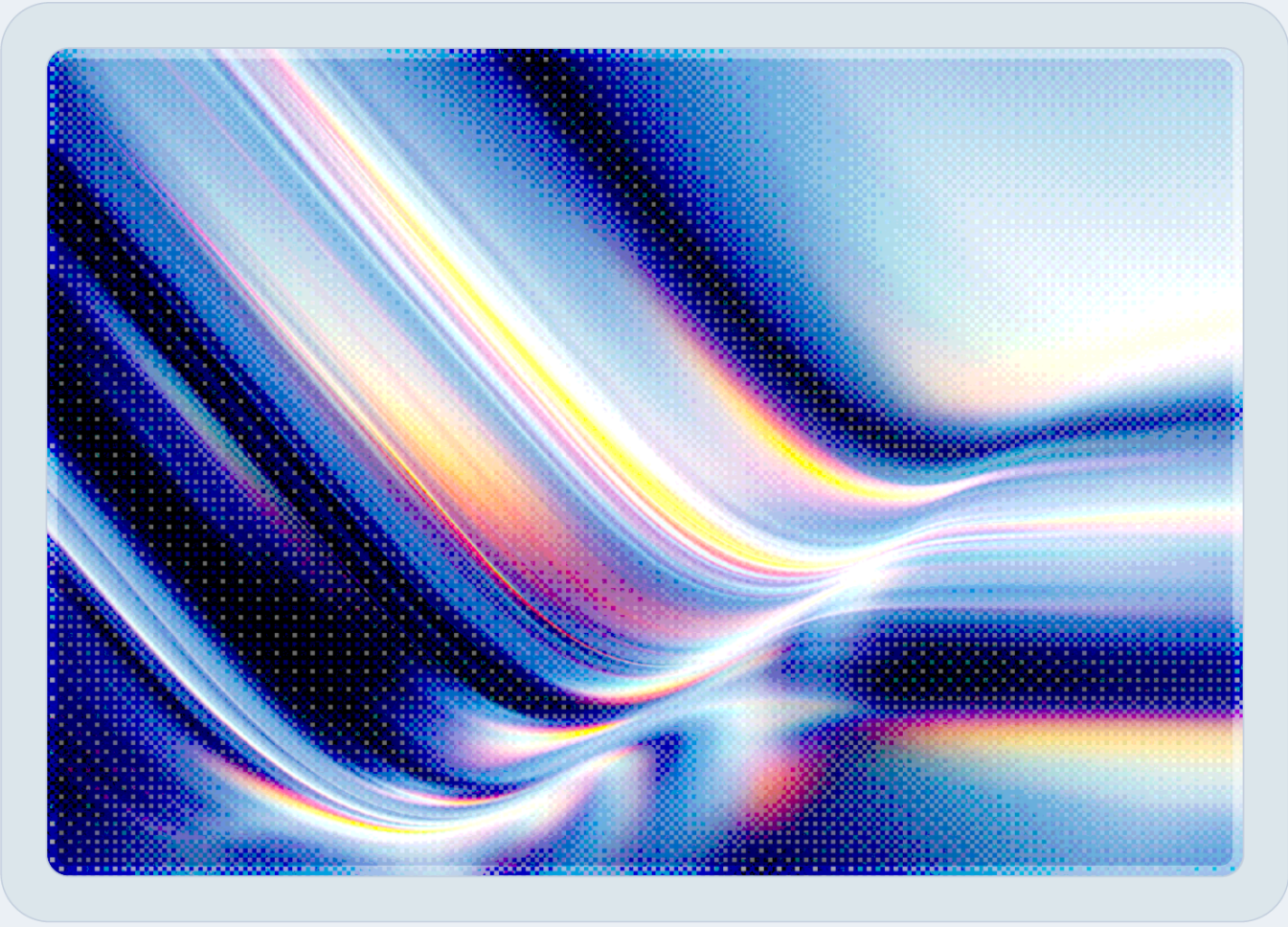
CSAI Foundation | Cloud Security Alliance

Skill Scanner Bypass: AI Agent Supply Chain Defense Gaps

Documented bypass techniques challenge every major public scanner for AI agent skills

2026-06-29

AI-assisted Rapid Research



CSAI | CSA cloud security alliance®

Key Takeaways

Current scanner generations have documented, reproducible bypass techniques across the most widely deployed tools, and structural limitations in remaining scanners suggest these gaps are architectural rather than incidental. The ClawHavoc campaign saw five malicious skills evade updated versions of both ClawScan and VirusTotal between February and May 2026 [3]; SkillSpector explicitly acknowledges that non-English content, image-based attacks, and runtime behavioral changes fall outside its current detection perimeter [6]; and formal analysis frameworks such as SkillFortify are structurally limited to executable code artifacts and cannot reason about natural-language instructions in SKILL.md [8]. Organizations should treat current skill scanners as one detection layer — not a gate — while the security tooling ecosystem develops more comprehensive detection approaches.

- No single scanner reliably detects all known attack classes; the ClawHavoc campaign saw five malicious skills evade updated versions of both ClawScan and VirusTotal between February and May 2026 [3].
- Document-Driven Implicit Payload Execution (DDIPE) achieves bypass rates between 11.6% and 33.5% under defended conditions; under identical defenses, conventional direct injection achieved 0% [5].
- Natural language attacks embedded in SKILL.md files are outside the scope of static code analyzers by design; NVIDIA's SkillSpector explicitly acknowledges that non-English content, image-based attacks, and runtime behavioral changes fall outside its current detection perimeter [6].
- Semantic manipulation of skill registry metadata can bias agent skill selection before any scanner is contacted, steering agents toward adversarial variants in 77.6% of paired trials [10].
- Behavioral dormancy — publishing clean code across many versions before activating a payload — defeated scanner trust signals in the postmark-mcp incident, which silently exfiltrated 3,000 to 15,000 corporate emails per day for more than a week before detection [12].

Background

AI agent skill ecosystems have evolved from experimental developer tools into production infrastructure supporting enterprises that deploy continuously-operational agents with access to internal data systems, email, databases, and code repositories — a transition that accelerated between 2023 and 2026. OpenClaw, an open-source AI agent framework, operates ClawHub, a community skill marketplace where thousands of third-party packages extend agent capabilities — enabling everything from database queries and email dispatch to code execution, file management, and external API access [2][3]. Competing and complementary ecosystems have emerged in parallel: the Model Context Protocol (MCP) defines a standard for connecting agents to tools via npm-distributed servers, while LangChain, CrewAI, AutoGPT, and other orchestration frameworks each maintain their own plugin or skill distribution channels. Across these ecosystems, the common design pattern is that skills run with the same operating-system permissions as the agent itself — giving a malicious skill the same access as the agent process, which in uncontained deployments amounts to arbitrary code execution on the host machine.

The breadth of this attack surface attracted systematic measurement in 2026. A large-scale empirical study scanning 42,447 agent skills across multiple registries found that 26.1% exhibit at least one security vulnerability, with 17 attack scenarios identified across three attack layers [4]. A parallel academic effort catalogued thousands of malicious tools in the wild that evade conventional detection [8]. A controlled security experiment published in 2026 demonstrated the practical reach of the problem: a single malicious skill, crafted to bypass existing scanners, successfully compromised more than 26,000 agents across individual and enterprise environments before detection [15]. These findings reflect a supply chain security ecosystem that has expanded far faster than its corresponding detection and governance infrastructure.

The security community has developed several dedicated scanning tools in response. NVIDIA open-sourced SkillSpector, which detects 68 vulnerability patterns across 17 categories via static analysis combined with an optional LLM-based semantic evaluation stage [6][7]. The qualixar/SkillFortify project introduced a formal analysis framework grounded in abstract interpretation, achieving 96.95% F1 with 100% precision on a 540-skill benchmark [8]. ClawHub integrated VirusTotal and its own ClawScan engine directly into the submission pipeline [1][3]. Academic researchers published SkillSieve, a hierarchical three-layer triage system combining static pre-filtering (regex, AST inspection, and metadata checks), parallel LLM sub-task semantic analysis, and a three-model jury for high-risk cases [9]. By mid-2026, organizations had more scanner choices than ever before. That abundance of tools did not translate into comprehensive coverage.

Security Analysis

What Scanners Are Designed to Detect

The most widely deployed skill scanners — including ClawScan, VirusTotal, and static analysis tools such as SkillSpector — share a common detection model: they examine executable artifacts of a skill using pattern matching and AST inspection. Even more sophisticated approaches such as SkillFortify's formal analysis and SkillSieve's LLM-layer pipeline apply their analysis primarily to code and dependency graphs, leaving the natural-language surfaces of a skill package outside their detection scope. SkillSpector's 17 detection categories cover the expected code-level attack classes: prompt injection strings embedded in code, dangerous subprocess calls, credential exfiltration patterns, privilege escalation via tool chaining, and malicious dependencies imported from compromised registries [6]. SkillFortify adds formal proofs of capability confinement and a trust score algebra computed across the entire Agent Dependency Graph [8]. These are real detection capabilities with demonstrated value: they detect a demonstrable portion of malicious skills, as validated by SkillFortify's 96.95% F1 score on benchmark data [8] and SkillSieve's triage performance [9], they reject dependency chains with known compromised packages, and they flag skills requesting permissions inconsistent with their declared purpose.

The limitation is structural. Static analysis operates on code. The modern AI agent skill is not only code: it includes a SKILL.md natural-language instruction file that the consuming AI agent reads as authoritative documentation, a metadata manifest that determines how skill registries surface the skill to users, and documentation pages whose code examples the agent may synthesize into its own task execution. Each of these non-code surfaces represents an attack vector that falls outside the scope of conventional static analysis by design.

Five Documented Bypass Techniques

Researchers and threat actors have developed at least five distinct approaches to evading skill security scanners, each exploiting a different structural gap.

File size inflation exploits the processing limits that content-analysis pipelines enforce before declining to examine a file. The brand-landingpage skill — a malicious payload used in the ClawHavoc campaign — padded its code with characters that inflated total file size beyond the threshold at which both ClawScan and VirusTotal decline to process content, successfully bypassing both scanners while delivering

functional but malicious behavior [1][3]. This technique requires no sophistication beyond understanding the target scanner's undocumented size limits, which are often discoverable through trial and error or source code review.

Document-Driven Implicit Payload Execution (DDIPE) is a more technically nuanced technique, described in detail in a 2026 academic study examining supply-chain poisoning against LLM coding agents [5]. Rather than embedding a standalone malicious instruction in skill code, DDIPE hides adversarial logic inside code examples and configuration templates within skill documentation — markdown code blocks and README snippets that the skill publisher presents as legitimate usage examples. Coding agents treat documentation as an authoritative reference when synthesizing code for a task; DDIPE exploits this tendency by crafting documentation examples that contain the attacker's payload within syntactically correct-looking sample code. Because the malicious content is written in the structure of an example rather than phrased as a behavioral instruction, existing alignment filters and static analyzers do not recognize it as an attack. Across 1,070 adversarial skills derived from 81 seed payloads covering 15 MITRE ATT&CK categories, DDIPE achieved bypass rates between 11.6% and 33.5% under conditions where conventional direct injection was completely defeated, with 2.5% of payloads evading both detection and alignment simultaneously [5].

Semantic registry manipulation operates at an earlier stage in the skill lifecycle: it manipulates how registries surface skills to agents before a scanner is ever contacted. Research examining SKILL.md-only attacks across ClawHub-style registry mechanisms demonstrated that small textual modifications to a skill's description can manipulate embedding-based retrieval search, achieving an 86% pairwise win rate in discovery-phase search results and raising the adversarial skill into the top-10 results 80% of the time [10]. In the selection phase, description-only framing of functionally equivalent adversarial versus benign skills caused agents to choose the adversarial variant in 77.6% of paired trials [10]. These manipulations occur entirely within natural-language metadata; no scanner that examines only executable code will intercept them.

Distributed payload decomposition defeats regex scanners and some LLM-based detectors by splitting malicious command sequences across multiple code blocks. The ClawHavoc campaign demonstrated this technique in practice: distributing commands across separate code blocks defeated both single-pass and multi-pass regex scanners that flag known malicious patterns as contiguous strings [1][2]. Because each individual fragment appears innocuous in isolation, automated tools may not reassemble and evaluate the full sequence. The 2604.02837 threat taxonomy formally categorizes this pattern as a fragmentation attack, noting its resistance to pattern-matching detection approaches [4].

Behavioral dormancy exploits the trust signals that scanner integrations build over time based on a package's publication history. The postmark-mcp supply chain incident, disclosed in late 2025, illustrated the technique precisely: the attacker published the package through 15 sequential clean versions (1.0.0

through 1.0.15), mirroring the official Postmark repository's code and executing legitimate API calls without triggering any alerts. Security tracking systems assigned the package baseline trust on the basis of this history. In version 1.0.16, a single line was added deep inside the package's main file, appending a hidden BCC address to every outgoing email payload. There was no prompt injection string, no malicious import, no unusual subprocess call — only a silent recipient address inserted into otherwise normal email API usage. The package proceeded to forward between 3,000 and 15,000 corporate emails per day to an attacker-controlled domain for more than a week before detection [12]. Static scanners had no basis on which to flag it; the payload was behavioral, not structural.

Evidence from the Field

The ClawHavoc campaign is the most extensively documented real-world test of skill scanner performance against a coordinated adversarial effort. The campaign introduced 1,184 malicious skills into ClawHub, targeting macOS and Windows systems running OpenClaw as a continuously operational agent [2][3]. OpenClaw's marketplace integrated VirusTotal scanning and its own ClawScan engine specifically to address the growing threat. Despite these controls, five malicious skills continued to evade detection between February and May 2026 after both scanners had been updated [3]. The ClawHavoc evidence suggests that scanner patches addressing individual techniques do not automatically close adjacent bypass classes: each of the five evasive skills exploited a structurally distinct gap, indicating that each technique requires independent detection logic rather than generalized defenses.

The following table summarizes the documented bypass techniques, the detection mechanisms each evades, and representative evidence.

Technique	Primary Gap Exploited	Scanner Evasion Evidence
File size inflation	Scanner processing size limits	brand-landingpage evaded ClawScan + VirusTotal [1][3]
DDIPE (documentation code examples)	Alignment filters + static analysis	11.6%–33.5% bypass under defended conditions; direct injection: 0% [5]
Semantic registry manipulation (SKILL.md)	Static code scope	77.6% adversarial selection rate; bypasses occur before scanner contact [10]
Distributed payload decomposition	Regex / single-pattern matching	Used in ClawHavoc; defeats multi-pass pattern scanners [1][2]

Technique	Primary Gap Exploited	Scanner Evasion Evidence
Behavioral dormancy (trusted version history)	Historical trust signal heuristics	postmark-mcp: 15 clean versions, then 3,000–15,000 emails/day exfiltrated [12]

Inherent Limits of Static Analysis Against Natural Language

SkillSpector's public documentation explicitly lists non-English skill content, image-based attacks, and dynamic or runtime behavioral changes as gaps the project acknowledges it does not yet address [6]. These are not edge cases. A skill that embeds malicious instructions in Japanese, Chinese, Arabic, or any other non-English language sits outside YARA signature coverage and outside the English-language training distribution of many LLM-based semantic analyzers. An image-embedded instruction — a PNG containing rendered text readable by an agent's vision capability — is invisible to any text-based static analyzer. A skill that changes behavior based on a runtime flag, a time-of-day check, or the presence of specific environment variables passes all pre-deployment checks while retaining the ability to activate a payload post-installation.

More fundamentally, the SkillFortify research notes that formal static analysis is limited to executable code artifacts and cannot reason about natural-language instructions in SKILL.md [8]. This is not a criticism of SkillFortify specifically; it is a structural property of the attack surface. SKILL.md files are read and interpreted by the consuming LLM at runtime, not by a static analyzer at review time. The semantic content of a SKILL.md file — including embedded behavioral directives phrased in natural language — shapes agent behavior through the same mechanism as a legitimate prompt, making detection a natural language understanding problem rather than a program analysis problem. Saha et al. [10] demonstrate that this surface is actively exploitable: the study examined real ClawHub skills and realistic registry mechanisms, publishing their code openly at github.com/ShoumikSaha/agent-skill-security.

Recommendations

Immediate Actions

Organizations deploying AI agents that consume third-party skills or MCP servers should stop treating scanner passage as a security gate and start treating it as a signal among many. No single publicly documented tool — SkillSpector, ClawScan, VirusTotal, or SkillFortify — currently catches every known bypass technique, and the structural gaps in each suggest that layered detection is necessary regardless of the specific tool chosen. Scanning remains worthwhile as a layer of defense, but the documented bypass rate under adversarial conditions means that a clean scan result cannot be treated as authorization to install. Before deploying any new skill into an environment with access to sensitive data or systems, security teams should verify publisher identity through out-of-band channels (signed commits, verified npm/PyPI maintainer accounts, organizational GitHub provenance), compare the claimed permissions in the skill manifest against the permissions the code actually requests, and review the SKILL.md natural-language instructions manually for behavioral directives that would not appear in static code analysis.

For high-sensitivity environments, organizations should apply sandbox isolation for all skill execution. Running skills in a restricted environment — network-isolated, with limited filesystem access and no access to credential stores — eliminates a substantial fraction of realistic attack impact even when a malicious skill evades all pre-deployment scanning. Sandbox execution converts a detection problem into a containment problem, which is a more tractable posture given the current state of scanner technology.

Short-Term Mitigations

Teams that have adopted a single scanner should move to a multi-scanner pipeline in the near term, applying tools with complementary detection models rather than duplicating coverage. A pipeline that combines static AST analysis (SkillSpector or SkillFortify), behavioral sandbox observation (dynamic execution in an isolated environment), and manual SKILL.md review addresses more of the documented bypass surface than any single tool alone. The SkillSieve three-layer architecture — static pre-filtering using regex and AST inspection, parallel LLM sub-task semantic analysis, and a multi-model jury for high-risk cases — offers a research-validated starting point for organizations building or procuring layered detection pipelines [9]. Multi-model verification is particularly relevant for DDIPE-style attacks: the 11.6%–33.5% single-layer bypass rates reported for DDIPE decline substantially when multiple

independent models must all be deceived simultaneously, and the study found that only 2.5% of adversarial payloads evaded both detection and alignment at once [5], suggesting that independent LLM judgment with diverse model choices provides meaningful residual protection.

Organizations should also instrument runtime behavior for installed skills. Monitoring for unexpected network egress, email recipients not present in outgoing message headers at install time, filesystem access outside declared permissions, and subprocess invocations inconsistent with the skill's stated purpose can surface behavioral dormancy attacks that pass all pre-deployment checks. The postmark-mcp technique — a single-line BCC addition in an otherwise legitimate package — would be detectable through email traffic analysis even without scanning the package itself.

Strategic Considerations

The AI agent skill supply chain currently lacks the governance infrastructure that mature software package ecosystems developed through years of incident response. The npm ecosystem has mandatory provenance attestations and sigstore-based signing for high-criticality packages; PyPI enforces two-factor authentication for maintainers of the top 1% of packages by download count; Sigstore provides supply chain transparency logs for container images. No equivalent signed-manifest standard exists for AI agent skills. Mandatory cryptographic signing of skill manifests — covering both executable code and natural-language SKILL.md content — would allow consuming agents and deployment tooling to verify that a skill's instructions at runtime match the version audited at publication time, closing the behavioral dormancy attack surface and providing a tamper-evident baseline from which manual SKILL.md review can proceed. Signing alone does not prevent an attacker from publishing a legitimately-signed malicious SKILL.md; detecting semantic manipulation in natural-language content requires a separate review layer, whether human, LLM-based, or both. CSA's AI Safety Initiative and the broader agentic AI security community should advocate for signed skill manifests as a baseline registry requirement across ClawHub, npm-based MCP servers, and all major skill distribution channels.

Benchmarking infrastructure should be a second priority. SkillSafetyBench and the SkillFortifyBench 540-skill corpus represent early steps toward standardized evaluation of scanner effectiveness [8][9]. Expanding these corpora with adversarial examples drawn from real-world campaigns — including ClawHavoc bypass techniques, DDIPE variants, and SKILL.md semantic attacks — would accelerate the development of more robust scanners and enable organizations to evaluate scanner purchases against consistent, reproducible test cases rather than vendor-supplied claims.

CSA Resource Alignment

This research note connects to several active CSA frameworks and publications.

MAESTRO (Multi-Agent Environment and System Threat and Risk Oracle) addresses threat modeling for agentic AI deployments across seven tiers. MAESTRO Tier 5 (Infrastructure and Scalability) covers supply chain risks to the underlying infrastructure on which agentic systems depend, including compromised dependencies, poisoned package repositories, and malicious tooling distributed through official channels. The skill scanner bypass techniques documented here map directly to Tier 5 threat categories. MAESTRO Tier 3 (Agent Memory and Context) is also relevant: semantic SKILL.md manipulation attacks are a form of context poisoning, injecting adversarial directives into the agent's runtime context through a trusted documentation channel.

CSA AICM (AI Controls Matrix) addresses AI supply chain security across its domains for Application Providers, Orchestrated Service Providers, and AI Customers. AICM controls covering third-party dependency validation, artifact integrity verification, and runtime behavioral monitoring are all applicable to skill supply chain risk. The AICM's shared responsibility model is particularly relevant here: skill registries, skill publishers, orchestration framework vendors, and deploying enterprises each bear distinct responsibility for supply chain security, and the current bypass landscape reflects gaps across all four layers.

CSA Agentic MCP Security Best Practices addresses MCP server security in depth, including tool poisoning attack vectors, server authentication, and least-privilege tool design [13]. The postmark-mcp attack pattern discussed in this note is a concrete instance of the threat categories that document covers. The May 2026 CSA research note on SKILL.md Agent Context Poisoning provides complementary analysis of the natural-language instruction manipulation surface [14].

References

[1] Underhill, K. "Hundreds of Malicious Skills Found in OpenClaw's ClawHub." eSecurity Planet, February 2026.

[2] CyberPress. "ClawHavoc Poisons OpenClaw's ClawHub With 1,184 Malicious Skills." CyberPress, February 2026.

[3] Unit 42, Palo Alto Networks. "OpenClaw's Skill Marketplace and the Emerging AI Supply Chain Threat." Palo Alto Networks Unit 42, June 2026.

[4] Li, Z., Wu, J., Ling, X., Cui, X., and Luo, T. "Towards Secure Agent Skills: Architecture, Threat Taxonomy, and Security Analysis." arXiv:2604.02837, April 2026.

[5] Qu, Y., Liu, Y., Geng, T., Deng, G., Li, Y., Zhang, L. Y., Zhang, Y., and Ma, L. "Supply-Chain Poisoning Attacks Against LLM Coding Agent Skill Ecosystems." arXiv:2604.03081, April 2026.

[6] NVIDIA. "SkillSpector: Security scanner for AI agent skills." GitHub, June 2026.

[7] NVIDIA. "NVIDIA Verified Agent Skills Provide Capability Governance for AI Agents." NVIDIA Technical Blog, May 2026.

[8] Bhardwaj, V. P. "Formal Analysis and Supply Chain Security for Agentic AI Skills." arXiv:2603.00195, March 2026.

[9] Hou, Y., Yang, Z., Pang, Z., and Ma, X. "SkillSieve: A Hierarchical Triage Framework for Detecting Malicious AI Agent Skills." arXiv:2604.06550, April 2026.

[10] Saha, S., Faghih, K., and Feizi, S. "Under the Hood of SKILL.md: Semantic Supply-chain Attacks on AI Agent Skill Registry." arXiv:2605.11418, May 2026.

[11] Mello, J. P. "The Postmark MCP server attack: 5 key takeaways." ReversingLabs Blog, October 2025.

[12] CSO Online. "Trust in MCP takes first in-the-wild hit via squatted Postmark connector." CSO Online, September 2025.

[13] CSA AI Safety Initiative. "Agentic MCP Security Best Practices." CSA Labs, March 2026.

[14] CSA AI Safety Initiative. "Agent Context Poisoning: SKILL.md and the New AI Supply Chain Attack Surface." CSA Labs, May 2026.

[15] CybersecurityNews. "Malicious AI Agent Skill Bypasses Security Scans and Seizes Full Control of Over 26,000 Agents." CybersecurityNews, June 2026.