Promptware: AI Agents as Attack Infrastructure

The Emerging Threat of Agentic Command-and-Control

Unofficial AI-assisted Research

Cloud Security Alliance AI Safety Initiative

2026-03-30

Key Takeaways

- **Promptware has evolved into a documented full kill chain.** A January 2026 academic analysis cataloged 36 documented promptware incidents between February 2023 and January 2026, spanning a seven-stage lifecycle that mirrors classical malware — from initial access and privilege escalation through persistence, command-and-control, lateral movement, and impact [1].
- **AI agents are now being used as command-and-control relays.** Researchers demonstrated in February 2026 that Microsoft Copilot and xAI Grok can be abused as covert C2 proxies without requiring API keys or user accounts, enabling malware to relay commands through trusted AI services [2].
- **Self-replicating AI worms exist and have been demonstrated.** The Morris II worm, demonstrated in March 2024, uses adversarial self-replicating prompts in RAG-based email assistants to steal data and propagate across AI platforms without user interaction [3].
- **Lateral movement is accelerating.** The share of documented promptware incidents involving cross-agent, cross-user, or cross-application propagation grew from zero in 2023 to 8 of 21 incidents recorded in 2025–2026 [1].
- **In-the-wild exploitation is confirmed.** Palo Alto Networks Unit 42 documented the first verified real-world detection of malicious indirect prompt injection against a production AI system — an event that occurred in December 2025 and was published in a research report on March 3, 2026 — identifying 22 distinct payload engineering methods [4].
- **State-sponsored actors have operationalized AI for attack development.** Google's Threat Intelligence Group documented North Korean, Iranian, and PRC-affiliated actors using AI tools across all stages of attack operations, including C2 infrastructure development and data exfiltration [5].

Background

When prompt injection was first described in 2022, it was analogized to SQL injection: a boundary-crossing vulnerability where attacker-controlled input could override intended instructions. That framing, while useful, obscured how far the threat would evolve. Prompt injection is no longer a discrete input-

validation flaw. It has become the delivery mechanism for a new category of attack payload that researchers now call "promptware" — malware that uses a large language model as its own execution engine [1].

The conceptual shift is important. Traditional malware executes machine code or script instructions against an operating system. Promptware executes natural-language instructions against an AI agent. Because modern AI agents are increasingly integrated with file systems, browsers, email clients, code repositories, API endpoints, and enterprise data stores, the blast radius of a successfully injected instruction can equal or exceed that of a traditional malware implant — and the attacker may never need to write a single line of executable code.

A January 2026 paper by Brodt, Feldman, Schneier, and Nassi — drawing on 36 documented incidents across three years — formalized the promptware kill chain into seven stages: initial access via direct or indirect injection or multimodal inputs; privilege escalation through jailbreaking; reconnaissance within the compromised agent context; persistence through memory poisoning; command-and-control via attacker-controlled external infrastructure; lateral movement across agents, users, or applications; and final-stage actions on objective, ranging from data exfiltration to remote code execution to unauthorized financial transactions [1].

The progression from proof-of-concept to production threat has been swift. In 2023, documented attacks typically involved only the second or third stage of the kill chain [1]. By 2025–2026, 15 of 21 recorded incidents exhibited four or more stages, with lateral movement appearing in eight cases [1]. The evolution tracks the deployment of agentic AI in enterprise environments: more capable agents processing richer, less-trusted inputs across broader organizational permissions create correspondingly richer attack surfaces.

Security Analysis

Indirect Prompt Injection as the Primary Vector

The most operationally significant attack vector in agentic systems is indirect prompt injection — the embedding of malicious instructions in external content that an agent retrieves and processes during normal operation. Unlike direct injection, which requires adversary access to the user interface, indirect injection operates through any data source the agent touches: web pages, documents, email content, calendar invitations, repository code comments, or database records.

OWASP's 2025 LLM Top 10 designates prompt injection as the leading risk for large language model deployments, specifically noting that agentic AI systems "massively increase the blast radius" by converting embedded instructions into real-world actions [6]. Unit 42 confirmed this theoretical concern with empirical data in a report published March 3, 2026, documenting the first verified in-the-wild detection of malicious indirect prompt injection — tracing the underlying event to December 2025 — operating against a production AI ad review system. Their analysis of attacker payloads identified 22 distinct engineering methods, with delivery approaches ranging from visible plaintext (37.8% of cases) to HTML attribute cloaking (19.8%) and CSS-based suppression (16.9%). Social engineering jailbreak techniques appeared in 85.2% of attacks, and 24.2% of malicious pages contained multiple simultaneous injected prompts. Observed impact objectives included SEO poisoning, credential theft, database destruction commands, and forced unauthorized purchases [4].

The persistence dimension of indirect injection deserves particular attention. Memory poisoning — injecting instructions into an agent's long-term memory so they persist across sessions — appeared in a substantial share of recent incidents in the Brodt et al. dataset, a technique that had essentially no presence in 2023 attacks [1]. The ChatGPT ZombAI case, documented in October 2024, demonstrated the technique in a production consumer AI deployment: instructions were written into ChatGPT's long-term memory causing the agent to regularly check an attacker-controlled GitHub repository for commands — functionally a remotely managed implant operating inside a consumer AI assistant [1].

AI Agents as Command-and-Control Infrastructure

The exploitation of AI platforms as C2 relays represents a qualitatively new threat category. Check Point Research demonstrated in February 2026 that Microsoft Copilot and xAI Grok can serve as covert command-and-control proxies for malware implants. The technique requires no API credentials. Malware on a compromised host enumerates system data, encodes it into URL query parameters, directs the AI service to fetch an attacker-controlled domain via its built-in web-browsing capability, and receives commands embedded in the AI's natural-language response. The result is C2 traffic that appears, from network monitoring tools, as ordinary user interaction with a trusted AI service [2]. Check Point Research reported that Microsoft acknowledged the findings and implemented mitigations to Copilot's web-fetch behavior [2].

A researcher known for AI security work published a complementary demonstration called "Agent Commander" in early 2026: a promptware-native C2 server where hijacked personal AI agents check in for new tasks and objectives expressed in natural language. Unlike classical C2, which communicates in binary protocols or obfuscated API calls, Agent Commander delivers instructions through the same

conversational interface the agent uses for legitimate work, making behavioral detection substantially more challenging — as the command channel is indistinguishable in format from legitimate user interactions [7].

VoidLink, a Linux post-exploitation C2 framework, exhibited code patterns that some researchers attributed to AI-assisted development [8]; if confirmed, such tooling would lower the technical barrier for adversaries while potentially complicating the attribution analysis based on developer coding style that security teams rely on.

Self-Replicating AI Worms and Lateral Movement

The Morris II worm, demonstrated by Cornell Tech and Technion researchers in March 2024, established that AI-native lateral movement is not merely theoretical. The worm uses adversarial self-replicating prompts embedded in RAG-based email environments: a single poisoned email causes the AI assistant to read, steal, and automatically resend confidential messages, with the injected prompt propagating to every downstream recipient. The technique was validated against Google Gemini Pro, OpenAI GPT-4, and the open-source LLaVA model [3].

EchoLeak (CVE-2025-32711, CVSS 9.3), discovered by Aim Security researchers, represents the escalation of this threat class into production enterprise systems. Disclosed in June 2025, EchoLeak chained four distinct bypasses to achieve zero-click exfiltration from Microsoft 365 Copilot: evasion of Microsoft's Cross Prompt Injection Attempt classifier, circumvention of link redaction through reference-style Markdown, abuse of auto-fetched images, and exploitation of a Microsoft Teams proxy permitted by the content security policy. The attack extracted data from Word documents, PowerPoint presentations, and Outlook emails with no user interaction required [9]. Microsoft patched the vulnerability server-side; no evidence of in-the-wild exploitation was reported.

Production AI coding assistants have emerged as a particularly high-risk target category because they combine code-execution capabilities with direct access to developer credentials, source repositories, and deployment pipelines. CVE-2025-53773, a prompt injection vulnerability in GitHub Copilot, achieved remote code execution across developer machines through a four-stage chain: injection via public repository code comments, modification of VS Code settings to enable autonomous execution mode, and subsequent command execution without user approval [1]. A related vulnerability in the Cursor AI coding tool (CVE-2025-59944) exploited a case-sensitivity bug in a protected file path, allowing injected instructions to escalate from configuration manipulation to remote code execution [10].

Model Context Protocol (MCP) — the emerging standard for connecting AI agents to tools and data sources — has introduced a new attack surface at the integration layer. CVE-2025-6515, documented by JFrog Security, demonstrated that MCP session IDs can be hijacked to disrupt model behavior without

modifying the model itself, establishing "prompt hijacking" as a distinct named technique [11]. AI coding assistants represented 7 of 21 documented promptware incidents in 2025–2026 [1], making developer tooling a priority category for security policy attention in current threat data.

Adversarial Scaling and State Actor Adoption

Many-shot jailbreaking, published by Anthropic researchers in April 2024, exploits the growth of LLM context windows to include up to 256 fabricated human-AI dialogues that progressively normalize prohibited outputs. Effectiveness scales with the number of shots following a power-law relationship. The technique succeeded against multiple major AI providers; fine-tuning and prompt classifier mitigations reduced the attack success rate from 61% to 2% in best-case scenarios, though the practical effectiveness of these defenses varies across attack sophistication levels and deployment configurations [12].

State-sponsored actors have moved beyond experimentation. Google's Threat Intelligence Group documented that North Korean, Iranian, and PRC-affiliated threat actors have used Gemini across all operational phases, including reconnaissance, phishing lure development, C2 infrastructure construction, and data exfiltration automation [5]. Microsoft's Security Blog published complementary analysis in March 2026 documenting AI as tradecraft — the systematic operational integration of AI tools across attack campaigns [13]. These disclosures indicate that AI tools have become embedded in adversarial operations, strongly suggesting that agentic attack capabilities are being evaluated and adopted beyond the research community [5, 13].

Recommendations

Immediate Actions

Organizations currently deploying AI agents in any capacity should audit the permissions and data access granted to those agents against the principle of least privilege. AI agents that can read, write, send, or execute should require explicit authorization for each capability class rather than receiving broad ambient access inherited from the user account they act on behalf of. Privileged actions — sending email, modifying files, executing code, making purchases — should be gated behind human confirmation workflows regardless of whether the requesting agent appears legitimate.

Security teams should explicitly add indirect prompt injection to their threat models for any AI-enabled application that processes external content. Web pages, documents, emails, calendar entries, and repository contents are all potential injection vectors; any AI workflow that retrieves and acts on such content without content isolation is exposed. Existing web application firewalls and input-validation controls provide limited or no defense against indirect prompt injection in AI agent workflows; AI-specific defenses — including content isolation, intent classifiers, and trust-boundary enforcement — are required to address the threat class.

Short-Term Mitigations

Architectural isolation provides a structural defense that is resilient to prompt-level bypass — unlike classifier-based mitigations, which must keep pace with evolving attack payloads. AI agents should operate within defined trust boundaries, with untrusted external content processed in isolated contexts that cannot directly trigger tool invocations or memory writes. Information-flow control approaches — such as Microsoft's FIDES architecture, which applies data-provenance tagging to constrain untrusted content from influencing privileged action channels — represent the direction of effective defense [14, 15].

Memory and persistence mechanisms in agentic systems require the same security controls applied to persistent storage elsewhere. Long-term agent memory should be auditable, and organizations should establish processes for detecting and clearing memory poisoning — instructions embedded in agent memory by prior sessions. Vector database contents used in RAG pipelines are equally susceptible and warrant analogous monitoring.

Network monitoring should be extended to cover AI service traffic. The Check Point findings demonstrate that normal-appearing interactions with trusted AI platforms can function as C2 channels [2]. Behavioral baselines for AI service usage — volume, timing, query patterns, external domain retrievals — can expose anomalous patterns consistent with relay abuse, even when payload content is not inspectable.

Strategic Considerations

The seven-stage promptware kill chain maps directly to established detection and response frameworks; organizations should begin building detection capabilities stage-by-stage rather than waiting for comprehensive AI-native security tooling to mature. Among the seven kill chain stages, privilege escalation attempts, anomalous memory writes, and unexpected outbound retrieval requests offer particularly actionable detection signals given the availability of existing monitoring infrastructure, though dedicated empirical evaluation of detection efficacy across stages has not yet been published.

Identity governance for AI agents requires the same rigor applied to human privileged accounts. The 2025 CSA Agentic Identity Survey found that 40% of organizations already have AI agents in production, yet only 18% are highly confident in their existing identity and access management systems' ability to govern agent identities [16]. Agents should be provisioned with unique, auditable, time-bound credentials; static shared credentials present elevated risk in any agentic deployment because a compromised agent inherits the full credential scope, providing meaningful containment only when credentials are scoped appropriately.

AI coding assistants, currently the highest-volume application category in empirical threat data for promptware incidents, warrant specific policy attention. Developer AI tools that can browse repositories, execute code, and modify configuration files need the same privileged-access workstation controls applied to human developers with equivalent access — including session logging, behavioral monitoring, and scope-limited credentials.

CSA Resource Alignment

This analysis connects directly to several active CSA frameworks and research streams. The **MAESTRO** (Multi-Agent Environment, Security, Threat, Risk, and Outcome) framework provides the threat modeling methodology best suited to the multi-stage, cross-agent attack patterns described here, including goal manipulation, inter-agent communication vulnerabilities, and cascading failure scenarios [17].

The **CSA Agentic AI Red Teaming Guide** documents 12 threat categories for autonomous agents — including agent authorization and control hijacking, multi-agent exploitation, and agent untraceability — that map directly to promptware kill chain stages. Its catalog of 629 security test cases and 16 LLM agent evaluations provides practitioners with actionable testing methodology [18].

The **CSA LLM Threats Taxonomy** classifies prompt injection within a nine-category threat framework and offers asset classification and lifecycle coverage aligned with the NIST AI RMF and MITRE ATLAS [19]. Organizations implementing the taxonomy should treat indirect prompt injection as a first-tier threat given the empirical evidence of in-the-wild exploitation.

The **CSA Agentic Identity Survey** findings on IAM gaps and the **Securing LLM-Backed Systems** guidance on authorization practices together form the governance foundation for the identity and permission controls recommended in this note [16, 20].

References

[1] Brodt, O., Feldman, E., Schneier, B., and Nassi, B. "The Promptware Kill Chain: How Prompt Injections Gradually Evolved Into a Multi-Step Malware Delivery Mechanism." arXiv:2601.09625v2. January 2026. https://arxiv.org/html/2601.09625v2

[2] Check Point Research. "AI in the Middle: Turning Web-Based AI Services into C2 Proxies — The Future of AI-Driven Attacks." Check Point Research Blog. February 17, 2026. https://research.checkpoint.com/2026/ai-in-the-middle-turning-web-based-ai-services-into-c2-proxies-the-future-of-ai-driven-attacks/

[3] Cohen, S., Bitton, R., and Nassi, B. "Here Comes The AI Worm: Unleashing Zero-click Worms that Target GenAI-Powered Applications." arXiv:2403.02817. Cornell Tech / Technion. March 5, 2024. https://arxiv.org/abs/2403.02817

[4] Palo Alto Networks Unit 42. "Fooling AI Agents: Web-Based Indirect Prompt Injection Observed in the Wild." Unit 42 Research Blog. March 3, 2026. https://unit42.paloaltonetworks.com/ai-agent-prompt-injection/

[5] Google Cloud. "Threat Actor Use of AI Tools." Google Threat Intelligence Group AI Threat Tracker. 2025. https://cloud.google.com/blog/topics/threat-intelligence/threat-actor-usage-of-ai-tools

[6] OWASP Gen AI Security Project. "LLM01:2025 Prompt Injection." OWASP Top 10 for Large Language Model Applications, 2025 Edition. https://genai.owasp.org/llmrisk/llm01-prompt-injection/

[7] Rehberger, J. (wunderwuzzi). "Agent Commander: Promptware-Powered Command and Control." embracethered.com. March 16, 2026. https://embracethered.com/blog/posts/2026/agent-commander-your-agent-works-for-me-now/

[8] CyberPress. "VoidLink Showcases AI Malware." February 11, 2026. https://cyberpress.org/voidlink-showcases-ai-malware/

[9] Aim Security. "EchoLeak: The First Zero-Click Prompt Injection Exploit in a Production LLM System." arXiv:2509.10540. September 2025. https://arxiv.org/abs/2509.10540

[10] McHugh, B., Sekrst, K., and Cefalu, A. (Preamble, Inc.). "Prompt Injection 2.0: Hybrid AI Threats." arXiv:2507.13169v1. 2025. https://arxiv.org/html/2507.13169v1

[11] JFrog Security. "CVE-2025-6515: MCP Prompt Hijacking Vulnerability." JFrog Security Blog. 2025. https://jfrog.com/blog/mcp-prompt-hijacking-vulnerability/

[12] Anthropic Research. "Many-Shot Jailbreaking." April 2, 2024. https://www.anthropic.com/research/many-shot-jailbreaking

[13] Microsoft Security Blog. "AI as Tradecraft: How Threat Actors Operationalize AI." March 6, 2026. https://www.microsoft.com/en-us/security/blog/2026/03/06/ai-as-tradecraft-how-threat-actors-operationalize-ai/

[14] Anthropic Research. "Mitigating the risk of prompt injections in browser use." 2025. https://www.anthropic.com/research/prompt-injection-defenses

[15] Microsoft MSRC. "How Microsoft Defends Against Indirect Prompt Injection Attacks." MSRC Blog. July 2025. https://www.microsoft.com/en-us/msrc/blog/2025/07/how-microsoft-defends-against-indirect-prompt-injection-attacks

[16] Baron, H. et al. "Securing Autonomous AI Agents." Cloud Security Alliance. Published February 2026 (survey fieldwork conducted September–October 2025).

[17] Cloud Security Alliance. "Cloud Threat Modeling 2025 v2.0." CSA Top Threats Working Group. 2025.

[18] Huang, K. et al. "Agentic AI Red Teaming Guide." Cloud Security Alliance AI Organizational Responsibilities Working Group. 2025.

[19] Burke, S., Capotondi, M., Catteddu, D., and Huang, K. "CSA Large Language Model (LLM) Threats Taxonomy." Cloud Security Alliance. 2025.

[20] Lee, N. and Voicu, L. "Securing LLM-Backed Systems: Essential Authorization Practices." CSA AI Technology and Risk Working Group. 2025.