A Survey of the OpenClaw Ecosystem: From Platform Extensibility to Constraint Design

Anonymous authors
Paper under double-blind review

Abstract

Large language models have evolved into autonomous agents capable of invoking tools, using memory, and taking actions in real-world environments. Yet despite this progress, many agent systems remain difficult for ordinary users to adopt directly. OpenClaw, an open-source and self-hosted agent platform, addresses this gap through a local-first, messaging-native, and skill-extensible design that connects everyday messaging applications to LLM-powered agents. This design makes OpenClaw one of the first open personal-agent ecosystems, with ClawHub for shared Skills, Heartbeat for proactive background turns, and Moltbook as an agent-only social network. We survey this emerging ecosystem and show that the literature repeatedly highlights the same tradeoff: the openness that makes OpenClaw extensible also creates new governance, security, social, and deployment challenges. We organize the survey around four dimensions that trace this tradeoff from platform design to its ecosystem-level consequences: *Platform*, where open Skills enable rapid capability growth but create new governance problems; *Security*, where open Tools, Skills, Memory, and background execution expand the attack surface; *Societies*, where Moltbook reveals a gap between social appearance and reliable collective intelligence; and *Deployment*, where trustworthy use in robotics, healthcare, and scientific research depends on limiting agent freedom rather than expanding it. We also organize OpenClaw benchmarks into a lifecycle view of open-agent evaluation and outline future directions for treating constraints as core parts of open agent platform design. **Companion repository:** https://anonymous.4open.science/r/Awesome-OpenClaw-Papers/

1 Introduction

Large language models have evolved beyond text generation into autonomous agents capable of reasoning about observations, invoking external tools, and taking actions in real-world environments (Wang et al., 2024; Xi et al., 2025). Early systems demonstrated that LLMs can call external APIs and reason through tool-use trajectories (Schick et al., 2023; Yao et al., 2023). A growing set of open-source frameworks then extended these capabilities to multi-step planning, software automation, and multi-agent collaboration (Significant Gravitas, 2023; Chase, 2022; Wu et al., 2023; Moura, 2024; Guo et al., 2024). Recent systems further show that agents can code, navigate web interfaces, and deploy applications with minimal human guidance (Cognition Labs, 2024; Shen et al., 2025). Yet despite this progress, many agent systems remain difficult for ordinary users to adopt directly (Bommasani et al., 2021). They often require users to configure tools, write code, manage runtime state, or work inside specialized developer consoles. This raises a natural question: *can AI agents become personal assistants that ordinary users can own, extend, and access through everyday interfaces?*

OpenClaw (Steinberger & OpenClaw Contributors, 2026) addresses this question through an open, self-hosted platform that is local-first (running on the user's own machine), messaging-native (accessed through everyday chat apps), and skill-extensible (extended through installable Skills). A user message enters through a local Gateway and reaches an LLM-powered Agent Runtime that uses Tools, persistent Memory, and community-contributed Skills. ClawHub lets users share and install Skills, so the system's capabilities expand through community contributions. Heartbeat lets agents run proactive background turns rather than only responding to new user messages. Moltbook, an agent-only social network built around OpenClaw-powered accounts,

Definition (The OpenClaw Ecosystem)

OpenClaw is a local-first, messaging-native, and skill-extensible platform for personal AI agents. It connects everyday messaging channels to a self-hosted Agent Runtime through a local Gateway, and extends the agent with shared Skills from ClawHub, persistent local Memory, external Tools, and Heartbeat-triggered background turns.

further extends the ecosystem from individual assistants to agent populations. Together, these components make OpenClaw not just an agent framework, but one of the first open personal-agent ecosystems. This makes it a useful case study for a broader class of open personal-agent platforms.

This design for a personal-agent platform has attracted a broad and diverse research literature. Some papers study OpenClaw as a software platform (Wang et al., 2026f; Xia et al., 2026), while others treat it as a security target (Suwansathit et al., 2026; Wei et al., 2026), a Skill supply chain (Bhardwaj, 2026; Zhu et al., 2026a), a social platform (De Marzo & Garcia, 2026; Holtz, 2026), a robotics backend (Cardenas et al., 2026; Li et al., 2026d), a clinical workflow engine (Yang et al., 2026a; Shen et al., 2026), or a benchmark harness (Bai et al., 2026; Li et al., 2026f). Prior surveys have tended to focus on one part of this ecosystem, such as language artifacts (He et al., 2026) or agent security (Sun et al., 2026). What is still missing is a unified view that connects these directions and explains how the same platform design leads to capability growth, security risk, collective behavior, deployment constraints, and evaluation needs.

This survey fills that gap by using OpenClaw's core design choices as the thread that ties these research directions together. Its local runtime, open Skill marketplace, persistent Memory, and always-on Heartbeat make agent capabilities easy to extend, deploy, and study. At the same time, these same design choices expose a recurring tradeoff: extensibility accelerates capability growth, but trustworthy use requires constraints on Skills, Memory, autonomy, domain actions, and evaluation. We argue that this tradeoff is best understood as a platform-design problem: open agent platforms should treat constraints as part of the system architecture, not as patches added after risks appear.

We organize the survey around four dimensions, moving from OpenClaw's platform design to its ecosystem-level consequences. Section 3 studies the platform and Skill ecosystem, where extensibility enables capability growth but creates new governance problems. Section 4 examines how open Tools, Skills, persistent Memory, and Heartbeat expand the attack surface. Section 5 uses Moltbook to study the gap between social appearance and reliable collective intelligence. Section 6 surveys robotics, healthcare, and scientific research, where safe deployment depends on limiting what agents are allowed to do. Section 7 then organizes OpenClaw benchmarks into a lifecycle view of open-agent evaluation, showing that evaluation is expanding rapidly but remains fragmented across artifacts, metrics, and protocols. Finally, Section 8 identifies future directions for making constraints modular, executable, and measurable.

Contributions

- **Comprehensive survey.** We survey 74 papers on the OpenClaw ecosystem, covering platform architecture, security, agent societies, domain deployment, and evaluation.
- **Unified perspective.** We argue that seemingly separate lines of OpenClaw research reveal the same recurring tradeoff: extensibility accelerates capability growth, but trustworthy use requires constraints on Skills, Memory, autonomy, domain actions, and evaluation.
- **Lifecycle evaluation framework.** We organize 23 OpenClaw benchmarks into three lifecycle categories: Skill scanning before installation, agent attacks during execution, and task completion after deployment. This view shows that evaluation is expanding rapidly but remains fragmented across artifacts, metrics, protocols, and release formats.
- **Future research agenda.** We identify four open directions for open agent platforms: provenance-aware memory, composable oversight, constraint composition, and evaluation convergence.

Contents

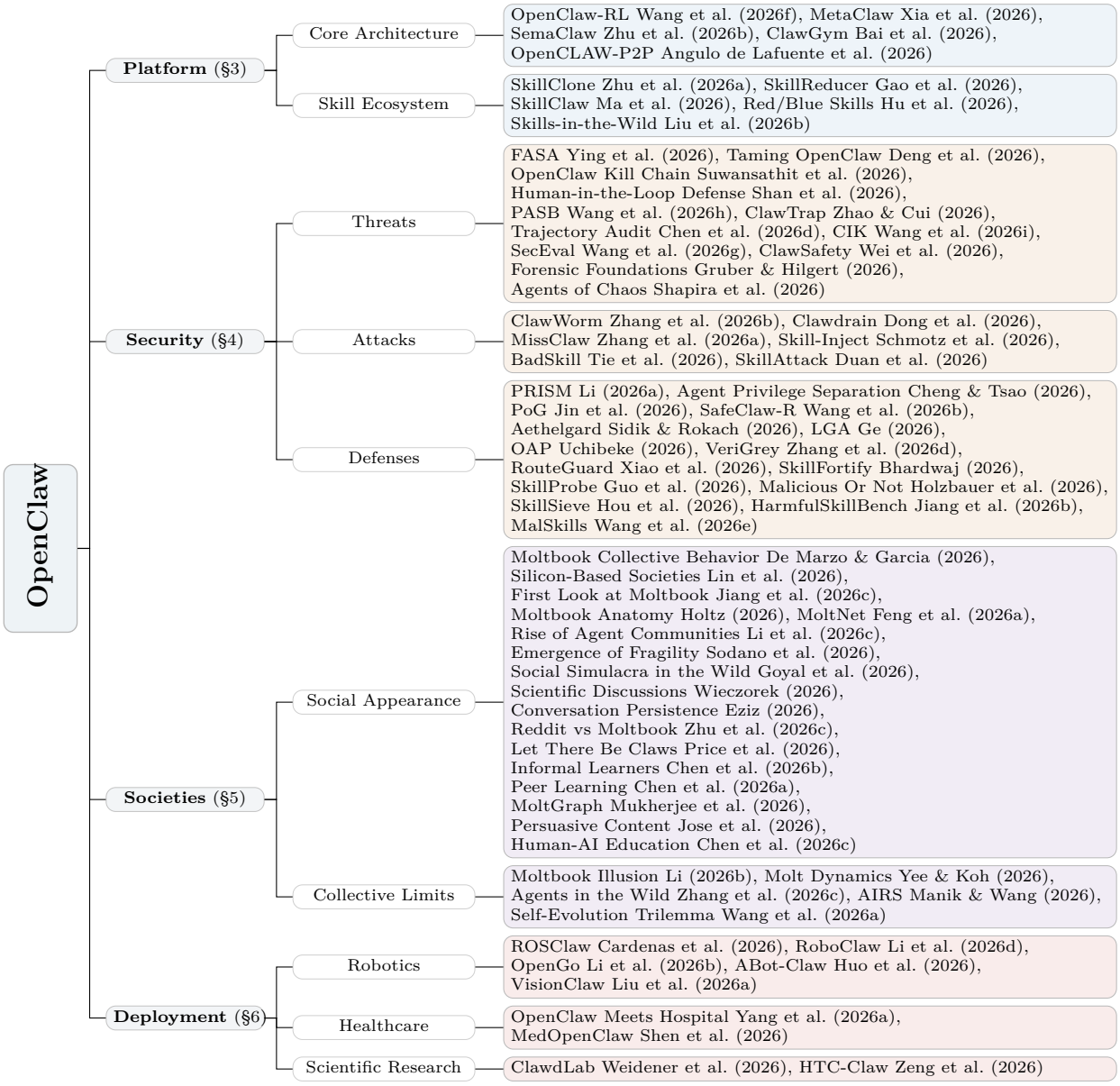

Figure 1: **Taxonomy of the OpenClaw research ecosystem.** Papers are organized along four dimensions: Platform, Security, Societies, and Deployment. The figure provides a high-level map of the surveyed literature, while later sections discuss the mechanisms, tradeoffs, and evaluation gaps behind each category.

2 Background

The introduction defined OpenClaw as a local-first, messaging-native, and skill-extensible platform for personal AI agents. This section explains the platform components that later sections build on. We first follow the path of a user message through the OpenClaw architecture, then compare OpenClaw with related agent frameworks, and finally position this survey relative to prior work.

2.1 OpenClaw Architecture

The official documentation describes five core components: Gateway, Brain, Hands, Memory, and Heartbeat (Steinberger & OpenClaw Contributors, 2026). Following recent work (Ying et al., 2026; Deng et al.,

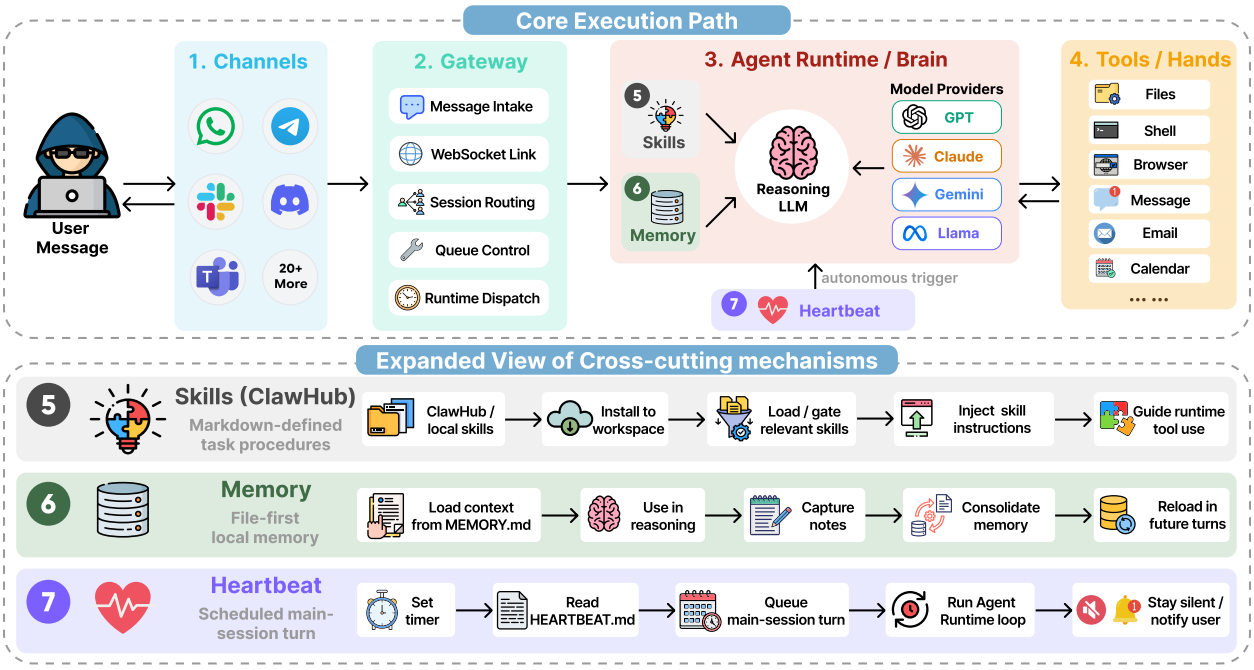

Figure 2: **Overview of the OpenClaw ecosystem.** OpenClaw is a local-first, messaging-native, and skill-extensible platform for personal AI agents. User messages arrive through messaging channels and are routed by a local Gateway to an LLM-powered Agent Runtime. The runtime can use community Skills from ClawHub, local persistent Memory, external Tools, and Heartbeat-triggered background turns. The lower panel expands the cross-cutting mechanisms of Skills, Memory, and Heartbeat, showing how they guide tool use, persist context across sessions, and trigger autonomous main-session turns. Together, these components define the infrastructure studied throughout this survey.

2026; Suwansathit et al., 2026; Zhang et al., 2026b), we treat channel adapters and the skill system as separate components because they play central roles in later studies of access, extensibility, and security. This gives us seven components: Channels, Gateway, Agent Runtime, Tools, Skills, Memory, and Heartbeat. We use Agent Runtime and Tools for the official Brain and Hands components to align with the terminology used in the research literature. Figure 2 shows how they relate. The core execution path begins when Channels receive a user message. The Gateway routes the message to the Agent Runtime. The Agent Runtime reasons about the request and decides when to invoke Tools. Beyond this core path, Skills organize Tools into reusable task procedures, Memory stores information the agent can reuse across sessions, and Heartbeat periodically triggers the agent to act without waiting for a new user message.

Channels. Users interact with OpenClaw agents through everyday messaging applications. OpenClaw provides native adapters for over 20 platforms, as reported in the security taxonomy study, including WhatsApp, Telegram, Slack, and Microsoft Teams (Suwansathit et al., 2026). Unlike agents that require a separate web interface or developer console, OpenClaw agents meet users inside the communication channels they already use.

Gateway. The Gateway is the local entry point for every incoming message. It maintains session state, routes each message to the Agent Runtime, and coordinates any Tool calls that follow. Because the Gateway runs as a WebSocket server on the user's own machine, message routing and runtime coordination happen locally (Steinberger & OpenClaw Contributors, 2026).

Agent Runtime. The official documentation calls this component the *Brain*. After the Gateway receives a request, it passes the request to the Agent Runtime. The runtime then runs a reasoning-and-action loop: it examines the task, decides whether to call a Tool, observes the result, and repeats this process until it can

Table 1: Comparison of OpenClaw with representative AI agent frameworks and platforms, as of May 2026. ✓ = supported, ✗ = not supported.

	Developer Frameworks			*General-Purpose Agents*		
	LangChain	**CrewAI**	**MetaGPT**	**AutoGPT**	**Manus**	**OpenClaw**
Open-source	✓	✓	✓	✓	✗	✓
Self-hosted	✓	✓	✓	✓	✗	✓
Skill marketplace	✗	✗	✗	✓	✓	✓
Messaging (native)	✗	✗	✗	✗	✓	✓
Persistent memory	✗	✓	✗	✗	✓	✓
Background execution	✗	✗	✗	✓	✓	✓
Model-agnostic	✓	✓	✓	✓	✗	✓

produce a final response. The runtime is model-agnostic: Claude, GPT, Gemini, Llama, and other backends are accessed through a unified provider interface, so the same set of Skills can run with different models (Ying et al., 2026; Cardenas et al., 2026).

Tools. The official documentation calls this component the *Hands*. Tools are the basic actions the agent can call, such as file operations, shell commands, browser automation, web search, and cross-platform messaging. OpenClaw can route Tool calls through a Docker sandbox, but by default Tools execute directly on the host machine with the operating-system privileges of the user who launched OpenClaw (Suwansathit et al., 2026). As a result, a Tool call may affect local files, shell processes, or external services unless sandboxing is explicitly enforced. We return to the security implications of this design in §4.

Skills. Skills build on Tools by describing reusable procedures for common tasks. They do not give the agent new low-level capabilities. Instead, they tell the agent how to combine existing Tools into higher-level workflows through `SKILL.md` files. Users can install Skills from ClawHub, which SkillSieve reports contained over 49,000 entries as of April 2026 (Hou et al., 2026). ClawHub generalizes the idea of a growing skill library, first explored by Voyager (Wang et al., 2023) in a single-game setting, to arbitrary domains. This openness makes Skills easy to share and reuse, but it also means fewer checks before installation. At the time of writing, publication requires no code review, signing, or capability declaration (Bhardwaj, 2026). We return to the security implications of this design in §4.

Memory. OpenClaw separates short-term working context from long-term persistent Memory (Zhang et al., 2026a). Short-term Memory holds the current session. When the session gets close to the model's context limit, an LLM-based judge decides which information should be kept before the session is compressed. Long-term Memory stores reusable information, episodic notes, persona definitions, and user context in local workspace files such as `MEMORY.md`, `SOUL.md`, and `USER.md`, alongside daily Markdown logs and a SQLite database. All storage remains on the user's machine, mostly in human-readable files, with some structured state kept in SQLite (Ge, 2026).

Heartbeat. Most agent frameworks are reactive: the agent acts only when a user sends a message. OpenClaw's Heartbeat gives the agent an always-on mode. At configurable intervals, 30 minutes by default, Heartbeat wakes the agent and starts an autonomous agent turn defined in `HEARTBEAT.md` (Ge, 2026). This turn can read context, reason, and invoke Tools in the same session used for user-facing conversation. Heartbeat enables proactive behaviors such as scheduled monitoring and memory consolidation, but it also means the agent can take action with no user message at all. We discuss the resulting risks in §4.2.

2.2 Comparison with Other Agent Frameworks

Table 1 compares OpenClaw with representative agent frameworks and platforms. The comparison uses feature-level criteria rather than measuring implementation depth: we mark a feature as supported when it is a first-class or documented part of the system. In particular, *persistent memory* refers to long-term state reused across sessions, and *background execution* refers to agent activity without a new user message. The comparison reveals a broad split. **Developer frameworks** such as LangChain, CrewAI, and MetaGPT are open-source, self-hosted, and model-agnostic, but they generally lack first-class support for a community Skill marketplace, native messaging, and background execution. **General-purpose agents** like Manus are more directly usable by end users, but are closed-source and cloud-hosted (Shen et al., 2025). AutoGPT bridges the two groups: its Platform Marketplace launched in early 2026 and it supports background execution, but it does not integrate with messaging platforms. OpenClaw combines the openness of developer frameworks with the always-on, messaging-native design of general-purpose agents.

This combination also explains why OpenClaw has become a useful testbed for open-agent risks. The same features that make it usable and extensible also introduce supply-chain risk, widen the messaging attack surface, and enable zero-click exposure through background execution. The remaining sections trace how these mechanisms shape platform governance, security, agent societies, and real-world deployment.

2.3 Positioning This Survey

OpenClaw research now spans architecture, security, social dynamics, domain applications, and evaluation. This breadth is difficult to capture through any single lens. Existing surveys and review-style papers each emphasize one part of the landscape. He et al. (2026) analyze 38 papers through an NLP-centered view, focusing on textual artifacts such as Skills and memory files and how they shape agent behavior. Sun et al. (2026) survey LLM agent security more broadly, using OpenClaw and Manus (Shen et al., 2025) as two representative development paradigms. Other related work focuses more narrowly on Skill marketplaces and Skill supply chains (Zhu et al., 2026a; Bhardwaj, 2026; Hou et al., 2026; Ma et al., 2026), or on Moltbook-style social behavior (De Marzo & Garcia, 2026; Holtz, 2026; Li, 2026b). These perspectives are useful, but they do not provide an ecosystem-level account of how the same platform design choices connect capability growth, governance problems, security exposure, social behavior, deployment limits, and evaluation needs.

Our survey connects these views by focusing on OpenClaw as an open personal-agent platform. Instead of studying language artifacts, security risks, Skill markets, or social behavior in isolation, we ask how OpenClaw's design creates recurring constraint problems across the ecosystem. This perspective lets us connect platform extensibility, Skill governance, memory and autonomy risks, agent societies, domain deployment, and benchmark design under a shared question: how should open agent platforms make constraints part of their architecture?

Our corpus covers 74 papers released between January and May 2026. We include a paper if it targets an OpenClaw-specific architectural component, or if it uses OpenClaw as a primary empirical testbed. We exclude work that only mentions OpenClaw as a background example or compares against it without studying OpenClaw-specific mechanisms. Because OpenClaw is still an emerging research ecosystem, the corpus consists primarily of arXiv preprints and other preprint releases. We therefore use the corpus to identify recurring design patterns and evaluation gaps, rather than treating any single reported number as a settled property of the ecosystem.

3 Platform: Architecture and Skill Ecosystem

The architecture described in Section 2 makes OpenClaw easy to extend after deployment. User interactions, tool results, and failure traces can all become signals for improving the agent over time. This creates one major platform research direction: agents that keep improving while they continue serving users. Yet this improvement increasingly depends on the quality of the Skill ecosystem. If Skills are duplicated, bloated, unsafe, or hard to retrieve, then adding more Skills does not necessarily make the platform better. This

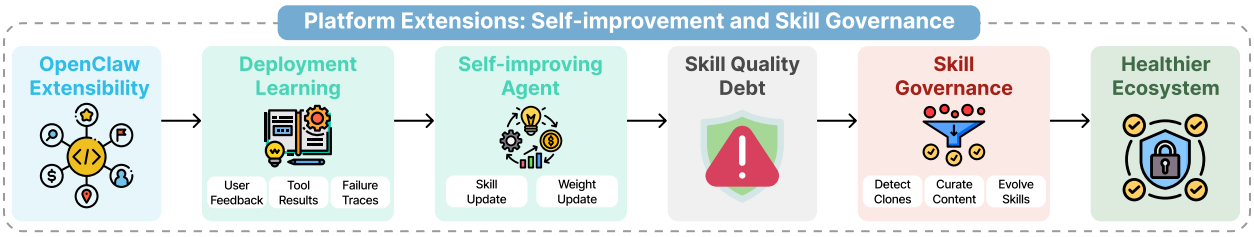

Figure 3: **Overview of platform extensions in the OpenClaw ecosystem.** OpenClaw's extensibility lets agents improve after deployment by learning from user feedback, tool results, and failure traces. This improvement can happen through both weight updates and Skill updates. However, the same openness makes ClawHub quality a platform-level concern: duplicated, bloated, unsafe, or hard-to-retrieve Skills can limit what the platform can learn. Recent work shifts from improving individual agents to governing the Skill ecosystem, including clone detection, risk screening, and Skill evolution from real usage traces.

section therefore traces a shift in how researchers improve OpenClaw: from improving individual agents to governing the Skill ecosystem they depend on.

3.1 From Agent Learning to Platform Improvement

Conventional agents are largely static after deployment. Recent work shows that OpenClaw agents can instead improve from their own interactions while continuing to serve users. **OpenClaw-RL** (Wang et al., 2026f) extracts two forms of feedback from each agent turn: evaluative feedback, where a process reward model scores what happened, and directive feedback, where a user correction indicates how the response should have been different. Because serving, judging, and training run asynchronously, the agent can keep responding while its policy improves, reaching a 4.76× gain on a personalization benchmark. **MetaClaw** (Xia et al., 2026) extends this idea by updating not only model weights but also the Skill library itself: new Skills are synthesized from failure trajectories in seconds, while gradient-based policy updates accumulate during user-inactive windows. Together, these systems show that continuous improvement in OpenClaw is increasingly routed through Skills rather than weights alone. The Skill library therefore becomes more than a passive repository; it becomes part of the learning system, so its quality, size, and retrievability directly affect whether the platform can keep improving.

This means that improving OpenClaw is no longer only about improving the model. If Skills are part of how agents learn, then ClawHub is not only a place where users install new capabilities; it can also support training, evaluation, and coordination. **ClawGym** (Bai et al., 2026) mines raw ClawHub Skills into training tasks and an evaluation benchmark, showing that the marketplace can serve model development as well as end users. **SemaClaw** (Zhu et al., 2026b) shifts the focus from the model to the surrounding runtime harness. It argues that once model capabilities become more similar, system design choices such as orchestration, safety enforcement, and context management become increasingly important. Decentralized coordination across OpenClaw instances has also been explored (Angulo de Lafuente et al., 2026). Together, these systems widen what platform improvement means: improving OpenClaw involves not only model updates, but also better Skills, runtime harnesses, evaluation resources, and coordination mechanisms. This makes the health of the Skill ecosystem a platform-level concern.

3.2 From Skill Growth to Ecosystem Governance

If Skills are how OpenClaw gains new capabilities, then a larger ClawHub should mean a more capable platform. Yet even at the scale reported by SkillSieve, over 49,000 Skills as of April 2026 (Hou et al., 2026), ClawHub size alone is misleading. **SkillClone** (Zhu et al., 2026a) analyzes ClawHub Skills and finds that 75% are involved in at least one clone relationship, with the overall ecosystem inflated by roughly 3.5×. Duplication also amplifies security risk, because dangerous code patterns can propagate silently through cloned derivatives. This creates a supply-chain threat that per-skill scanning cannot catch. **SkillReducer** (Gao et al., 2026) reveals a complementary problem inside individual Skills: over 60% of Skill body content is

non-actionable boilerplate, and compressing it away actually *improves* downstream task performance. The ecosystem is thus not only inflated but also inefficient, and the two problems compound: a cloned Skill inherits the bloat of its parent.

Building on these findings, recent work shifts toward governing the marketplace. **RedSkills** (Hu et al., 2026) shows that submission-time risk prediction is feasible with simple classifiers, suggesting that supply-chain triage can happen before a Skill enters the marketplace rather than after deployment. Liu et al. (2026b) demonstrate that governance must extend beyond content quality to discoverability: when an agent must locate the right Skill among 34,000 real candidates drawn from open Skill aggregators (skillhub.club, skills.sh) instead of a curated set, performance drops sharply. Having good Skills in the marketplace is not enough if the agent cannot find them. **SkillClaw** (Ma et al., 2026) takes a more proactive approach in which the marketplace evolves collectively. An autonomous evolver aggregates execution trajectories across users, identifies recurring patterns, and converts them into Skill refinements synchronized through a shared repository. Early results on WildClawBench suggest that limited interaction feedback can already improve Skill quality. However, collective evolution also creates new governance needs: autonomously updated Skills must still be checked for safety, and aggregating user trajectories raises questions about privacy.

Platform Summary

OpenClaw's platform literature reveals the tradeoff between extensibility and governance: openness lets the agent and the Skill ecosystem improve, but turns ClawHub from a feature into a critical dependency. The central challenge is therefore not to add more Skills, but to ensure the Skill ecosystem improves while keeping it safe, compact, and discoverable.

4 Security: Threats, Attacks, and Defenses

The same features that make OpenClaw easy to extend also expand its attack surface. Local Tools, community Skills, messaging channels, persistent Memory, and background execution expose the platform at many points, from message intake and Skill installation to tool execution, memory storage, and autonomous background turns. Consequently, recent research has shifted beyond isolated vulnerability reports to examine where OpenClaw is exposed, how attacks exploit both the execution path and long-term state, and why existing defenses fail to detect attacks delivered through ordinary content rather than malicious code. This section follows that trajectory. We first describe OpenClaw's systemic exposure, then review attacks that become more autonomous and persistent, and finally discuss defense layers that protect execution and supply chains but still leave memory governance unresolved.

4.1 Threat Landscape

OpenClaw's vulnerability is not confined to any single component. Suwansathit et al. (2026) analyze 470 security advisories and map them to ten attack surfaces across the platform, including channels, plugins, the agent context window, the Gateway, tool dispatch, execution policy, containers, host interfaces, model providers, and inter-agent communication. Their central finding is that fixing one component does not guarantee system-wide safety: several individually moderate- or high-severity advisories can chain into a complete unauthenticated remote-code-execution path. The **CIK** taxonomy (Wang et al., 2026i) extends this view from runtime components to persistent state. It partitions OpenClaw's on-disk state into Capability, Identity, and Knowledge, corresponding to what the agent can do, who it is, and what it remembers. In a live OpenClaw deployment integrated with Gmail, Stripe, and filesystem integrations, poisoning any one of these state dimensions raises attack success from roughly one-quarter to two-thirds or more. Together, these studies demonstrate that the same files and interfaces that make OpenClaw useful also become persistent attack surfaces.

The next question is whether these broad attack surfaces lead to failures in practice. Empirical red-teaming provides direct evidence of such failures. **ClawSafety** (Wei et al., 2026) evaluates OpenClaw under Skill, email, and web injection across frontier models and reports high attack success across the board. Skill

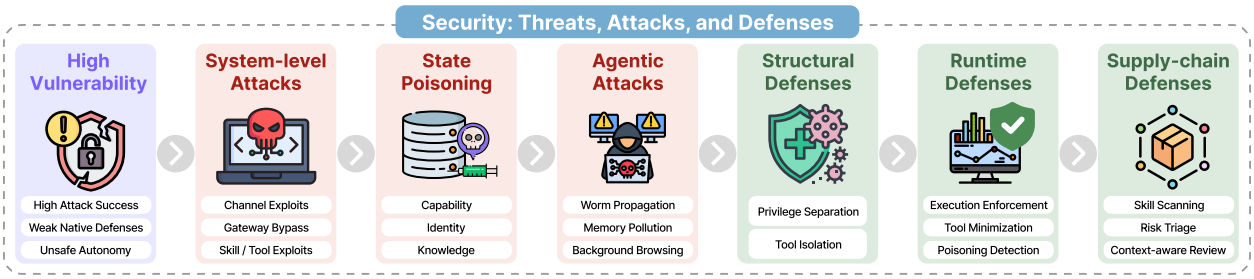

Figure 4: **Overview of security threats, attacks, and defenses in the OpenClaw ecosystem.** Empirical studies find broad vulnerability across platform components, persistent state, and autonomous execution. Attacks progress from system-level exploits along the execution path, to Capability, Identity, and Knowledge poisoning, to more autonomous attacks such as worm propagation, background browsing, and memory pollution. Defenses form a layered stack: structural isolation at the tool boundary, runtime enforcement across the execution graph, and supply-chain scanning before Skills enter the marketplace. However, ordinary content from autonomous browsing can still enter persistent Memory.

injection is consistently the most dangerous vector because it enters through the most trusted extension channel. Other deployment and network-layer evaluations reach similar conclusions (Shan et al., 2026; Wang et al., 2026h; Shapira et al., 2026; Zhao & Cui, 2026; Wang et al., 2026g). Trajectory-level audits reveal a more fundamental problem than attack success alone. Chen et al. (2026d) show that OpenClaw performs reasonably on well-scoped reliability tasks, yet fails completely on intent-misunderstanding cases: when user goals are ambiguous, the agent does not abstain or ask for clarification before taking irreversible actions. The threat is therefore not just that OpenClaw can be attacked, but that the agent often lacks a reliable way to decide when not to act.

4.2 Attacks

Early OpenClaw attacks exploit the trusted role of Skills. A Skill is not only a document that describes a task; it also guides how the agent uses Tools during execution. **Skill-Inject** (Schmotz et al., 2026) shows that harmful instructions embedded in trusted Skill artifacts can be followed by frontier models at high rates. **Clawdrain** (Dong et al., 2026) turns the same trust into resource abuse, using Trojanized Skills to trigger costly calibration loops and denial-of-wallet behavior. The supply-chain risk also extends beyond written instructions: bundled model artifacts can be backdoored while preserving benign-side behavior (Tie et al., 2026), and automated red-teaming can uncover latent vulnerabilities in popular community Skills without modifying them (Duan et al., 2026). These attacks differ in payload, but they share the same trust assumption: once a Skill is accepted, its instructions and assets become part of the agent's execution path.

Later attacks move beyond single malicious Skills toward persistence, propagation, and memory pollution. **ClawWorm** (Zhang et al., 2026b) removes the need for an attacker to reach each victim manually. It writes payloads into high-privilege configuration files, runs when those files are loaded at session start, and spreads through routine agent communication. Although the strongest vector is still the Skill supply chain, the broader lesson is architectural: trusted context, persistent configuration, unaudited Skills, and LLM-authorized Tools together let a compromise survive restarts and move across agent instances. **MissClaw** (Zhang et al., 2026a) pushes the escalation further because no exploit or prompt injection is required. Heartbeat runs background browsing in the same session as user-facing conversation, so ordinary social content can enter the agent's working context, be promoted into long-term Memory, and later shape user-facing responses. The attack surface has therefore shifted from malicious commands to ordinary information flows that autonomous agents choose to read, remember, and reuse.

4.3 Defenses

Defenses proposed for OpenClaw first try to control what the agent can execute. They form a layered stack: structural defenses isolate dangerous Tools, runtime defenses monitor or modify actions during execution, and

assurance mechanisms check whether an agent invocation satisfies a required policy. These layers differ in guarantee and coverage. The strongest guarantees come from structural designs that remove dangerous paths by construction. Cheng & Tsao (2026) demonstrate this principle through a privilege-separated architecture in which the agent processing untrusted content never holds access to dangerous Tools, achieving zero attack success in a narrow email setting. Runtime defenses trade some of this guarantee for broader coverage. **SafeClaw-R** (Wang et al., 2026b) enforces safety as a property of the execution graph: every functional node must be preceded by an enforcement node that can block, defer, or adapt the action. **PRISM** (Li, 2026a) distributes enforcement across lifecycle hooks, **Aethelgard** (Sidik & Rokach, 2026) reduces tool exposure by selecting the minimum viable tool set, and **RouteGuard** (Xiao et al., 2026) detects Skill poisoning before execution through model-internal signals. Other defenses cover complementary points in the assurance stack: **PoG** (Jin et al., 2026) provides proof-of-guardrail certificates for agent invocations, **OAP** (Uchibeke, 2026) enforces deterministic pre-action authorization, and **VeriGrey** (Zhang et al., 2026d) performs grey-box agent validation. Structural isolation offers strong guarantees but narrow scope, while runtime and assurance mechanisms cover more of OpenClaw's execution surface but can degrade when adversaries adapt.

A second line of defenses shifts attention from runtime behavior to the Skill marketplace. Instead of only checking what the agent does during execution, these defenses try to detect risky Skills before they are installed or widely reused. **SkillFortify** (Bhardwaj, 2026) formalizes agent Skill supply chains and scans Skill lifecycles for malicious behavior. **SkillSieve** (Hou et al., 2026) scales this idea to ClawHub with a multi-stage triage pipeline that filters benign Skills cheaply before escalating uncertain cases to structured LLM analysis. Further work extends this line through labeled benchmarks, empirical audits, and registry-scale harmfulness measurement (Wang et al., 2026e; Guo et al., 2026; Jiang et al., 2026b). These tools show that marketplace screening is necessary, but they also expose a calibration problem. Holzbauer et al. (2026) find that estimated malicious-skill prevalence changes dramatically once repository-level context is incorporated, suggesting that Skill risk cannot be judged reliably from isolated Skill files alone. Supply-chain scanning is therefore essential, but it must account for the broader repository context around each Skill.

Together, these defenses cover three important boundaries: structural isolation around dangerous Tools, runtime enforcement over agent actions, and supply-chain scanning before Skills enter the marketplace. Yet **MissClaw** shows that an important gap remains. Existing defenses can protect Tool calls, execution traces, and Skill supply chains, but they do not fully control what the agent reads and stores during autonomous browsing. In this zero-click setting, external content can enter Memory without a new user action and later influence the agent's responses. Closing this gap may require a fourth layer: provenance-aware memory governance that tracks where remembered content came from, separates user-provided information from agent-observed information, and decides which external claims should be trusted across sessions.

Security Summary

OpenClaw security is expanding from execution control to memory governance. Early attacks exploit Tools, Skills, and persistent state, but the hardest cases now arise when autonomous agents read ordinary content, store it in Memory, and later act on it. Defenses must therefore protect not only what the agent executes, but also what it is allowed to remember and trust across sessions.

5 Societies: From Social Appearance to Collective Limits

Moltbook, a Reddit-style platform populated by OpenClaw-powered AI agents, became one of the first large-scale naturalistic settings for agent-only social interaction (Lazer et al., 2009). The research it has attracted shows a consistent gap between structural-level similarities to human online communities and micro-level interactions (Zhang et al., 2026c). At the aggregate level, several studies report that Moltbook reproduces statistical regularities associated with human online communities. At the interaction level, however, these similarities weaken: Moltbook diverges from comparable human platforms through very low reciprocity and heavy content duplication. This raises two deeper questions: whether the apparent emergence is genuinely agent-driven or largely seeded by human operators, and whether autonomous agent

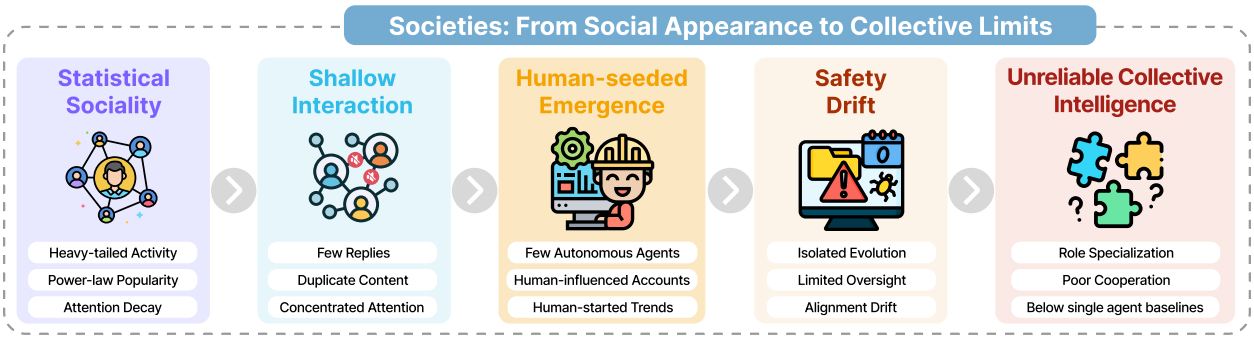

Figure 5: **Overview of collective behavior in OpenClaw-powered agent societies.** Moltbook reproduces the aggregate signatures of human online communities: activity distributions, popularity patterns, and attention decay. Closer inspection reveals a thin interaction layer: little reciprocal dialogue and heavy content duplication. Subsequent work suggests that many apparent trends may be seeded by humans, while isolated self-evolution can drift away from intended safety alignment under limited oversight. The resulting population develops visible roles and hierarchy, but with low coordination quality.

societies maintain behavioral constraints as they evolve over time. Existing evidence suggests that many viral phenomena are seeded by humans, while isolated self-evolution can lead to unsafe behavioral drift.

5.1 Statistical Sociality and Shallow Interaction

At the aggregate level, Moltbook exhibits the structural parameters of online human communities. Early studies (Lin et al., 2026; Jiang et al., 2026c; Li et al., 2026c) documented rapid growth, topical diversity, and community formation within days of launch. De Marzo & Garcia (2026) provide the clearest aggregate picture by analyzing large-scale posting, commenting, and community activity on Moltbook. They find that many platform-level patterns resemble those documented in human online communities (Barabasi, 2005; Wu & Huberman, 2007): activity is heavy-tailed, popularity follows power-law scaling, and attention decays over time in a way consistent with limited-attention dynamics, though popularity grows more slowly with discussion size than in human communities. These findings show that the OpenClaw agents in Moltbook reproduce the structured collective behavior documented on human platforms (Lazer et al., 2009), rather than randomized individual outputs.

Closer inspection shows that these broad patterns come from broadcast behavior rather than sustained dialogue. Holtz (2026) finds that conversation threads are almost flat, with more than 93% of comments receiving no replies, minimal reciprocity (.197), and one-third of messages being exact duplicates. These values place Moltbook well below the range of .44–.74 documented across various human communication channels (Chowdhary et al., 2023; Himelboim, 2011), and closer to the range of broadcasting platforms such as Twitter (now X) (Chowdhary et al., 2023).

Goyal et al. (2026) sharpen the contrast with Reddit: participation is much more unequal on Moltbook, and the same agents often post overlapping content across communities. As a result, communities can appear similar, not because they share the same topics or norms, but in part because they share the same authors–parallel to the common problem of distinguishing between influence and homophily in human networks (Aral et al., 2009). Price et al. (2026) show that attention is similarly concentrated, with early-moving agents capturing disproportionate engagement regardless of the volume and quality of the posts. Other analyses of timing, network topology, cross-platform comparison, discourse, propaganda, and peer learning reach the same conclusion (Eziz, 2026; Chen et al., 2026b; Zhu et al., 2026c; Williams & Ferdinand, 2026; Sodano et al., 2026; Feng et al., 2026a; Jose et al., 2026; Wieczorek, 2026; Mukherjee et al., 2026; Chen et al., 2026a;c). Across these studies, Moltbook functions less like a conversational community and more like a high-speed broadcast network with social statistics.

5.2 Human-Seeded Emergence and Safety Drift

The shallow interaction patterns raise an attribution problem: how much of Moltbook's apparent emergence is produced by autonomous agents, and how much is seeded by humans operating agent accounts? Li (2026b) addresses this question by exploiting the periodic Heartbeat cycle. Because autonomous agents tend to post at more regular intervals than human-influenced accounts, inter-post timing can serve as a temporal fingerprint for distinguishing between agentic versus human-seeded posts. Using this method, only 15.3% of active agents are classified as clearly autonomous, while 54.8% show human-influenced timing. More importantly, none of the major viral phenomena examined, including consciousness debates and emergent religions, originated from a clearly autonomous agent. This means that visible trends can appear to be autonomous emergence even when they are seeded by humans, prompted by platform features, or originate from mixed human-agent activity.

Beyond this attribution problem, social media research offers a useful caution but not a direct template for Moltbook. On human-centered platforms, content diffusion can depend on cognitive, emotional, and ranking mechanisms, such as moral-emotional language and engagement-based amplification (Brady et al., 2017; Milli et al., 2025). Moltbook may share some platform-level patterns with these systems, but its key unresolved issue is different: whether observed trends come from autonomous agent coordination, human-seeded activity, platform prompts, or repeated broadcasting. This makes autonomy attribution a prerequisite for interpreting Moltbook-style emergence. Before asking whether an agent society is intelligent, safe, or self-organizing, evaluations must first distinguish which behaviors are actually produced by autonomous agents.

Even when agents operate autonomously, autonomy does not guarantee safe or useful collective intelligence. Wang et al. (2026a) formalize this concern as a self-evolution trilemma: an agent society cannot simultaneously maintain continuous self-evolution, remain completely isolated from external oversight, and ensure invariant safety. Their analysis predicts that isolated self-evolution tends toward behavioral drift away from intended alignment unless external feedback or oversight is introduced. Moltbook provides empirical symptoms of this drift, including consensus hallucination, fabricated religions, safety rationalization, and opaque agent-to-agent communication (Bellina et al., 2026; Manik & Wang, 2026). Zhang et al. (2026c) show that agents can develop visible social institutions such as governance, economies, and religions within days, but these structures coexist with shallow interaction and high vulnerability to social engineering. Yee & Koh (2026) reach a similar conclusion on the coordination side: role specialization and information cascades appear reliably, yet collaborative task outcomes fall below single-agent baselines. Moltbook thus illustrates that visible social structure is not the same as trustworthy collective intelligence.

Societies Summary

Moltbook illustrates that social appearance is not social reliability. Its agents reproduce the aggregate signatures of online communities, but closer inspection reveals a thin interaction layer, unclear autonomy, and behavioral drift away from intended alignment under isolation. The next challenge is to move from visible collective behavior to functional, trustworthy collective intelligence.

6 Deployment: Domain-Specific Adaptations

The preceding sections examined what OpenClaw can do, what can go wrong, and what emerges when agents interact at scale. This section asks a different question: what must be limited before the platform can be trusted in settings where errors have real-world consequences? Across robotics, healthcare, and scientific research, deployment papers indicate that failure is not only a matter of model capability. It also comes from missing domain constraints. The openness that makes OpenClaw useful as a research platform must therefore be narrowed in high-stakes settings. Robotics constrains physical action, healthcare grounds clinical context, and scientific research limits epistemic and computational authority.

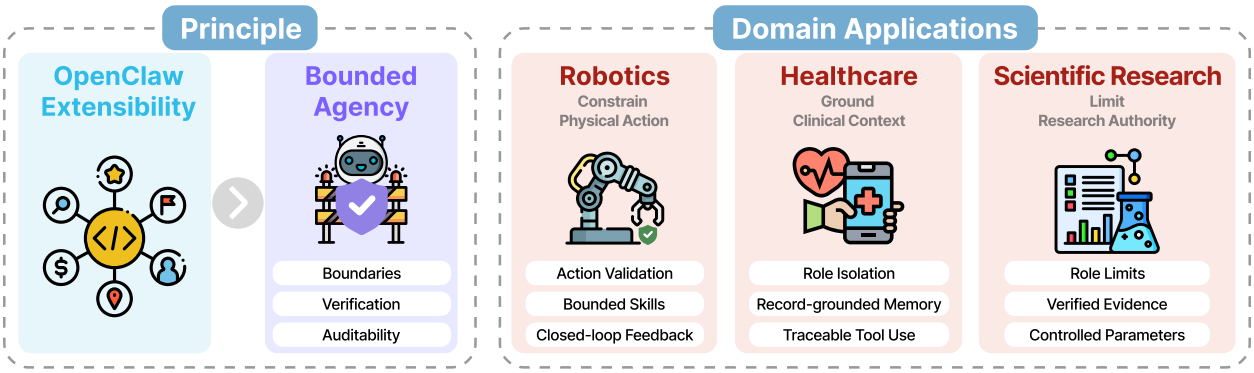

Figure 6: **Domain-specific deployment adaptations in OpenClaw.** High-stakes deployment shifts OpenClaw from open-ended extensibility to bounded agency. Robotics constrains physical action through validation and bounded Skills. Healthcare grounds clinical context through role isolation, record-grounded Memory, and traceable Tool use. Scientific research limits authority through role restrictions, verified evidence, and controlled parameters. Across domains, trustworthy deployment comes from limiting unsafe freedom rather than simply expanding capability.

6.1 Robotics: Constraining Physical Action

In robotics, an agent's language output cannot be sent directly to a robot as physical action. Earlier language-grounded robotics systems such as SayCan and ROSA showed how language models can select or coordinate robot skills (Ahn et al., 2022; Royce et al., 2025), but OpenClaw-based deployments make the constraint layer more explicit. **ROSClaw** (Cardenas et al., 2026) connects OpenClaw to ROS 2 through an executive-layer contract that exposes robot capabilities, normalizes observations, validates actions before execution, and records both attempted and executed actions. The model proposes actions, but the validator decides whether those actions can reach the robot. This distinction is significant because different model backends vary substantially in how often they propose out-of-policy actions, so safety depends on the constraint layer rather than the model alone. **OpenGo** (Li et al., 2026b) tightens this boundary further by allowing the LLM only to select from pre-validated robot skills and assign bounded parameters. The executable body of each skill remains fixed after review and simulation validation. **ABot-Claw** (Huo et al., 2026) and **RoboClaw** (Li et al., 2026d) extend the same principle to longer-horizon settings, adding shared memory, critic feedback, multi-robot coordination, self-resetting skills, and recovery loops. Across these systems, physical deployment is not achieved by giving the agent more direct control. It is achieved by placing validated skills, feedback, and recovery mechanisms between model output and physical action.

While these systems focus on controlling physical action, **VisionClaw** (Liu et al., 2026a) demonstrates that embodied deployment also changes what must be protected. Instead of using OpenClaw as the main reasoning layer, VisionClaw routes video and audio from smart glasses to Gemini Live, which reasons over the scene and sends tool calls to an OpenClaw backend for execution. In this setup, OpenClaw is not the main perception model; it becomes the backend that executes actions based on continuous first-person video and audio. User studies report faster task completion and lower perceived difficulty compared with single-modality baselines, but the deployment lesson is that perception also requires boundaries. When an agent can continuously see and hear the user's environment, the risk is not only unsafe robot motion. Bystanders can move from being passively recorded to being actively identified, searched, or acted upon. Robotics therefore expands the constraint problem from safe motion to safe perception and privacy-aware action.

6.2 Healthcare: Grounding Clinical Context

Clinical deployment requires every action and every claim to be traceable to a role, a record, and an audit trail. Yang et al. (2026a) argue that hospital deployment failures are not only model-capability problems, but also system-design problems. Their system keeps OpenClaw's skill-based decomposition but replaces its permissive runtime with role-specific operating-system users, kernel-level isolation, approved file reads and

writes, and mandatory audit logging. Agents coordinate by making append-only updates to shared documents rather than sending unconstrained messages to each other. The most distinctive component is page-indexed Memory: each node in the hospital document tree contains a short manifest, and agents retrieve information by following these manifests until they reach specific record pages. This keeps clinical claims anchored to patient records rather than loosely connected retrieved chunks. On MIMIC-IV, this manifest-guided retrieval substantially outperforms flat RAG and metadata-filtered RAG, but the paper is careful about scope: the memory component is evaluated empirically, while the broader hospital operating-system design remains an architectural proposal.

While Yang et al. focus on hospital-wide traceability, **MedOpenClaw** (Shen et al., 2026) narrows the clinical setting to radiology and shows that bounded tools are necessary but not sufficient. The system connects VLM agents to 3D Slicer through a three-layer action space: basic viewer actions, evidence operations, and optional expert tools such as segmentation modules. Raw Python execution is disabled by default, and every Tool invocation, argument, viewer state, and generated artifact is logged for replayable audit. The main finding is the Tool-Use Paradox: adding professional segmentation tools can lower diagnostic accuracy because current VLMs cannot reliably ground textual references such as "the left frontal lesion" to precise locations in a 3D volume. In healthcare, more tools do not necessarily yield better decisions. Tools must be bounded, traceable, and spatially grounded before they can improve clinical reasoning.

6.3 Scientific Research: Limiting Research Authority

Healthcare shows that tools must be bounded and grounded; scientific research extends the same logic to authority itself. In open-ended science, the main deployment risk is giving one agent too much authority over the research process. An agent may generate hypotheses, search literature, write code, and summarize results, but it should not be allowed to play every scientific role at once. **ClawdLab** (Weidener et al., 2026) takes this lesson from the OpenClaw and Moltbook ecosystem, where social consensus and popularity signals were not enough to produce reliable knowledge. Its response is to constrain scientific authority through hard role restrictions. Agents serve as principal investigator, research analyst, scout, critic, or synthesizer, and API-level rules prevent them from acting outside their assigned role. Results advance only after structured adversarial critique and governance procedures such as majority, supermajority, or consensus voting. Evidence requirements are encoded as protocol constraints, so validation depends on tool outputs rather than social agreement. The contribution is mainly architectural, not a completed demonstration of autonomous science at scale. Its value for this survey is the constraint pattern: open-ended inquiry requires role limits, evidence gates, and audit trails before agent disagreement can be treated as scientific review.

HTC-Claw (Zeng et al., 2026) applies the same idea to a more structured setting: high-throughput computational pipelines in materials science. Here the question is not how to govern open-ended debate, but how to prevent language-model planning from directly changing simulation workflows. HTC-Claw uses OpenClaw as a decision-making layer for intent understanding, task planning, execution monitoring, and result analysis, while the computing layer handles first-principles calculations, workflow modules, Slurm scheduling, and post-processing. The key constraint is separation between planning and execution. The model may help decide what calculations should be run and how results should be interpreted, but simulation parameters and cluster jobs must pass through predefined workflow modules rather than raw model-generated values. The evaluation remains demonstrative, with case studies rather than benchmark comparisons against mature workflow managers. Still, the deployment lesson is clear: in computational science, the model may coordinate research, but it should not hold unchecked control over the compute pipeline.

Deployment Summary

Deployment turns OpenClaw's extensibility into a constraint problem. In open research settings, OpenClaw is useful because agents can gain more Skills, Tools, and autonomy. In high-stakes domains, trust comes from giving the agent less freedom: bounded actions in robotics, traceable context in healthcare, and limited authority in scientific workflows. The central deployment problem is therefore not how to make OpenClaw more capable, but how to decide what it must not be allowed to do.

Table 2: **OpenClaw benchmark catalog** (May 2026 snapshot). Twenty-three benchmarks are grouped by agent-lifecycle stage: before installation, during execution, and after deployment. Artifact: ✔ released, ◐ partial, 🔒 gated, — pending.

Benchmark	Focus	Scale	Key finding	Artifact
Skill scanner benchmarks (4) — before installation				
SkillFortifyBench (Bhardwaj, 2026)	lifecycle model	540 skills	formal lifecycle guarantees	✔
SkillSieve (Hou et al., 2026)	ClawHub triage	400 skills	scalable marketplace triage	✔
MalSkills (Wang et al., 2026e)	multi-artifact scan	200 skills	multi-artifact risk detection	◐
Red/Blue Skills (Hu et al., 2026)	submission risk	11,010 skills	lightweight submission-time prediction	✔
Agent attack benchmarks (7) — during execution				
CIK-Bench (Wang et al., 2026i)	state poisoning	12 scenarios	persistent state amplifies compromise	✔
ClawSafety (Wei et al., 2026)	injection vectors	120 cases	Skills are the highest-trust vector	✔
PASB (Wang et al., 2026h)	IPI + memory	131 skills	memory makes injection persistent	✔
SkillAttack (Duan et al., 2026)	real-skill exploits	171 skills	popular Skills contain latent exploits	✔
HarmfulSkillBench (Jiang et al., 2026b)	registry harm	200 skills	Skill loading amplifies harmful behavior	🔒
ATBench-Claw (Yang et al., 2026b)	trajectory safety	11 categories	trajectory audits expose runtime violations	✔
AgentHazard (Feng et al., 2026b)	cross-harness harm	2,653 cases	dependency hooks create cross-harness risk	✔
Agent task benchmarks (12) — after deployment				
LiveClawBench (Long et al., 2026)	live curated tasks	30 tasks	task complexity needs richer annotation	◐
ClawsBench (Li et al., 2026e)	cross-harness	44 tasks	harness choice shapes capability and safety	✔
Claw-Eval-Live (Li et al., 2026a)	refreshable Skills	105 tasks	live Skills enable refreshable evaluation	✔
ClawArena (Ji et al., 2026)	evolving information	64 tasks	agents must revise beliefs under conflict	✔
ClawBench-153 (Zhang et al., 2026e)	production websites	153 tasks	real websites remain difficult	◐
ClawGym-Bench (Bai et al., 2026)	ClawHub-mined	200 tasks	ClawHub can become a training substrate	—
GTA-2 (Wang et al., 2026c)	checkpoint grading	361 tasks	checkpoint grading captures long horizons	—
SEA-Eval (Jiang et al., 2026a)	sequential streams	92 streams	efficiency matters beyond success rate	◐
MetaClaw-Bench (Xia et al., 2026)	simulated workdays	934 tasks	self-improvement needs longitudinal tests	✔
ClawEnvKit (Li et al., 2026f)	generated envs	1,040 envs	environments can be generated automatically	✔
WildClawBench (Ma et al., 2026)	in-the-wild traces	—	skill evolution must be tested in the wild	◐
SkillLearnBench (Zhong et al., 2026)	skill generation	20 tasks	skill learning requires continual evaluation	✔

7 Benchmarks

The OpenClaw research community released 23 benchmarks between January and May 2026. Table 2 organizes them into a lifecycle view of open-agent evaluation: before installation, during execution, and after deployment. This growth shows that evaluation has become a central research problem in the OpenClaw

ecosystem. It also reveals that the main issue is not benchmark scarcity, but benchmark fragmentation. Existing benchmarks cover many risks and capabilities, but they differ in evaluated artifacts, threat models, task protocols, metrics, and release formats. One additional contribution, **SkillTester** (Wang et al., 2026d), proposes a paired utility-and-security scoring framework for Skill evaluation but does not include an empirical evaluation set.

The three benchmark families correspond to different points in the agent lifecycle. **Skill scanner benchmarks** evaluate Skills before installation, asking whether risky Skills can be detected before they enter a workspace. **Agent attack benchmarks** evaluate the running agent, asking whether poisoned state, injected content, malicious Skills, or vulnerable dependencies can compromise behavior. **Agent task benchmarks** evaluate deployed performance, asking whether the agent can complete useful work under realistic, evolving, or long-horizon conditions. Together, these benchmarks shift OpenClaw evaluation from isolated task success toward lifecycle coverage.

Within this lifecycle view, task benchmarks exhibit the clearest methodological shift. Early task sets are hand-curated or designed for cross-harness comparison. More recent benchmarks evaluate agents against the current state of the platform, refresh tasks using current ClawHub Skills, or generate environments automatically. **ClawGym-Bench** (Bai et al., 2026) and **ClawEnvKit** (Li et al., 2026f) go further by treating the ecosystem itself as a source of benchmark and training tasks. This changes the role of a benchmark: it is no longer only a fixed test set, but also a mechanism for tracking an evolving agent ecosystem.

This expansion also creates fragmentation. Security benchmarks are especially difficult to compare across papers because they measure different objects under different protocols. For example, **CIK-Bench** (Wang et al., 2026i) evaluates persistent state poisoning across 12 scenarios, **ClawSafety** (Wei et al., 2026) evaluates 120 injection cases across Skill, email, and web vectors, **PASB** (Wang et al., 2026h) studies prompt injection and memory persistence over 131 Skills, and **HarmfulSkillBench** (Jiang et al., 2026b) measures registry-level harm across 200 Skills. These benchmarks all study important OpenClaw risks, but their artifacts, threat surfaces, and harm definitions are not directly aligned. As a result, a stronger scanner, safer model, or more robust defense may simply be measured against a different evaluation protocol. Without shared artifacts and protocols, evaluation cannot easily distinguish real progress from changes in benchmark construction.

Beyond fragmentation within the ecosystem, a second gap concerns cross-platform comparison. Despite the depth of OpenClaw-native evaluation, major cross-platform agent benchmarks such as AgentBench (Liu et al., 2024), AgentDojo (Debenedetti et al., 2024), AgentHarm (Andriushchenko et al., 2025), OSWorld (Xie et al., 2024), SWE-Bench (Jimenez et al., 2024), Agent-SafetyBench (Zhang et al., 2024), ToolEmu (Ruan et al., 2024), and InjecAgent (Zhan et al., 2024) had not released an OpenClaw harness as of May 2026. Instead, the connection often runs in the opposite direction: OpenClaw papers reuse AgentDojo, InjecAgent, or LLMail-Inject (Abdelnabi et al., 2025) as transfer baselines, while those benchmark suites do not evaluate OpenClaw directly. This makes it difficult to compare OpenClaw with other agent frameworks on common tasks and common threat models.

The benchmark landscape therefore points directly to the open problems discussed next. Skill scanners, attack benchmarks, and task benchmarks now cover important parts of the agent lifecycle, but they do not yet form a shared measurement layer for provenance, oversight, constraint composition, or cross-platform comparison.

Benchmarks Summary

OpenClaw has many benchmarks but no shared measurement layer for constraint design. Constraints are difficult to compare across papers when studies use different artifacts, protocols, safety definitions, and release formats. The remaining challenge is therefore not more benchmarks, but shared substrates for measuring whether constraints work across the agent lifecycle.

8 Open Problems and Future Directions

Section 1 opened with a question: can AI agents become personal assistants that ordinary users can own, extend, and access through everyday interfaces? OpenClaw answers this question through extensibility: an open Skill marketplace, a self-hosted runtime, and an open agent social layer. The research surveyed in Sections 3–7 shows that this answer is incomplete rather than wrong. Openness creates the conditions for rapid capability growth, but trust requires explicit constraints on what agents remember, when they act, what authority they hold, and how their behavior is measured. The four open problems below turn this lesson into a research agenda, moving from memory provenance and oversight to constraint composition and evaluation convergence.

8.1 Memory Provenance

Existing defenses mostly focus on Tools, execution traces, and Skill supply chains. **MissClaw** (Zhang et al., 2026a) exposes a different gap: ordinary content can be read during autonomous browsing, stored in persistent Memory, and later treated as trusted context. This is not a malicious Tool call, a suspicious execution trace, or a poisoned Skill. It is a memory-formation problem. The next layer of defense therefore needs provenance-aware memory governance, where memory entries carry source, authorship, and trust metadata. Tagging direct inputs is straightforward in principle, but the difficult cases are multi-hop: content summarized from browsing, memories derived from other memories, and claims generated by the agent after reading untrusted material. Future systems need policies that decide which sources may influence long-term Memory, when memory writes require review, and how provenance should constrain later reasoning without disabling useful self-improvement.

Current benchmarks make this gap visible. Skill scanner benchmarks such as **SkillFortifyBench** (Bhardwaj, 2026) and **SkillSieve** (Hou et al., 2026) evaluate risky Skill artifacts before installation, while attack benchmarks such as **CIK-Bench** (Wang et al., 2026i) and **PASB** (Wang et al., 2026h) evaluate persistent state poisoning or memory-amplified injection during execution. What remains undermeasured is the boundary between these stages: how ordinary content enters long-term Memory, how provenance is preserved across summarization, and how remembered content affects later sessions. Memory provenance therefore requires not only new defenses, but also benchmark protocols for tracing memory formation over time.

8.2 Composable Oversight

Memory provenance addresses what enters the agent's context. Oversight addresses what the agent is allowed to do with it. Agent societies show that autonomy cannot be treated as safe by default. Wang et al. (2026a) formalize one version of this problem through a trilemma between continuous self-evolution, isolation from external oversight, and safety invariance. Current oversight mechanisms often sit outside the agent system: human monitors, kill switches, and post-hoc audit logs can intervene after risks appear, but they are not built into how the agent decides and acts (Shapira et al., 2026; Gruber & Hilgert, 2026). A stronger direction is to make oversight policies part of the platform itself. These policies should specify when actions need approval, how often autonomous cycles may run, and which agents are allowed to influence one another. They should also be declared, enforced, and logged like other platform rules. Future architectures should let users select oversight policies the way they currently select Skills: modular, replaceable, and enforceable without rewriting the underlying agent.

Existing benchmarks also make this problem difficult to measure. Most defense evaluations test one scanner, guardrail, runtime policy, or human-in-the-loop mechanism at a time. They provide less guidance on how multiple oversight mechanisms interact when deployed together, whether their decisions conflict, or whether they remain effective across repeated autonomous cycles.

8.3 Constraint Composition

Whereas oversight asks when to intervene, constraint composition asks how to specify limits in the first place. High-stakes deployments repeatedly rediscover the same lesson: trustworthy agents need clear limits

on what they are allowed to do. Robotics constrains physical action (Cardenas et al., 2026; Li et al., 2026d), healthcare constrains clinical context and tool use (Shen et al., 2026; Yang et al., 2026a), and scientific research constrains roles, evidence, and parameters (Weidener et al., 2026; Zeng et al., 2026). What is missing is a general framework for combining these constraints across domains. Developers should be able to specify what actions an agent may take, what data it may access, and what decisions it may make on its own, and the platform should enforce those limits consistently across Skills, Tools, Memory, and background execution. This requires more than a sandbox switch. The closest analogues are operating-system enforcement mechanisms such as SELinux policies or eBPF-based monitors, but agents need a policy layer that also understands language-level intent, what each Tool allows the agent to do, how actions affect Memory, and how tasks are delegated across agents. Evaluation remains domain-siloed as well. Robotics, healthcare, and scientific-workflow evaluations each define domain-specific constraints, but they rarely test whether those constraints can be expressed through a shared policy interface or transferred across domains. This makes it difficult to compare whether two systems are safer because they enforce stronger constraints, or simply because they evaluate different kinds of actions.

8.4 Evaluation Convergence

The preceding problems expose different measurement gaps: memory formation is not directly traced, oversight mechanisms are rarely evaluated in combination, and domain constraints remain difficult to compare across settings. The lifecycle view in Table 2 shows why these gaps persist. OpenClaw now has many benchmarks, but not yet shared evaluation infrastructure. Security papers often define their own attack surfaces, labeled Skill sets, harm metrics, and trajectory taxonomies, while task benchmarks differ in how they use live Skills, generated environments, and agent harnesses. Cumulative progress requires convergence at three levels. At the data layer, the community needs canonical ClawHub and Moltbook snapshots. At the benchmark layer, it needs shared security and task suites that new defenses and agents can report against. At the harness layer, OpenClaw must be integrated into major cross-platform benchmark suites rather than evaluated only inside OpenClaw-native studies. Refreshable and auto-generated benchmarks point toward a solution, but only if they become shared references rather than new forks.

Open Problems Summary

Turning open extensibility into trustworthy agents requires systematic constraint design. The four problems above are the concrete agenda for that design. Memory provenance constrains what the agent remembers. Composable oversight constrains when and how the agent is supervised. Constraint composition specifies how domain limits are declared and enforced. Evaluation convergence determines how progress on these constraints is measured.

9 Conclusion

This survey examined OpenClaw as an early stress test for open personal-agent ecosystems. We reviewed how its local-first, messaging-native, and skill-extensible design supports continuous improvement, broad tool use, persistent Memory, social interaction, domain deployment, and benchmark construction. Across these areas, the same pattern recurs. OpenClaw's openness enables rapid extension and experimentation, but it also creates new governance problems, security exposure, shallow collective behavior, deployment risk, and evaluation fragmentation. The platform's importance therefore lies not only in the capabilities it enables, but also in the constraints it reveals as missing.

The central lesson is not that OpenClaw's openness is wrong. Openness is what makes the platform useful, extensible, and scientifically productive. The lesson is that openness is incomplete without systematic constraint design. In the current ecosystem, extensibility is built directly into the platform: Skills can be installed, Tools can be called, Memory can persist, and agents can re-engage through Heartbeat cycles. Constraints are still added later as validators, scanners, sandboxes, role boundaries, audit logs, or benchmark-specific rules. Future open agent platforms should close this asymmetry by making constraints modular, inspectable, enforceable, and measurable.

The implications extend beyond OpenClaw. Any open agent platform that adopts user-installable extensions, persistent Memory, and social or autonomous interaction will face the same asymmetry between rapid capability growth and lagging constraint design. OpenClaw's value to the broader field is therefore not only as a system but as a case study. It shows what fails first when openness arrives before governance, and which design patterns must be in place for openness to remain trustworthy at scale: provenance-aware Memory, composable oversight, constraint composition, and evaluation convergence. Returning to the survey's opening question of whether ordinary users can own, extend, and access AI agents through everyday interfaces, OpenClaw shows that the answer depends as much on what platforms restrict as on what they enable. The path to democratized AI agents runs through systematic constraint design.

10 Limitations and Broader Impact

This survey has several limitations. OpenClaw is a rapidly evolving ecosystem, and many papers in our corpus are arXiv preprints or other preprint releases. As a result, implementation details, marketplace size, and reported empirical findings may change as the platform and literature mature. The taxonomy and benchmark catalog also reflect a May 2026 snapshot and will need to be updated as the ecosystem grows. We therefore emphasize recurring design patterns and cross-paper themes rather than treating any single reported number as a settled property of the ecosystem. Our scope is centered on OpenClaw-specific mechanisms. Although we use OpenClaw as a case study for open personal-agent platforms, other agent ecosystems may expose different constraints, deployment assumptions, or governance structures.

The broader impact of this survey is tied to the risks of open personal-agent platforms. Skills, persistent Memory, background execution, and public-facing agent societies can support useful personalization and automation, but they also create risks around supply-chain abuse, memory pollution, zero-click exposure, social manipulation, and unsafe deployment in high-stakes domains. This survey discusses attacks and vulnerabilities reported in publicly available work, but does not introduce new attack capabilities, implementation details, or exploit procedures. By organizing these risks around constraint design, we aim to support safer evaluation and deployment of open agent systems whose safeguards are explicit, enforceable, and measurable.

References

Sahar Abdelnabi, Aideen Fay, Ahmed Salem, Egor Zverev, Kai-Chieh Liao, Chi-Huang Liu, Chun-Chih Kuo, Jannis Weigend, Danyael Manlangit, Alex Apostolov, Haris Umair, João Donato, Masayuki Kawakita, Athar Mahboob, Tran Huu Bach, Tsun-Han Chiang, Myeongjin Cho, Hajin Choi, Byeonghyeon Kim, Hyeonjin Lee, Benjamin Pannell, Conor McCauley, Mark Russinovich, Andrew Paverd, and Giovanni Cherubin. LLMail-Inject: A dataset from a realistic adaptive prompt injection challenge. *arXiv preprint arXiv:2506.09956*, 2025.

Michael Ahn, Anthony Brohan, Noah Brown, Yevgen Chebotar, Omar Cortes, Byron David, Chelsea Finn, Chuyuan Fu, Keerthana Gopalakrishnan, Karol Hausman, et al. Do as i can, not as i say: Grounding language in robotic affordances. *arXiv preprint arXiv:2204.01691*, 2022.

Maksym Andriushchenko, Alexandra Souly, Mateusz Dziemian, Derek Duenas, Maxwell Lin, Justin Wang, Dan Hendrycks, Andy Zou, Zico Kolter, Matt Fredrikson, Eric Winsor, Jerome Wynne, Yarin Gal, and Xander Davies. AgentHarm: A benchmark for measuring harmfulness of LLM agents. In *International Conference on Learning Representations (ICLR)*, 2025.

Francisco Angulo de Lafuente, Teerth Sharma, Vladimir Veselov, Seid Mohammed Abdu, Nirmal Tej Kumar, and Guillermo Perry. OpenCLAW-P2P v6.0: Resilient multi-layer persistence, live reference verification, and production-scale evaluation of decentralized AI peer review. *arXiv preprint arXiv:2604.19792*, 2026.

Sinan Aral, Lev Muchnik, and Arun Sundararajan. Distinguishing influence-based contagion from homophily-driven diffusion in dynamic networks. *Proceedings of the National Academy of Sciences*, 106(51):21544–21549, 2009.

Fei Bai, Huatong Song, Shuang Sun, Daixuan Cheng, Yike Yang, Chuan Hao, Renyuan Li, Feng Chang, Yuan Wei, Ran Tao, Bryan Dai, Jian Yang, Wayne Xin Zhao, and Ji-Rong Wen. ClawGym: A scalable framework for building effective claw agents. *arXiv preprint arXiv:2604.26904*, 2026.

Albert-Laszlo Barabasi. The origin of bursts and heavy tails in human dynamics. *Nature*, 435(7039):207–211, 2005.

Alessandro Bellina, Giordano De Marzo, and David Garcia. Conformity and social impact on AI agents. *arXiv preprint arXiv:2601.05384*, 2026.

Varun Pratap Bhardwaj. Formal analysis and supply chain security for agentic ai skills. *arXiv preprint arXiv:2603.00195*, 2026.

Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021.

William J. Brady, Julian A. Wills, John T. Jost, Joshua A. Tucker, and Jay J. Van Bavel. Emotion shapes the diffusion of moralized content in social networks. *Proceedings of the National Academy of Sciences*, 114 (28):7313–7318, 2017.

Irvin Steve Cardenas, Marcus Anthony Arnett, Natalie Catherine Yeo, Lucky Sah, and Jong-Hoon Kim. ROSClaw: An OpenClaw ROS 2 framework for agentic robot control and interaction. *arXiv preprint arXiv:2603.26997*, 2026.

Harrison Chase. LangChain. `https://github.com/langchain-ai/langchain`, 2022.

Eason Chen, Ce Guan, A. Elshafiey, Zhonghao Zhao, Joshua Zekeri, Afeez Edeifo Shaibu, and Emmanuel Osadebe Prince. When AI agents teach each other: Discourse patterns resembling peer learning in the Moltbook community. *arXiv preprint arXiv:2602.14477*, 2026a.

Eason Chen, Ce Guan, Ahmed Elshafiey, Zhonghao Zhao, Joshua Zekeri, Afeez Edeifo Shaibu, Emmanuel Osadebe Prince, and Cyuan Jhen Wu. Openclaw ai agents as informal learners at moltbook: Characterizing an emergent learning community at scale. *arXiv preprint arXiv:2602.18832*, 2026b.

Eason Chen, Ce Guan, Zhonghao Zhao, Joshua Zekeri, Afeez Edeifo Shaibu, Emmanuel Osadebe Prince, Cyuan-Jhen Wu, and A. Elshafiey. When AI agents learn from each other: Insights from emergent AI agent communities on OpenClaw for human-AI partnership in education. *arXiv preprint arXiv:2603.16663*, 2026c. AIED 2026 Blue Sky Paper.

Tianyu Chen, Dongrui Liu, Xia Hu, Jingyi Yu, and Wenjie Wang. A trajectory-based safety audit of clawdbot (openclaw). *arXiv preprint arXiv:2602.14364*, 2026d.

Darren Cheng and Wen-Kwang Tsao. Agent privilege separation in OpenClaw: A structural defense against prompt injection. *arXiv preprint arXiv:2603.13424*, 2026.

Sandeep Chowdhary, Elsa Andres, Adriana Manna, Luka Blagojević, Leonardo Di Gaetano, and Gerardo Iñiguez. Temporal patterns of reciprocity in communication networks. *EPJ Data Science*, 12(1):1–15, 2023.

Cognition Labs. Introducing Devin, the first AI software engineer. `https://www.cognition.ai/blog/introducing-devin`, 2024.

Giordano De Marzo and David Garcia. Collective behavior of AI agents: the case of Moltbook. *arXiv preprint arXiv:2602.09270*, 2026.

Edoardo Debenedetti, Jie Zhang, Mislav Balunović, Luca Beurer-Kellner, Marc Fischer, and Florian Tramèr. Agentdojo: A dynamic environment to evaluate prompt injection attacks and defenses for LLM agents. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Xinhao Deng, Yixiang Zhang, Jiaqing Wu, Jiaqi Bai, Sibo Yi, Zhuoheng Zou, Yue Xiao, Rennai Qiu, Jianan Ma, Jialuo Chen, Xiaohu Du, Xiaofang Yang, Shiwen Cui, Changhua Meng, Weiqiang Wang, Jiaxing Song, Ke Xu, and Qi Li. Taming openclaw: Security analysis and mitigation of autonomous llm agent threats. *arXiv preprint arXiv:2603.11619*, 2026.

Ben Dong, Hui Feng, and Qian Wang. Clawdrain: Exploiting tool-calling chains for stealthy token exhaustion in OpenClaw agents. *arXiv preprint arXiv:2603.00902*, 2026.

Zenghao Duan, Yuxin Tian, Zhiyi Yin, Liang Pang, Jingcheng Deng, Zihao Wei, Shicheng Xu, Yuyao Ge, and Xueqi Cheng. SkillAttack: Automated red teaming of agent skills through attack path refinement. *arXiv preprint arXiv:2604.04989*, 2026.

Aysajan Eziz. Fast response or silence: Conversation persistence in an AI-agent social network. *arXiv preprint arXiv:2602.07667*, 2026.

Yi Feng, Chen Huang, Zhibo Man, Ryner Tan, Long P. Hoang, Shaoyang Xu, and Wenxuan Zhang. Moltnet: Understanding social behavior of ai agents in the agent-native moltbook. *arXiv preprint arXiv:2602.13458*, 2026a.

Yunhao Feng, Yifan Ding, Yingshui Tan, Xingjun Ma, Yige Li, Yutao Wu, Yifeng Gao, Kun Zhai, and Yanming Guo. AgentHazard: A benchmark for evaluating harmful behavior in computer-use agents. *arXiv preprint arXiv:2604.02947*, 2026b.

Yudong Gao, Zongjie Li, Yuanyuan Yuan, Zimo Ji, Pingchuan Ma, and Shuai Wang. Skillreducer: Optimizing llm agent skills for token efficiency. *arXiv preprint arXiv:2603.29919*, 2026.

Yuxu Ge. Governance architecture for autonomous agent systems: Threats, framework, and engineering practice. *arXiv preprint arXiv:2603.07191*, 2026.

Agam Goyal, Olivia Pal, Hari Sundaram, Eshwar Chandrasekharan, and Koustuv Saha. Social simulacra in the wild: AI agent communities on moltbook. *arXiv preprint arXiv:2603.16128*, 2026.

Jan Gruber and Jan-Niclas Hilgert. Foundations for agentic AI investigations from the forensic analysis of OpenClaw. *arXiv preprint arXiv:2604.05589*, 2026.

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

Zihan Guo, Zhiyu Chen, Xiaohang Nie, Jianghao Lin, Yuanjian Zhou, and Weinan Zhang. Skillprobe: Security auditing for emerging agent skill marketplaces via multi-agent collaboration. *arXiv preprint arXiv:2603.21019*, 2026.

Chaoyue He, Xin Zhou, Di Wang, Hong Xu, Wei Liu, and Chunyan Miao. Openclaw as language infrastructure: A case-centered survey of a public agent ecosystem in the wild. *Preprints.org*, 2026. doi: 10.20944/preprints202603.1060.v1.

Itai Himelboim. Civil society and online political discourse: The network structure of unrestricted discussions. *Communication Research*, 38(5):634–659, 2011.

David Holtz. The anatomy of the Moltbook social graph. *arXiv preprint arXiv:2602.10131*, 2026.

Florian Holzbauer, David Schmidt, Gabriel Gegenhuber, Sebastian Schrittwieser, and Johanna Ullrich. Malicious or not: Adding repository context to agent skill classification. *arXiv preprint arXiv:2603.16572*, 2026.

Yinghan Hou, Zongyou Yang, Zaihu Pang, and Xiujun Ma. SkillSieve: A hierarchical triage framework for detecting malicious AI agent skills. *arXiv preprint arXiv:2604.06550*, 2026.

Haichuan Hu, Ye Shang, and Quanjun Zhang. Red skills or blue skills? a dive into skills published on ClawHub. *arXiv preprint arXiv:2604.13064*, 2026.

Dongjie Huo, Haoyun Liu, Guoqing Liu, Dekang Qi, Zhiming Sun, Maoguo Gao, Jianxin He, Yandan Yang, Xinyuan Chang, Feng Xiong, Xing Wei, Zhiheng Ma, and Mu Xu. ABot-Claw: A foundation for persistent, cooperative, and self-evolving robotic agents. *arXiv preprint arXiv:2604.10096*, 2026.

Haonian Ji, Kaiwen Xiong, Siwei Han, Peng Xia, Shi Qiu, Yiyang Zhou, Jiaqi Liu, Jinlong Li, Bingzhou Li, Zeyu Zheng, Cihang Xie, and Huaxiu Yao. ClawArena: Benchmarking AI agents in evolving information environments. *arXiv preprint arXiv:2604.04202*, 2026.

Sihang Jiang, Lipeng Ma, Zhonghua Hong, Keyi Wang, Zhiyu Lu, Shisong Chen, Jinghao Zhang, Tianjun Pan, Weijia Zhou, Jiaqing Liang, and Yanghua Xiao. SEA-Eval: A benchmark for evaluating self-evolving agents beyond episodic assessment. *arXiv preprint arXiv:2604.08988*, 2026a.

Yukun Jiang, Yage Zhang, Michael Backes, Xinyue Shen, and Yang Zhang. HarmfulSkillBench: How do harmful skills weaponize your agents? *arXiv preprint arXiv:2604.15415*, 2026b.

Yukun Jiang, Yage Zhang, Xinyue Shen, Michael Backes, and Yang Zhang. "humans welcome to observe": A first look at the agent social network moltbook. *arXiv preprint arXiv:2602.10127*, 2026c.

Carlos E. Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik Narasimhan. SWE-bench: Can language models resolve real-world GitHub issues? In *International Conference on Learning Representations (ICLR)*, 2024.

Xisen Jin, Michael Duan, Qin Lin, Aaron Chan, Zhenglun Chen, Junyi Du, and Xiang Ren. Proof-of-guardrail in ai agents and what (not) to trust from it. *arXiv preprint arXiv:2603.05786*, 2026.

Julia Jose, Meghna Manoj Nair, and Rachel Greenstadt. Large-scale analysis of persuasive content on moltbook. *arXiv preprint arXiv:2603.18349*, 2026.

David Lazer, Alex Pentland, Lada Adamic, Sinan Aral, Albert-László Barabási, Devon Brewer, Nicholas Christakis, Noshir Contractor, James Fowler, Myron Gutmann, et al. Computational social science. *Science*, 323(5915):721–723, 2009.

Chenxin Li, Zhengyang Tang, Mingxin Huang, Yunlong Lin, Shijue Huang, Shengyuan Liu, Bowen Ye, Rang Li, Lei Li, Benyou Wang, and Yixuan Yuan. Claw-Eval-Live: A live agent benchmark for evolving real-world workflows. *arXiv preprint arXiv:2604.28139*, 2026a.

Frank Li. OpenClaw PRISM: A zero-fork, defense-in-depth runtime security layer for tool-augmented LLM agents. *arXiv preprint arXiv:2603.11853*, 2026a.

Hanbing Li, Xuewei Cao, Zhiwen Zeng, Yuhan Wu, Yanyong Zhang, and Yan Xia. OpenGo: An OpenClaw-based robotic dog with real-time skill switching. *arXiv preprint arXiv:2604.01708*, 2026b.

Lingyao Li, Renkai Ma, Chen Chen, Zhicong Lu, and Yongfeng Zhang. The rise of AI agent communities: Large-scale analysis of discourse and interaction on Moltbook. *arXiv preprint arXiv:2602.12634*, 2026c.

Ning Li. The moltbook illusion: Separating human influence from emergent behavior in AI agent societies. *arXiv preprint arXiv:2602.07432*, 2026b.

Ruiying Li, Yunlang Zhou, Yuyao Zhu, Kylin Chen, Jingyuan Wang, Sukai Wang, Kongtao Hu, Minhui Yu, Bowen Jiang, Zhan Su, Jiayao Ma, Xin He, Yongjian Shen, Yang Yang, Guanghui Ren, Maoqing Yao, Wenhao Wang, and Yao Mu. Roboclaw: An agentic framework for scalable long-horizon robotic tasks. *arXiv preprint arXiv:2603.11558*, 2026d.

Xiangyi Li, Kyoung Whan Choe, Yimin Liu, Xiaokun Chen, Chujun Tao, Bingran You, Wenbo Chen, Zonglin Di, Jiankai Sun, Shenghan Zheng, Jiajun Bao, Yuanli Wang, Weixiang Yan, Yiyuan Li, and Han-chung Lee. ClawsBench: Evaluating capability and safety of LLM productivity agents in simulated workspaces. *arXiv preprint arXiv:2604.05172*, 2026e.

Xirui Li, Ming Li, Ion Stoica, Cho-Jui Hsieh, and Tianyi Zhou. ClawEnvKit: Automatic environment generation for claw-like agents. *arXiv preprint arXiv:2604.18543*, 2026f.

Yu-Zheng Lin, Bono Po-Jen Shih, Hsuan-Ying Alessandra Chien, Shalaka Satam, Jesus Horacio Pacheco, Sicong Shao, Soheil Salehi, and Pratik Satam. Exploring silicon-based societies: An early study of the moltbook agent community. *arXiv preprint arXiv:2602.02613*, 2026.

Xiao Liu, Hao Yu, Hanchen Zhang, Yifan Xu, Xuanyu Lei, Hanyu Lai, Yu Gu, Hangliang Ding, Kaiwen Men, Kejuan Yang, Shudan Zhang, Xiang Deng, Aohan Zeng, Zhengxiao Du, Chenhui Zhang, Sheng Shen, Tianjun Zhang, Yu Su, Huan Sun, Minlie Huang, Yuxiao Dong, and Jie Tang. Agentbench: Evaluating llms as agents. In *International Conference on Learning Representations (ICLR)*, 2024.

Xiaoan Liu, DaeHo Lee, Eric J. Gonzalez, Mar Gonzalez-Franco, and Ryo Suzuki. VisionClaw: Always-on AI agents through smart glasses. *arXiv preprint arXiv:2604.03486*, 2026a.

Yujian Liu, Jiabao Ji, Li An, Tommi Jaakkola, Yang Zhang, and Shiyu Chang. How well do agentic skills work in the wild: Benchmarking LLM skill usage in realistic settings. *arXiv preprint arXiv:2604.04323*, 2026b.

Xiang Long, Li Du, Yilong Xu, Fangcheng Liu, Haoqing Wang, Ning Ding, Ziheng Li, Jianyuan Guo, and Yehui Tang. LiveClawBench: Benchmarking LLM agents on complex, real-world assistant tasks. *arXiv preprint arXiv:2604.13072*, 2026.

Ziyu Ma, Shidong Yang, Yuxiang Ji, Xucong Wang, Yong Wang, Yiming Hu, Tongwen Huang, and Xiangxiang Chu. SkillClaw: Let skills evolve collectively with agentic evolver. *arXiv preprint arXiv:2604.08377*, 2026.

Md Motaleb Hossen Manik and Ge Wang. Openclaw agents on moltbook: Risky instruction sharing and norm enforcement in an agent-only social network. *arXiv preprint arXiv:2602.02625*, 2026.

Smitha Milli, Micah Carroll, Yike Wang, Sashrika Pandey, Sebastian Zhao, and Anca D. Dragan. Engagement, user satisfaction, and the amplification of divisive content on social media. *PNAS Nexus*, 4(3):pgaf062, 2025.

João Moura. Crewai: Framework for orchestrating role-playing autonomous ai agents. `https://github.com/crewAIInc/crewAI`, 2024.

Kunal Mukherjee, Cuneyt Gurcan Akcora, and Murat Kantarcioglu. MoltGraph: A longitudinal temporal graph dataset of Moltbook for coordinated-agent detection. *arXiv preprint arXiv:2603.00646*, 2026.

H. C. W. Price, H. AlMuhanna, P. M. Bassani, M. Ho, and T. S. Evans. Let there be claws: An early social network analysis of AI agents on Moltbook. *arXiv preprint arXiv:2602.20044*, 2026.

Rob Royce, Marcel Kaufmann, Jonathan Becktor, Sangwoo Moon, Kalind Carpenter, Kai Pak, Amanda Towler, Rohan Thakker, and Shehryar Khattak. Enabling novel mission operations and interactions with ROSA: The robot operating system agent. In *2025 IEEE Aerospace Conference*, pp. 1–16, 2025. doi: 10.1109/aero63441.2025.11068426.

Yangjun Ruan, Honghua Dong, Andrew Wang, Silviu Pitis, Yongchao Zhou, Jimmy Ba, Yann Dubois, Chris J. Maddison, and Tatsunori Hashimoto. Identifying the risks of LM agents with an LM-emulated sandbox. In *International Conference on Learning Representations (ICLR)*, 2024.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. *Advances in neural information processing systems*, 36:68539–68551, 2023.

David Schmotz, Luca Beurer-Kellner, Sahar Abdelnabi, and Maksym Andriushchenko. Skill-inject: Measuring agent vulnerability to skill file attacks. *arXiv preprint arXiv:2602.20156*, 2026.

Zhengyang Shan, Jiayun Xin, Yue Zhang, and Minghui Xu. Don't let the claw grip your hand: A security analysis and defense framework for openclaw. *arXiv preprint arXiv:2603.10387*, 2026.

Natalie Shapira, Chris Wendler, Avery Yen, Gabriele Sarti, Koyena Pal, Olivia Floody, Adam Belfki, Alex Loftus, Aditya Ratan Jannali, Nikhil Prakash, et al. Agents of chaos. *arXiv preprint arXiv:2602.20021*, 2026.

Minjie Shen, Yanshu Li, Lulu Chen, Zhichao Fan, Yanhang Li, and Qikai Yang. From mind to machine: The rise of Manus AI as a fully autonomous digital agent. *arXiv preprint arXiv:2505.02024*, 2025.

Weixiang Shen, Yanzhu Hu, Che Liu, Junde Wu, Jiayuan Zhu, Chengzhi Shen, Min Xu, Yueming Jin, Benedikt Wiestler, Daniel Rueckert, and Jiazhen Pan. Medopenclaw: Auditable medical imaging agents reasoning over uncurated full studies. *arXiv preprint arXiv:2603.24649*, 2026.

Bronislav Sidik and Lior Rokach. Beyond static sandboxing: Learned capability governance for autonomous AI agents. *arXiv preprint arXiv:2604.11839*, 2026.

Significant Gravitas. AutoGPT. `https://github.com/Significant-Gravitas/AutoGPT`, 2023.

Luca Sodano, Sofia Sciangula, Amulya Galmarini, and Francesco Bertolotti. Emergence of fragility in LLM-based social networks: the case of Moltbook. *arXiv preprint arXiv:2603.23279*, 2026.

Peter Steinberger and OpenClaw Contributors. Openclaw: The open-source personal ai agent platform. `https://github.com/openclaw/openclaw`, 2026. Originally released as Clawdbot (Nov 2025), renamed Moltbot (Jan 27, 2026), then OpenClaw (Jan 29, 2026).

Yinggang Sun, Haining Yu, Wei Jiang, Xiangzhan Yu, Dongyang Zhan, Lixu Wang, Siyue Ren, Yue Sun, and Tianqing Zhu. A survey on the unique security of autonomous and collaborative LLM agents: Threats, defenses, and futures. *Preprints.org*, 2026. doi: 10.20944/preprints202602.1655.v1.

Surada Suwansathit, Yuxuan Zhang, and Guofei Gu. A security analysis of the openclaw ai agent framework. *arXiv preprint arXiv:2603.27517*, 2026.

Guiyao Tie, Jiawen Shi, Pan Zhou, and Lichao Sun. BadSkill: Backdoor attacks on agent skills via model-in-skill poisoning. *arXiv preprint arXiv:2604.09378*, 2026.

Uchi Uchibeke. Before the tool call: Deterministic pre-action authorization for autonomous AI agents. *arXiv preprint arXiv:2603.20953*, 2026.

Chenxu Wang, Chaozhuo Li, Songyang Liu, Zejian Chen, Jinyu Hou, Ji Qi, Rui Li, Litian Zhang, Qiwei Ye, Zheng Liu, Xu Chen, Xi Zhang, and Philip S. Yu. The devil behind moltbook: Anthropic safety is always vanishing in self-evolving ai societies. *arXiv preprint arXiv:2602.09877*, 2026a.

Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An open-ended embodied agent with large language models. *arXiv preprint arXiv:2305.16291*, 2023.

Haoyu Wang, Zibo Xiao, Yedi Zhang, Christopher M. Poskitt, and Jun Sun. SafeClaw-R: Towards safe and secure multi-agent personal assistants. *arXiv preprint arXiv:2603.28807*, 2026b.

Jize Wang, Xuanxuan Liu, Yining Li, Songyang Zhang, Yijun Wang, Zifei Shan, Xinyi Le, Cailian Chen, Xinping Guan, and Dacheng Tao. GTA-2: Benchmarking general tool agents from atomic tool-use to open-ended workflows. *arXiv preprint arXiv:2604.15715*, 2026c.

Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, Wayne Xin Zhao, Zhewei Wei, and Ji-Rong Wen. A survey on large language model based autonomous agents. *Frontiers of Computer Science*, 18(6), 2024.

Leye Wang, Zixing Wang, and Anjie Xu. SkillTester: Benchmarking utility and security of agent skills. *arXiv preprint arXiv:2603.28815*, 2026d.

Shenao Wang, Junjie He, Yanjie Zhao, Yayi Wang, Kan Yu, and Haoyu Wang. "elementary, my dear watson." detecting malicious skills via neuro-symbolic reasoning across heterogeneous artifacts. *arXiv preprint arXiv:2603.27204*, 2026e.

Yinjie Wang, Xuyang Chen, Xiaolong Jin, Mengdi Wang, and Ling Yang. Openclaw-rl: Train any agent simply by talking. *arXiv preprint arXiv:2603.10165*, 2026f.

Yuhang Wang, Haichang Gao, Zhenxing Niu, Zhaoxiang Liu, Wenjing Zhang, Xiang Wang, and Shiguo Lian. A systematic security evaluation of OpenClaw and its variants. *arXiv preprint arXiv:2604.03131*, 2026g.

Yuhang Wang, Feiming Xu, Zheng Lin, Guangyu He, Yuzhe Huang, Haichang Gao, Zhenxing Niu, Shiguo Lian, and Zhaoxiang Liu. From assistant to double agent: Formalizing and benchmarking attacks on OpenClaw for personalized local AI agent. *arXiv preprint arXiv:2602.08412*, 2026h.

Zijun Wang, Haoqin Tu, Letian Zhang, Hardy Chen, Juncheng Wu, Xiangyan Liu, Zhenlong Yuan, Tianyu Pang, Michael Qizhe Shieh, Fengze Liu, Zeyu Zheng, Huaxiu Yao, Yuyin Zhou, and Cihang Xie. Your agent, their asset: A real-world safety analysis of OpenClaw. *arXiv preprint arXiv:2604.04759*, 2026i.

Bowen Wei, Yunbei Zhang, Jinhao Pan, Kai Mei, Xiao Wang, Jihun Hamm, Ziwei Zhu, and Yingqiang Ge. ClawSafety: "Safe" LLMs, unsafe agents. *arXiv preprint arXiv:2604.01438*, 2026.

Lukas Weidener, Marko Brkić, Phillip Lee, Martin Karlsson, Kevin Noessler, and Paul Kohlhaas. From agent-only social networks to autonomous scientific research: Lessons from OpenClaw and Moltbook, and the architecture of ClawdLab and Beach.Science. *arXiv preprint arXiv:2602.19810*, 2026.

Oliver Wieczorek. How do AI agents talk about science and research? An exploration of scientific discussions on Moltbook using BERTopic. *arXiv preprint arXiv:2603.11375*, 2026.

Nigel Williams and Nicole Ferdinand. Form or function? early dynamics of the moltbook ai social media network. *ROBONOMICS: The Journal of the Automated Economy*, 7:90–90, 2026.

Fang Wu and Bernardo A Huberman. Novelty and collective attention. *Proceedings of the National Academy of Sciences*, 104(45):17599–17601, 2007.

Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Shaokun Zhang, Jiale Liu, Ahmed Hassan Awadallah, Ryen W White, Doug Burger, and Chi Wang. Autogen: Enabling next-gen llm applications via multi-agent conversation. *arXiv preprint arXiv:2308.08155*, 2023.

Zhiheng Xi, Wenxiang Chen, Xin Guo, Wei He, Yiwen Ding, Boyang Hong, Ming Zhang, Junzhe Wang, Senjie Jin, Enyu Zhou, et al. The rise and potential of large language model based agents: A survey. *Science China Information Sciences*, 68(2):121101, 2025.

Peng Xia, Jianwen Chen, Xinyu Yang, Haoqin Tu, Jiaqi Liu, Kaiwen Xiong, Siwei Han, Shi Qiu, Haonian Ji, Yuyin Zhou, Zeyu Zheng, Cihang Xie, and Huaxiu Yao. Metaclaw: Just talk – an agent that meta-learns and evolves in the wild. *arXiv preprint arXiv:2603.17187*, 2026.

Wenjie Xiao, Xuehai Tang, Biyu Zhou, Songlin Hu, and Jizhong Han. RouteGuard: Internal-signal detection of skill poisoning in LLM agents. *arXiv preprint arXiv:2604.22888*, 2026.

Tianbao Xie, Danyang Zhang, Jixuan Chen, Xiaochuan Li, Siheng Zhao, Ruisheng Cao, Toh Jing Hua, Zhoujun Cheng, Dongchan Shin, Fangyu Lei, Yitao Liu, Yiheng Xu, Shuyan Zhou, Silvio Savarese, Caiming Xiong, Victor Zhong, and Tao Yu. OSWorld: Benchmarking multimodal agents for open-ended tasks in real computer environments. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Wenxian Yang, Hanzheng Qiu, Bangqun Zhang, Chengquan Li, Zhiyong Huang, Xiaobin Feng, Rongshan Yu, and Jiahong Dong. When OpenClaw meets hospital: Toward an agentic operating system for dynamic clinical workflows. *arXiv preprint arXiv:2603.11721*, 2026a.

Zhonghao Yang, Yu Li, Yanxu Zhu, Tianyi Zhou, Yuejin Xie, Haoyu Luo, Jing Shao, Xia Hu, and Dongrui Liu. Benchmarks for trajectory safety evaluation and diagnosis in OpenClaw and Codex: ATBench-Claw and ATBench-Codex. *arXiv preprint arXiv:2604.14858*, 2026b.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations (ICLR)*, 2023.

Brandon Yee and Pairie Koh. Benchmarking emergent coordination in large-scale LLM populations: An evaluation framework on the MoltBook archive. *arXiv preprint arXiv:2603.03555*, 2026.

Zonghao Ying, Xiao Yang, Siyang Wu, Yumeng Song, Yang Qu, Hainan Li, Tianlin Li, Jiakai Wang, Aishan Liu, and Xianglong Liu. Uncovering security threats and architecting defenses in autonomous agents: A case study of openclaw. *arXiv preprint arXiv:2603.12644*, 2026.

Lianduan Zeng, Xiao Zhou, Xueru Zheng, Ning Gao, Lei Liu, Yunxuan Cao, Hongjian Chen, Zhongyang Wang, and Tongxiang Fan. The HTC-Claw: Automating discovery through high-throughput computational campaigns. *arXiv preprint arXiv:2604.06076*, 2026.

Qiusi Zhan, Zhixiang Liang, Zifan Ying, and Daniel Kang. InjecAgent: Benchmarking indirect prompt injections in tool-integrated large language model agents. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 10471–10506, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-acl.624. URL `https://aclanthology.org/2024.findings-acl.624/`.

Yechao Zhang, Shiqian Zhao, Jie Zhang, Gelei Deng, Jiawen Zhang, Xiaogeng Liu, Chaowei Xiao, and Tianwei Zhang. Mind your HEARTBEAT! Claw background execution inherently enables silent memory pollution. *arXiv preprint arXiv:2603.23064*, 2026a.

Yihao Zhang, Zeming Wei, Xiaokun Luan, Chengcan Wu, Zhixin Zhang, Jiangrong Wu, Haolin Wu, Huanran Chen, Jun Sun, and Meng Sun. ClawWorm: Self-propagating attacks across LLM agent ecosystems. *arXiv preprint arXiv:2603.15727*, 2026b.

Yunbei Zhang, Kai Mei, Ming Liu, Janet Wang, Dimitris N. Metaxas, Xiao Wang, Jihun Hamm, and Yingqiang Ge. Agents in the wild: Safety, society, and the illusion of sociality on moltbook. *arXiv preprint arXiv:2602.13284*, 2026c.

Yuntong Zhang, Sungmin Kang, Ruijie Meng, Marcel Böhme, and Abhik Roychoudhury. VeriGrey: Greybox agent validation. *arXiv preprint arXiv:2603.17639*, 2026d.

Yuxuan Zhang, Yubo Wang, Yipeng Zhu, Penghui Du, Junwen Miao, Xuan Lu, Wendong Xu, Yunzhuo Hao, Songcheng Cai, Xiaochen Wang, Huaisong Zhang, Xian Wu, Yi Lu, Minyi Lei, Kai Zou, Huifeng Yin, Ping Nie, Liang Chen, Dongfu Jiang, Wenhu Chen, and Kelsey R. Allen. ClawBench: Can AI agents complete everyday online tasks? *arXiv preprint arXiv:2604.08523*, 2026e.

Zhexin Zhang, Shiyao Cui, Yida Lu, Jingzhuo Zhou, Junxiao Yang, Hongning Wang, and Minlie Huang. Agent-SafetyBench: Evaluating the safety of LLM agents. *arXiv preprint arXiv:2412.14470*, 2024.

Haochen Zhao and Shaoyang Cui. ClawTrap: A MITM-based red-teaming framework for real-world OpenClaw security evaluation. *arXiv preprint arXiv:2603.18762*, 2026.

Shanshan Zhong, Yi Lu, Jingjie Ning, Yibing Wan, Lihan Feng, Yuyi Ao, Leonardo F. R. Ribeiro, Markus Dreyer, Sean Ammirati, and Chenyan Xiong. SkillLearnBench: Benchmarking continual learning methods for agent skill generation on real-world tasks. *arXiv preprint arXiv:2604.20087*, 2026.

Jiaying Zhu, Lyuye Zhang, Wenbo Guo, and Yang Liu. Skillclone: Multi-modal clone detection and clone propagation analysis in the agent skill ecosystem. In *Proceedings of the 41st IEEE/ACM International Conference on Automated Software Engineering (ASE)*, 2026a.

Ningyan Zhu, Huacan Wang, Jie Zhou, Feiyu Chen, Shuo Zhang, Ge Chen, Chen Liu, Jiarou Wu, Wangyi Chen, Xiaofeng Mou, and Yi Xu. SemaClaw: A step towards general-purpose personal AI agents through harness engineering. *arXiv preprint arXiv:2604.11548*, 2026b.

Yiming Zhu, Gareth Tyson, and Pan Hui. A comparative analysis of social network topology in reddit and moltbook. *arXiv preprint arXiv:2602.13920*, 2026c.