V-C vary in threats, architectures, environments, and outcome definitions, limiting direct comparison across attacks and defenses.

Standardized threat and attacker models. The proposed representation, $\mathcal{T} = \langle \mathcal{S}, \mathcal{B}, \mathcal{P}, \mathcal{A} \rangle$, links affected surfaces, interaction boundaries, violated security properties, and reported empirical architectures. Attacker knowledge, capabilities, privileges, temporal scope, and operational success criteria are separate experimental attributes that benchmarks should report alongside this mapping.

Comparable security metrics. Attack Success Rate requires an explicit definition of success. Evaluations should distinguish model influence, agent redirection, and verified system impact, and report benign task utility alongside attack effectiveness. Where relevant, they should also measure persistence, propagation, defense overhead, false-positive rates, and the severity of resulting state changes.

Reproducibility and environment state. Reproducibility requires documenting both the agent configuration and its execution environment. Releases should include model versions and settings, prompts, tool schemas and versions, permissions, initial memory, environment snapshots, agent topology, execution traces, and evaluation scripts where possible. Snapshotting and replay can support controlled comparisons, while dependencies on changing services and nondeterministic execution should be documented.

Open Question OQ13

How can agent-security benchmarks preserve reproducibility and historical comparability as models, tools, environments, and multi-agent interactions change?

Continuous benchmarks and adaptive defenses. Maintained evaluation suites should preserve versioned tasks, configurations, and results while testing updated systems. Defense evaluation should include adaptive attackers that respond to the deployed mitigation, with attacker access and budgets explicitly reported.

These directions connect threat modeling to measurable security outcomes. The research agenda emphasizes evaluation across architectures, interaction horizons, and system boundaries, supported by explicit threat models, comparable metrics, and reproducible execution records.

VII. CONCLUSION

Agentic AI extends LLM security to persistent state, autonomous actions, delegated authority, and interactions with other agents and external environments. Through a structured review of 66 studies, we developed a cross-dimensional

framework linking affected system surfaces, trust boundaries, security properties, and empirically examined architectures. Our analysis of 22 artifact-backed red-teaming studies and 11 representative security benchmarks characterizes the coverage and maturity of the reviewed evidence. Within this subset, coverage concentrates on prompt/reasoning, memory, and tool-mediated attacks, while persistent, complex multi-agent, Human-Agent, systemic, and long-horizon threats receive less attention. Variation in architectures, metrics, defense evaluation, and artifact completeness limits comparison and reproducibility. These findings concern the selected studies rather than the full extent of empirical research. The 13 open research questions provide an agenda for complementing threat discovery with more systematic, realistic, and reproducible security evaluation.

REFERENCES

- [1] L. Wang, C. Ma, X. Feng, Z. Zhang, H. Yang, J. Zhang, Z. Chen, J. Tang, X. Chen, Y. Lin *et al.*, “A survey on large language model based autonomous agents,” *Frontiers of Computer Science*, vol. 18, no. 6, p. 186345, 2024.
- [2] Z. Xi, W. Chen, X. Guo, W. He, Y. Ding, B. Hong, M. Zhang, J. Wang, S. Jin, E. Zhou *et al.*, “The rise and potential of large language model based agents: A survey,” *Science China Information Sciences*, vol. 68, no. 2, p. 121101, 2025.
- [3] Q. Wu, G. Bansal, J. Zhang, Y. Wu, B. Li, E. Zhu, L. Jiang, X. Zhang, S. Zhang, J. Liu *et al.*, “Autogen: Enabling next-gen llm applications via multi-agent conversations,” in *First conference on language modeling*, 2024.
- [4] S. Yao, J. Zhao, D. Yu, N. Du, I. Shafran, K. R. Narasimhan, and Y. Cao, “React: Synergizing reasoning and acting in language models,” in *The eleventh international conference on learning representations*, 2022.
- [5] N. Shinn, F. Cassano, A. Gopinath, K. Narasimhan, and S. Yao, “Reflexion: Language agents with verbal reinforcement learning,” *Advances in neural information processing systems*, vol. 36, pp. 8634–8652, 2023.
- [6] K. Greshake, S. Abdelnabi, S. Mishra, C. Endres, T. Holz, and M. Fritz, “Not what you’ve signed up for: Compromising real-world llm-integrated applications with indirect prompt injection,” in *Proceedings of the 16th ACM workshop on artificial intelligence and security*, 2023, pp. 79–90.
- [7] E. Debenedetti, J. Zhang, M. Balunovic, L. Beurer-Kellner, M. Fischer, and F. Tramèr, “Agentdojo: A dynamic environment to evaluate prompt injection attacks and defenses for llm agents,” *Advances in Neural Information Processing Systems*, vol. 37, pp. 82 895–82 920, 2024.
- [8] Z. Chen, Z. Xiang, C. Xiao, D. Song, and B. Li, “AgentPoison: Red-teaming LLM agents via poisoning memory or knowledge bases,” in *Advances in Neural Information Processing Systems*, vol. 37, 2024.
- [9] P. He, Y. Lin, S. Dong, H. Xu, Y. Xing, and H. Liu, “Red-teaming llm multi-agent systems via communication attacks,” in *Findings of the Association for Computational Linguistics: ACL 2025*, 2025, pp. 6726–6747.
- [10] A. Abuadbba, N. Sultan, S. Nepal, and S. Jha, “Human society-inspired approaches to agentic ai security: The 4c framework,” *arXiv preprint arXiv:2602.01942*, 2026.
- [11] N. Singh, A. Abuadbba, Y. Gao, S. Nepal, and H. Kim, “Shifting from injection to interaction: Rethinking web security in the age of llms and beyond,” *arXiv preprint arXiv:2609.03999*, 2026.
- [12] Z. Deng, Y. Guo, C. Han, W. Ma, J. Xiong, S. Wen, and Y. Xiang, “AI agents under threat: A survey of key security challenges and future pathways,” *ACM Computing Surveys*, vol. 57, no. 7, pp. 1–36, 2025.
- [13] A. Shahriar, M. N. Rahman, S. Ahmed, F. Sadeque, and M. R. Parvez, “A survey on agentic security: Applications, threats and defenses,” *arXiv preprint arXiv:2510.06445*, 2025.
- [14] A. Chhabra, S. Datta, S. K. Nahin, and P. Mohapatra, “Agentic AI security: Threats, defenses, evaluation, and open challenges,” *IEEE Access*, vol. 14, pp. 49 455–49 482, 2026.
- [15] J. Kim, X. Liu, Z. Wang, S. Qiu, B. Li, W. Guo, and D. Song, “The attack and defense landscape of agentic AI: A comprehensive survey,” *arXiv preprint arXiv:2603.11088*, 2026, accepted to the 35th USENIX Security Symposium (USENIX Security 2026); extended version.
- [16] OWASP GenAI Security Project, “OWASP Top 10 for Agentic Applications,” 2025, agentic Security Initiative, accessed September 2026. [Online]. Available: <https://genai.owasp.org/2025/12/09/owasp-top-10-for-agentic-applications-the-benchmark-for-agentic-security-in-the-age-of-autonomous-ai/>
- [17] MITRE, “ATLAS: Adversarial threat landscape for artificial-intelligence systems,” <https://atlas.mitre.org/>, 2026, version 2026.01, accessed September 2026.
- [18] N. Wang, K. Walter, Y. Gao, and A. Abuadbba, “Understanding the adversarial landscape of large language models through the lens of attack objectives,” *IEEE Security & Privacy*, vol. 24, no. 1, pp. 53–60, 2026.
- [19] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, “Attention is all you need,” *Advances in neural information processing systems*, vol. 30, 2017.
- [20] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell *et al.*, “Language models are few-shot learners,” *Advances in neural information processing systems*, vol. 33, pp. 1877–1901, 2020.
- [21] A. Abuadbba, K. Moore, D. Goel, C. Hicks, V. Mavroudis, B. Hasircioglu, and P. Jennings, “From promise to peril: rethinking cybersecurity red and blue teaming in the age of llms,” *IEEE Security & Privacy*, vol. 24, no. 2, pp. 53–63, 2026.
- [22] A. Bandi, B. Kongari, R. Naguru, S. Pasnoor, and S. V. Vilipala, “The rise of agentic ai: A review of definitions, frameworks, architectures, applications, evaluation metrics, and challenges,” *Future Internet*, vol. 17, no. 9, p.

- 404, 2025.
- [23] M. Abou Ali, F. Dornaika, and J. Charafeddine, “Agentic ai: a comprehensive survey of architectures, applications, and future directions,” *Artificial Intelligence Review*, vol. 59, no. 1, p. 11, 2025.
 - [24] X. Dong, X. Zhang, W. Bu, D. Zhang, and F. Cao, “A survey of llm-based agents: Theories, technologies, applications and suggestions,” in *2024 3rd International Conference on Artificial Intelligence, Internet of Things and Cloud Computing Technology (AIoTC)*. IEEE, 2024, pp. 407–413.
 - [25] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W.-t. Yih, T. Rocktäschel *et al.*, “Retrieval-augmented generation for knowledge-intensive nlp tasks,” *Advances in neural information processing systems*, vol. 33, pp. 9459–9474, 2020.
 - [26] N. L. B. Nguyen, W. Ma, V. Vo, A. Abuadbbba, M. Fang, J. Zhang, and Y. Xiang, “Five queries are enough: Query-efficient and surrogate-free membership inference attacks on rag via entailment,” *arXiv preprint arXiv:2605.24312*, 2026, accepted to the 35th USENIX Security Symposium (USENIX Security 2026).
 - [27] T. Schick, J. Dwivedi-Yu, R. Dessì, R. Raileanu, M. Lomeli, E. Hambro, L. Zettlemoyer, N. Cancedda, and T. Scialom, “Toolformer: Language models can teach themselves to use tools,” *Advances in neural information processing systems*, vol. 36, pp. 68 539–68 551, 2023.
 - [28] Significant Gravitas, “Autogpt,” 2026, accessed: April 17, 2026. Available online: <https://github.com/Significant-Gravitas/AutoGPT>. [Online]. Available: <https://github.com/Significant-Gravitas/AutoGPT>
 - [29] yoheinakajima, “Babyagi,” 2026, accessed: April 17, 2026. Available online: <https://github.com/yoheinakajima/babyagi>. [Online]. Available: <https://github.com/yoheinakajima/babyagi>
 - [30] Z. Liu, W. Yao, J. Zhang, L. Xue, S. Heinecke, R. RN, Y. Feng, Z. Chen, J. C. Niebles, D. Arpit *et al.*, “Bollaa: Benchmarking and orchestrating llm autonomous agents,” in *ICLR 2024 Workshop on Large Language Model (LLM) Agents*, 2024.
 - [31] S. Hong, M. Zhuge, J. Chen, X. Zheng, Y. Cheng, J. Wang, C. Zhang, Z. Wang, S. K. S. Yau, Z. Lin *et al.*, “Metagpt: Meta programming for a multi-agent collaborative framework,” in *The twelfth international conference on learning representations*, 2023.
 - [32] C. Qian, W. Liu, H. Liu, N. Chen, Y. Dang, J. Li, C. Yang, W. Chen, Y. Su, X. Cong *et al.*, “Chatdev: Communicative agents for software development,” in *Proceedings of the 62nd annual meeting of the association for computational linguistics (volume 1: Long papers)*, 2024, pp. 15 174–15 186.
 - [33] G. Li, H. Hammoud, H. Itani, D. Khizbullin, and B. Ghanem, “Camel: Communicative agents for” mind” exploration of large language model society,” *Advances in neural information processing systems*, vol. 36, pp. 51 991–52 008, 2023.
 - [34] Y. Du, S. Li, A. Torralba, J. B. Tenenbaum, and I. Mor-datch, “Improving factuality and reasoning in language models through multiagent debate,” in *Forty-first international conference on machine learning*, 2024.
 - [35] T. Liang, Z. He, W. Jiao, X. Wang, Y. Wang, R. Wang, Y. Yang, S. Shi, and Z. Tu, “Encouraging divergent thinking in large language models through multi-agent debate,” in *Proceedings of the 2024 conference on empirical methods in natural language processing*, 2024, pp. 17 889–17 904.
 - [36] Z. Liu, Y. Zhang, P. Li, Y. Liu, and D. Yang, “Dynamic llm-agent network: An llm-agent collaboration framework with agent team optimization,” *arXiv preprint arXiv:2310.02170*, 2023.
 - [37] M. Zhuge, W. Wang, L. Kirsch, F. Faccio, D. Khizbullin, and J. Schmidhuber, “Gptswarm: Language agents as optimizable graphs,” in *Forty-first International Conference on Machine Learning*, 2024.
 - [38] V. S. Narajala and O. Narayan, “Securing agentic ai: A comprehensive threat model and mitigation framework for generative ai agents,” *arXiv preprint arXiv:2504.19956*, 2025.
 - [39] Y. Guo, P. Liu, W. Ma, Z. Deng, X. Zhu, P. Di, X. Xiao, and S. Wen, “Systematic analysis of mcp security,” *arXiv preprint arXiv:2508.12538*, 2025.
 - [40] F. Perez and I. Ribeiro, “Ignore previous prompt: Attack techniques for language models,” *arXiv preprint arXiv:2211.09527*, 2022.
 - [41] J. Zhang, S. Yang, and B. Li, “Udora: A unified red teaming framework against llm agents by dynamically hijacking their own reasoning,” *arXiv preprint arXiv:2503.01908*, 2025.
 - [42] T. Jiang, Y. Wang, J. Liang, and T. Wang, “Agent-LAB: Benchmarking LLM agents against long-horizon attacks,” *arXiv preprint arXiv:2602.16901*, 2026.
 - [43] A. Bondarenko, D. Volk, D. Volkov, and J. Ladish, “Demonstrating specification gaming in reasoning models,” *arXiv preprint arXiv:2502.13295*, 2025.
 - [44] Ö. V. Çağatan and X. Zhao, “Reward hacking in language model agents: Revisiting AI safety gridworlds,” *arXiv preprint arXiv:2606.15385*, 2026.
 - [45] J. Betley, D. C. H. Tan, N. Warncke, A. Szytber-Betley, X. Bao, M. Soto, N. Labenz, and O. Evans, “Emergent misalignment: Narrow finetuning can produce broadly misaligned LLMs,” in *Proceedings of the 42nd International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, vol. 267. PMLR, 2025, pp. 4043–4068.
 - [46] E. Hubinger, C. Denison, J. Mu, M. Lambert, M. Tong, M. MacDiarmid, T. Lanham, D. M. Ziegler, T. Maxwell, N. Cheng, A. Jermyn, A. Askeel *et al.*, “Sleepers agents: Training deceptive LLMs that persist through safety training,” *arXiv preprint arXiv:2401.05566*, 2024.
 - [47] R. Greenblatt, C. Denison, B. Wright, F. Roger, M. MacDiarmid, S. Marks, J. Treutlein, T. Belonax, J. Chen, D. Duvenaud *et al.*, “Alignment faking in large language models,” *arXiv preprint arXiv:2412.14093*,

- 2024.
- [48] H. Zhang, J. Huang, K. Mei, Y. Yao, Z. Wang, C. Zhan, H. Wang, and Y. Zhang, “Agent security bench (ASB): Formalizing and benchmarking attacks and defenses in LLM-based agents,” in *International Conference on Learning Representations (ICLR)*, 2025. [Online]. Available: <https://openreview.net/forum?id=V4y0CpX4hK>
 - [49] F. He, T. Zhu, D. Ye, B. Liu, W. Zhou, and P. S. Yu, “The emerged security and privacy of llm agent: A survey with case studies,” *ACM Computing Surveys*, vol. 58, no. 6, pp. 1–36, 2025.
 - [50] Y. Li, H. Wen, W. Wang, X. Li, Y. Yuan, G. Liu, J. Liu, W. Xu, X. Wang, Y. Sun *et al.*, “Personal llm agents: Insights and survey about the capability, efficiency and security,” *arXiv preprint arXiv:2401.05459*, 2024.
 - [51] M. Yu, F. Meng, X. Zhou, S. Wang, J. Mao, L. Pan, T. Chen, K. Wang, X. Li, Y. Zhang *et al.*, “A survey on trustworthy llm agents: Threats and countermeasures,” in *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*, 2025, pp. 6216–6226.
 - [52] N. Maloyan and D. Namiot, “Prompt injection attacks on agentic coding assistants: A systematic analysis of vulnerabilities in skills, tools, and protocol ecosystems,” *International Journal of Open Information Technologies*, vol. 14, no. 2, pp. 1–10, 2026.
 - [53] Q. Zhan, Z. Liang, Z. Ying, and D. Kang, “Injecagent: Benchmarking indirect prompt injections in tool-integrated large language model agents,” in *Findings of the Association for Computational Linguistics: ACL 2024*, 2024, pp. 10 471–10 506.
 - [54] H. Chang, E. Bao, X. Luo, and T. Yu, “Overcoming the retrieval barrier: Indirect prompt injection in the wild for llm systems,” *arXiv preprint arXiv:2601.07072*, 2026.
 - [55] S. Johnson, V. Pham, and T. Le, “The dangers of indirect prompt injection attacks on llm-based autonomous web navigation agents: A demonstration,” in *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, 2025, pp. 729–738.
 - [56] M. Andriushchenko, A. Souly, M. Dziemian, D. Duenas, M. Lin, J. Wang, D. Hendrycks, A. Zou, Z. Kolter, M. Fredrikson, Y. Gal, and X. Davies, “AgentHarm: A benchmark for measuring harmfulness of LLM agents,” in *The Thirteenth International Conference on Learning Representations*, 2025. [Online]. Available: https://proceedings.iclr.cc/paper_files/paper/2025/hash/c493d23af93118975cdbe32cbe7323f5-Abstract-Conference.html
 - [57] J. Y. F. Chiang, S. Lee, J.-B. Huang, F. Huang, and Y. Chen, “Why are web ai agents more vulnerable than standalone llms? a security analysis,” *arXiv preprint arXiv:2502.20383*, 2025.
 - [58] T. Hagendorff, E. Derner, and N. Oliver, “Large reasoning models are autonomous jailbreak agents,” *Nature Communications*, 2026.
 - [59] Y. Mao, P. Liu, T. Cui, C. Liu, M. Xing, and D. You, “Stop fixating on prompts: Reasoning hijacking and constraint tightening for red-teaming llm agents,” *arXiv preprint arXiv:2604.05549*, 2026.
 - [60] G. Zhao, H. Wu, X. Zhang, and A. V. Vasilakos, “Shadowcot: Cognitive hijacking for stealthy reasoning backdoors in llms,” *arXiv preprint arXiv:2504.05605*, 2025.
 - [61] Z. Xiang, F. Jiang, Z. Xiong, B. Ramasubramanian, R. Poovendran, and B. Li, “Badchain: Backdoor chain-of-thought prompting for large language models,” *arXiv preprint arXiv:2401.12242*, 2024.
 - [62] M. Kuo, J. Zhang, A. Ding, Q. Wang, L. DiValentin, Y. Bao, W. Wei, H. Li, and Y. Chen, “H-cot: Hijacking the chain-of-thought safety reasoning mechanism to jailbreak large reasoning models, including openai o1/o3, deepseek-r1, and gemini 2.0 flash thinking,” *arXiv preprint arXiv:2502.12893*, 2025.
 - [63] M. Zhang, Y. Zhang, J. Jia, Z. Wang, S. Liu, and T. Chen, “One token embedding is enough to deadlock your large reasoning model,” *arXiv preprint arXiv:2510.15965*, 2025.
 - [64] Y. Li, J. Wang, H. Zhu, J. Lin, S. Chang, and M. Guo, “Thinktrap: Denial-of-service attacks against black-box llm services via infinite thinking,” *arXiv preprint arXiv:2512.07086*, 2025.
 - [65] T. Sternak, D. Runje, D. Granoša, and C. Wang, “Automating prompt leakage attacks on large language models using agentic approach,” in *2025 MIPRO 48th ICT and Electronics Convention*. IEEE, 2025, pp. 99–104.
 - [66] S. Dong, S. Xu, P. He, Y. Li, J. Tang, T. Liu, H. Liu, and Z. Xiang, “Memory injection attacks on llm agents via query-only interaction,” *arXiv preprint arXiv:2503.03704*, 2025.
 - [67] S. S. Srivastava and H. He, “MemoryGraft: Persistent compromise of LLM agents via poisoned experience retrieval,” *arXiv preprint arXiv:2512.16962*, 2025.
 - [68] H. Liu, D. Xu, Q. Ma, S. Xu, and D. Qiu, “Memory poisoning propagation and repair mechanism in multi-agent collaborative environments,” in *Proceedings of the 2nd International Conference on Artificial Intelligence, Digital Media Technology and Social Computing*. ACM, 2026, pp. 225–231.
 - [69] V. Torra and M. Bras-Amorós, “Memory poisoning and secure multi-agent systems,” *arXiv preprint arXiv:2603.20357*, 2026.
 - [70] B. Wang, W. He, S. Zeng, Z. Xiang, Y. Xing, J. Tang, and P. He, “Unveiling privacy risks in llm agent memory,” in *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2025, pp. 25 241–25 260.
 - [71] X. Lyu, J. He, N. Wang, Y. Hu, T. Li, D. Chen, S. Li, and Y. Chen, “Adam: A systematic data extraction attack on agent memory via adaptive querying,” *arXiv preprint arXiv:2604.09747*, 2026.
 - [72] J. Liu, D. Cao, Y. Wei, T. Su, Y. Liang, Y. Dong, Y. Liu, Y. Zhao, and X. Hu, “Topology matters: Measuring memory leakage in multi-agent LLMs,” in *Findings*

- of the Association for Computational Linguistics: ACL 2026. Association for Computational Linguistics, 2026, pp. 39 728–39 746. [Online]. Available: <https://aclanthology.org/2026.findings-acl.1980/>
- [73] Z. Qi, H. Zhang, E. Xing, S. Kakade, and H. Lakkaraju, “Follow my instruction and spill the beans: Scalable data extraction from retrieval-augmented generation systems,” *arXiv preprint arXiv:2402.17840*, 2024.
- [74] C. Jiang, X. Pan, G. Hong, C. Bao, and M. Yang, “Rag-thief: Scalable extraction of private data from retrieval-augmented generation applications with agent-based attacks,” *arXiv preprint arXiv:2411.14110*, vol. 4, 2024.
- [75] Y. Liu, Y. Xie, M. Luo, Z. Liu, Z. Zhang, K. Zhang, Z. Li, P. Chen, S. Wang, and D. She, “Exploit tool invocation prompt for tool behavior hijacking in llm-based agentic system,” *arXiv e-prints*, pp. arXiv–2509, 2025.
- [76] Z. Wang, Y. Gao, Y. Wang, S. Liu, H. Sun, H. Cheng, G. Shi, H. Du, and X. Li, “MCPTox: A benchmark for tool poisoning on real-world MCP servers,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 40, no. 42, 2026, pp. 35 811–35 819. [Online]. Available: <https://ojs.aaai.org/index.php/AAAI/article/view/40895>
- [77] S. Jamshidi, K. W. Nafi, A. M. Dakhel, N. Shahabi, F. Khomh, and N. Ezzati-Jivan, “Securing the model context protocol: Defending llms against tool poisoning and adversarial attacks,” *arXiv preprint arXiv:2512.06556*, 2025.
- [78] D. Zhang, Z. Li, X. Luo, X. Liu, P. Li, and W. Xu, “MCP security bench (MSB): Benchmarking attacks against model context protocol in LLM agents,” in *The Fourteenth International Conference on Learning Representations*, 2026. [Online]. Available: <https://openreview.net/forum?id=irxxkFMrry>
- [79] M. Bhatt, V. S. Narajala, and I. Habler, “Etdi: Mitigating tool squatting and rug pull attacks in model context protocol (mcp) by using oauth-enhanced tool definitions and policy-based access control,” in *2025 Cyber Awareness and Research Symposium (CARS)*. IEEE, 2025, pp. 1–6.
- [80] N. Acharya and G. K. Gupta, “A formal security framework for mcp-based ai agents: Threat taxonomy, verification models, and defense mechanisms,” *arXiv preprint arXiv:2604.05969*, 2026.
- [81] W. Zhao, J. Liu, B. Ruan, S. Li, and Z. Liang, “When mcp servers attack: Taxonomy, feasibility, and mitigation,” *arXiv preprint arXiv:2509.24272*, 2025.
- [82] S. Raza, R. Sapkota, M. Karkee, and C. Emmanouilidis, “Trism for agentic ai: A review of trust, risk, and security management in llm-based agentic multi-agent systems,” *arXiv preprint arXiv:2506.04133*, 2025.
- [83] A. Dehghantanha and S. Homayoun, “Sok: The attack surface of agentic ai–tools, and autonomy,” *arXiv preprint arXiv:2603.22928*, 2026.
- [84] Z. Ji, D. Wu, W. Jiang, P. Ma, Z. Li, Y. Gao, S. Wang, and Y. Li, “Taming various privilege escalation in llm-based agent systems: A mandatory access control framework,” *arXiv preprint arXiv:2601.11893*, 2026.
- [85] S. Mitra, R. Patel, S. Mittal, M. R. Rahman, and S. Rahimi, “Agenticcyops: Securing multi-agentic ai integration in enterprise cyber operations,” *arXiv preprint arXiv:2603.09134*, 2026.
- [86] V. K. Nguyen and M. I. Husain, “Penetration testing of agentic ai: A comparative security analysis across models and frameworks,” *arXiv preprint arXiv:2512.14860*, 2025.
- [87] S. J. Lazer, K. Aryal, M. Gupta, and E. Bertino, “A survey of agentic ai and cybersecurity: Challenges, opportunities and use-case prototypes,” *arXiv preprint arXiv:2601.05293*, 2026.
- [88] M. Nakamura, A. Kumar, S. Das, S. Abdelnabi, S. Mahmud, F. Fioretto, S. Zilberstein, and E. Bagdasarian, “Colosseum: Auditing collusion in cooperative multi-agent systems,” *arXiv preprint arXiv:2602.15198*, 2026.
- [89] H. Lee, V.-D. Yun, D. Panagou, and S. P. Karimireddy, “Robust multi-agent LLMs under byzantine faults,” *arXiv preprint arXiv:2605.09076*, 2026.
- [90] A. El Mir, M. Takáč, and S. Lahlou, “Byzantine cheap talk: Adversarial resilience and topology effects in LLM coordination games,” *arXiv preprint arXiv:2606.07790*, 2026.
- [91] J. Hu, X. Huang, Y. Sun, Y. Dong, and X. Huang, “Lying with truths: Open-channel multi-agent collusion for belief manipulation via generative montage,” in *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. San Diego, California, United States: Association for Computational Linguistics, 2026, pp. 5979–5996.
- [92] X. Zeng and F. Rudzicz, “Voluntary collusion with secret tools in competing LLM agents,” *arXiv preprint arXiv:2605.27593*, 2026.
- [93] S. Yin, X. Pang, Y. Ding, M. Chen, Y. Bi, Y. Xiong, W. Huang, Z. Xiang, J. Shao, and S. Chen, “SafeAgent-Bench: A benchmark for safe task planning of embodied LLM agents,” *arXiv preprint arXiv:2412.13178*, 2024.
- [94] N. Sivaroopan, K. Thilakarathna, A. Zomaya, Y. Guo, J. Plested, T. Lynar, J. Yang, W. Yang *et al.*, “Shield: An auto-healing agentic defense framework for llm resource exhaustion attacks,” *arXiv preprint arXiv:2601.19174*, 2026.
- [95] Z. Anbiaee, M. Rabbani, M. Mirani, G. Piya, I. Opushnyev, A. Ghorbani, and S. Dadkhah, “Security threat modeling for emerging ai-agent protocols: A comparative analysis of mcp, a2a, agora, and anp,” *arXiv preprint arXiv:2602.11327*, 2026.
- [96] Y. Qu, Y. Liu, T. Geng, G. Deng, Y. Li, L. Y. Zhang, Y. Zhang, and L. Ma, “Supply-chain poisoning attacks against llm coding agent skill ecosystems,” *arXiv preprint arXiv:2604.03081*, 2026.
- [97] X. Jiang, S. Yang, W. Yang, Y. Liu, and C. Ji, “Agentic ai as a cybersecurity attack surface: Threats, exploits, and defenses in runtime supply chains,” *arXiv preprint arXiv:2602.19555*, 2026.
- [98] A. Abdennebi, N. Kara, L. Lahlou, and H. Ould-

- Slimane, “Lang—a governance-aware agentic ai platform for unified security operations,” *arXiv preprint arXiv:2604.05440*, 2026.
- [99] X. Gu, X. Zheng, T. Pang, C. Du, Q. Liu, Y. Wang, J. Jiang, and M. Lin, “Agent smith: A single image can jailbreak one million multimodal llm agents exponentially fast,” *arXiv preprint arXiv:2402.08567*, 2024.
 - [100] A. Amayuelas, X. Yang, A. Antoniadou, W. Hua, L. Pan, and W. Y. Wang, “Multiagent collaboration attack: Investigating adversarial attacks in large language model collaborations via debate,” in *Findings of the Association for Computational Linguistics: EMNLP 2024*, 2024, pp. 6929–6948.
 - [101] W. Yang, X. Bi, Y. Lin, S. Chen, J. Zhou, and X. Sun, “Watch out for your agents! investigating backdoor threats to llm-based agents,” *Advances in Neural Information Processing Systems*, vol. 37, pp. 100 938–100 964, 2024.
 - [102] Y. Wang, D. Xue, S. Zhang, and S. Qian, “Badagent: Inserting and activating backdoor attacks in llm agents,” in *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2024, pp. 9811–9827.
 - [103] J.-T. Huang, J. Zhou, T. Jin, X. Zhou, Z. Chen, W. Wang, Y. Yuan, M. Lyu, and M. Sap, “On the resilience of LLM-based multi-agent collaboration with faulty agents,” in *Proceedings of the 42nd International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, vol. 267. PMLR, 2025, pp. 26 202–26 226. [Online]. Available: <https://proceedings.mlr.press/v267/huang25ay.html>
 - [104] J. Fang, Z. Yao, R. Wang, H. Ma, X. Wang, and T.-S. Chua, “We should identify and mitigate third-party safety risks in mcp-powered agent systems,” *arXiv preprint arXiv:2506.13666*, 2025.
 - [105] T. Gasmi, R. Guesmi, I. Belhadj, and J. Bennaceur, “Bridging ai and software security: A comparative vulnerability assessment of llm agent deployment paradigms,” *arXiv preprint arXiv:2507.06323*, 2025.
 - [106] A. Zou, M. Lin, E. Jones, M. Nowak, M. Dziemian, N. Winter, V. Nathanael, A. Croft, X. Davies, J. Patel, R. Kirk, Y. Gal, D. Hendrycks, J. Z. Kolter, and M. Fredrikson, “Security challenges in AI agent deployment: Insights from a large scale public competition,” in *Advances in Neural Information Processing Systems*, vol. 38, 2025.
 - [107] H. Atta, K. Huang, K. R. Lambros, Y. Mehmood, Z. Baig, M. A. Rahman, M. Bhatt, M. Haq, M. Aatif, N. Shahzad *et al.*, “Laaf: Logic-layer automated attack framework a systematic red-teaming methodology for llm vulnerabilities in agentic large language model systems,” *arXiv preprint arXiv:2603.17239*, 2026.
 - [108] I. Kavathekar, H. Jain, A. Rathod, P. Kumaraguru, and T. Ganu, “TAMAS: Benchmarking adversarial risks in multi-agent LLM systems,” in *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, 2026, pp. 31 238–31 268.