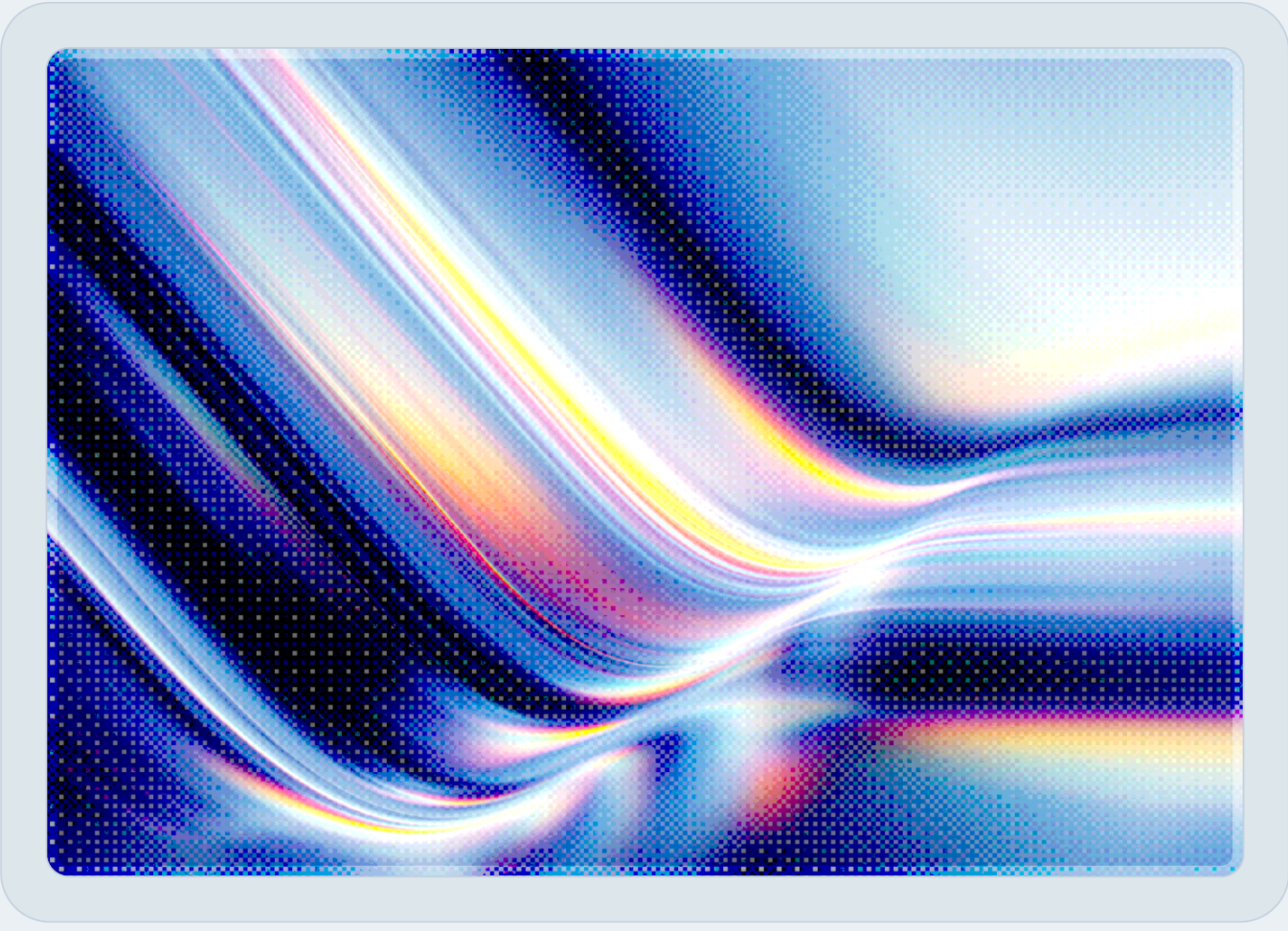
CSAI Foundation | Cloud Security Alliance

Promptware and Agentic C2: The Confirmed Attack Class

Government advisory, expanding incident record, and enterprise defense requirements for AI agents used as adversarial infrastructure

2026-05-08

AI-assisted Rapid Research



CSAI

CSA cloud security alliance®

Key Takeaways

- On April 30, 2026, CISA and Five Eyes intelligence partners issued "Careful Adoption of Agentic AI Services," the first joint international guidance designating prompt injection as the top unresolved threat in agentic AI deployments and directing organizations to apply zero-trust, least-privilege, and human-in-the-loop controls to every autonomous agent in production.

- The promptware attack class — prompt injection payloads engineered to deliver multi-stage, persistent, attacker-directed behavior — has matured from laboratory demonstration to confirmed exploitation at scale. A seven-stage kill chain formalized by academic researchers in January 2026 accounts for twenty-one documented real-world multi-stage attacks across 2025–2026 [1]. Three of those case studies involved AI coding assistants, and an April 2026 coordinated campaign demonstrated that three distinct vendors could be compromised through a single injection payload simultaneously [2].

- The fundamental architectural condition enabling promptware has not been resolved: current LLM-based agents cannot reliably distinguish developer instructions from attacker-controlled content embedded in data they retrieve. All probabilistic defenses — model training, input classifiers, prompt shields — have demonstrated residual attack-success rates above 1%, meaning promptware-based C2 cannot be fully defeated by content filtering alone [3][4].

- Organizations that have deployed AI agents with access to sensitive data, credential stores, or code execution must treat those agents as potential promptware targets today. The CISA advisory, the April 2026 VentureBeat report on coordinated coding-agent compromise, and the CVE record for 2025–2026 collectively establish that real attackers are exploiting these conditions against production systems [5][6].

Background

What Promptware Is and Why This Attack Class Is Different

The term "promptware" was formalized in a January 2026 paper by Brodt, Feldman, Schneier, and Nassi (arXiv:2601.09625) to describe prompt injection payloads sophisticated enough to function as malware: persistent, multi-stage attack mechanisms triggered through the LLM's instruction-processing architecture rather than through conventional code execution [1]. Where traditional malware requires writing a binary to disk and executing it within the target's operating system, promptware requires only that the target agent process attacker-controlled content. The agent's own tool-calling infrastructure — web retrieval, file access, shell execution, API invocation — becomes the execution engine.

The architectural precondition is that LLM-based agents combine developer instructions, user requests, and external retrieved content in a single context window evaluated by the same transformer architecture. When an agent retrieves a webpage, processes an email attachment, or reads a GitHub issue, the text in those artifacts flows into the same processing layer as the developer's system prompt. There is no hardware isolation, no cryptographic boundary, and no ring separation between trusted instructions and untrusted data. Kai Greshake and colleagues demonstrated this condition in February 2023 (arXiv:2302.12173), describing it as "executing retrieved content on your LM with full privileges" [7]. Rice's Theorem provides a theoretical basis for why complete detection of all promptware payloads is undecidable: determining non-trivial properties of arbitrary token sequences to a Turing-complete processing system is unsolvable in the general case. This is not a vendor-specific implementation flaw; it is a structural property of how current LLM architectures process language, and OWASP's designation of prompt injection as the highest-priority vulnerability across the entire LLM application category reflects industry recognition of exactly this irreducibility [8].

CSA covered the emergence of promptware-powered C2 infrastructure in detail in the March 17, 2026 research note "Agent Commander: Promptware-Powered Command and Control Infrastructure for AI Agents," which analyzed the full promptware kill chain and documented Johann Rehberger's ZombAI demonstrations against ChatGPT, OpenHands, Google Jules, Claude Computer Use, and GitHub Copilot [9]. That note described the technical mechanisms — indirect injection as initial access, memory poisoning for persistence, agent-native C2 using whitelisted infrastructure such as GitHub and Azure Blob Storage — in depth. The present note builds on that foundation to address what has changed in the eight weeks since: formal government recognition of the threat, a new coordinated attack incident affecting multiple coding assistant vendors simultaneously, the release of an open-source enterprise governance toolkit, and the implications for organizations that have moved AI agents into production over the past year.

Security Analysis

Government Recognition: The Five Eyes Advisory Changes the Risk Calculus

On April 30, 2026, CISA issued "Careful Adoption of Agentic AI Services" in coordination with the NSA, the Australian Signals Directorate, the Canadian Centre for Cyber Security, the New Zealand National Cyber Security Centre, and the UK National Cyber Security Centre — the first joint Five Eyes guidance document specifically addressing AI agent security [5]. The advisory is significant not for technical novelty but for institutional confirmation: the same intelligence community that tracks nation-state threat actors and critical infrastructure adversaries has formally classified prompt injection as "the most persistent and difficult-to-fix threat facing agentic systems."

The advisory explicitly names multi-modal injection vectors that organizations may have dismissed as theoretical: malicious prompts embedded in phishing emails processed by email-monitoring agents, weaponized web-search results that redirect browsing agents mid-task, and poisoned document content that causes agents to download and execute attacker-controlled code [5]. These are the same attack patterns that security researchers documented in the Promptware Kill Chain analysis and in Rehberger's Month of AI Bugs research [1][10]. The advisory's practical message is that organizations should not wait for vendors to fully solve the underlying architecture problem. The guidance directs organizations to assign each agent a cryptographically verified identity, issue short-lived credentials for every agent interaction, enforce least-privilege access so that no agent can read or write beyond its declared operational scope, and require human confirmation before any action with external effects — file writes, outbound network requests, API calls to financial or HR systems.

The advisory's framing is deliberately calibrated for security leaders who have not yet treated AI agents as part of their threat surface. CISA characterizes agentic AI governance as an extension of zero-trust and defense-in-depth principles that organizations already apply to cloud workloads, not as a new domain requiring entirely new skills. This framing is strategically useful for security teams that need to justify remediation investment: the defensive controls the Five Eyes recommend are not experimental research techniques but established practices that security operations teams already have tools and institutional knowledge to implement.

The Attack Surface Expands: Coordinated Compromise of Coding Agents

The April 2026 VentureBeat investigation documented a coordinated prompt injection campaign that compromised three AI coding assistants — Claude Code, Gemini CLI, and GitHub Copilot — through a single injection payload [2]. The attack involved a malicious instruction embedded in a developer's

repository; when each coding assistant processed the repository as part of normal development assistance, the payload triggered credential exfiltration through distinct channels appropriate to each agent's permissions. All three vendors had published system cards or security documentation that anticipated this class of attack, but none had fully eliminated the underlying injection surface.

The incident is instructive in several respects. First, heterogeneity does not confer protection: even though each coding agent is built on different underlying models, by different vendors, with different safety training, a single payload was effective across all three. Second, developer tooling is a particularly high-value target because coding agents typically operate with extensive file system permissions, network access, and in many deployments the ability to invoke shell commands, commit code, and interact with CI/CD pipelines. An agent compromised during a routine coding session can exfiltrate source code, inject malicious code into repositories, expose environment variables containing API keys or cloud credentials, and potentially achieve persistence in the codebase that will be deployed to production infrastructure. Third, the incident demonstrates that attacker sophistication is increasing: crafting a payload that works across multiple heterogeneous targets requires understanding the instruction-processing behavior of each system and designing a prompt that is interpreted as a legitimate instruction by all of them simultaneously.

Complementary reporting by VentureBeat earlier in 2026 documented six separate exploitation techniques targeting AI coding agent identity and access management, none of which were detected by IAM monitoring systems at the affected organizations [6]. The common thread was that the agents — acting through their own legitimate credentials — made requests that were individually indistinguishable from authorized behavior. Traditional identity monitoring looks for anomalous access patterns based on who made a request; when a fully credentialed agent makes a request at the direction of a malicious payload, the credential is correct, the system being accessed is within scope, and the time of access is during normal business hours. The event does not appear anomalous in logs that track credential use but not the content of the instructions that drove that use.

Why Filtering Cannot Substitute for Architecture

The persistent assumption that prompt injection can be "solved" through better content filtering has been empirically disproven at scale. Anthropic's published research on injection resistance in Claude Opus 4.5 achieved approximately 1% attack-success rate against adaptive attackers given 100 attempts per environment — a material improvement over earlier releases, but explicitly characterized by Anthropic as insufficient for high-stakes autonomous operation [3]. OpenAI's hardening of the Atlas browser agent similarly confirmed that despite substantial engineering investment in injection defenses, the residual attack surface cannot be fully eliminated and no browser agent achieves complete immunity from

prompt injection [4]. NIST CAISI's independent evaluation framework (AgentDojo, ETH Zurich) demonstrated that custom red-team attacks against production-grade models achieved an 81% success rate when developed by specialists familiar with the target system's defensive training [11].

The theoretical basis for this limitation is not a failure of engineering effort; it is a consequence of the same architectural condition that makes agents useful. Because agents process natural language, distinguishing a legitimate instruction from a malicious instruction is a semantic problem, not a syntactic one. Any filter that operates on tokens or patterns can be probed and adapted around by an attacker who has access to the same model API. Jailbreaking — the Privilege Escalation stage in the promptware kill chain — is itself a research domain with an established literature, and successful jailbreaks have been documented for every major model released to date. The research community consensus, reflected in both the CISA advisory and the OWASP Agentic Top 10, is that defenses which rely on model-level judgment are a valuable layer but cannot be the primary layer. The primary layer must consist of deterministic controls: capability restrictions that do not depend on the model recognizing an attack, human confirmation gates that cannot be bypassed through prompt manipulation, and execution environments that constrain what actions a compromised agent can take regardless of what it has been instructed to do.

Microsoft's April 2026 release of the Agent Governance Toolkit formalizes this architectural approach as deployable tooling rather than research direction [12]. The toolkit — open-source under the MIT license, available in Python, TypeScript, Rust, Go, and .NET — functions as a stateless policy engine that intercepts every agent action before execution with sub-millisecond latency. It is framework-agnostic, integrating with LangChain, CrewAI, Google ADK, and Microsoft's own agent frameworks, and is the first open-source toolkit designed to provide coverage of all ten OWASP Agentic Top 10 risks through deterministic policy enforcement [13]. The toolkit's release — alongside CISA's April 30 advisory and the concurrent Kill-Chain Canaries tracking research published at arXiv in March 2026 — signals that the industry has moved from characterizing the problem to building deployable infrastructure for managing it [14].

Recommendations

Immediate Actions

Any organization with AI agents deployed in production that process user-uploaded content, retrieve content from external URLs or email, or operate with access to file systems, credentials, or execution environments should conduct an emergency capability audit. The audit should answer four questions for

each deployed agent: What tools can this agent invoke? Which of those tools have external effects (network access, file writes, credential use, code execution)? Does the agent process content originating from outside the organization's control? And is human confirmation required before any high-consequence action? Any agent that can execute code, write files, or make outbound network requests while also processing externally sourced content should be treated as a promptware target and its tool permissions reviewed before the next operational cycle.

For organizations using AI coding assistants in development workflows — which the April 2026 VentureBeat investigation identified as the most actively exploited category — the immediate priority is reviewing repository-level permissions. Coding agents should not have write access to production branches or CI/CD pipeline configuration without explicit human gate approval. Environment variables and credentials should not be accessible to the agent's session by default; they should be provisioned per-task through short-lived secrets injection, in keeping with the CISA advisory's credential management guidance [5].

MCP server inventories should be audited immediately. Any MCP server whose provenance cannot be confirmed — including servers installed via community repositories, AI-generated tool lists, or developer dependency chains — should be disabled until reviewed. Tool description changes since initial installation should be treated as potential supply chain tampering. The supply chain attack vector documented by CyberArk's Full-Schema Poisoning research and the MCPTox benchmark established that attackers need not compromise the model to redirect its behavior; poisoning the tool definition visible to the model is sufficient [15][16].

Short-Term Mitigations

Human-in-the-loop confirmation should be implemented as a system-level control — enforced by the orchestration infrastructure, not by instructions to the model — for any agent action that has external effects. This is the single recommendation that appears in every authoritative guidance document published in 2026, from CISA's advisory to the OWASP Agentic Top 10 to Microsoft's Agent Governance Toolkit documentation, and it is the one control that cannot be bypassed through a successful prompt injection. An agent that receives a malicious instruction to send an email, delete a file, or execute a command cannot complete that action if the orchestration layer requires a human to approve the specific action before execution.

Network egress filtering should be applied at the infrastructure level for all agent execution environments. Agents should be permitted to make outbound connections only to a maintained allowlist of domains and IP ranges. The ZombAI demonstrations, the EchoLeak attack (CVE-2025-32711), and the Claude Computer Use C2 demonstration all required that the agent be able to reach attacker-

controlled internet infrastructure [9]. Network controls do not prevent injection; they degrade the attacker's ability to use a compromised agent as a C2 endpoint, data exfiltration channel, or malware download client.

Agent activity logging should capture the full content of every prompt delivered to the agent, every tool call and its parameters, every external HTTP request, and every action taken. Log storage must be out-of-band from systems the agent can access, so that a compromised agent cannot tamper with or delete its own audit trail. Microsoft Defender's integration with Microsoft 365 Copilot, which detects XPIA (Cross-Prompt Injection Attack) activity through behavioral correlation rather than signature matching, provides a reference architecture for behavioral monitoring that generalizes beyond the Microsoft ecosystem [17]. Organizations with agents on other platforms should implement equivalent behavioral telemetry through their existing SIEM and SOAR infrastructure.

For organizations prepared to invest in tooling, Microsoft's Agent Governance Toolkit provides a framework-agnostic, open-source implementation of the deterministic controls that the CISA advisory and OWASP frameworks recommend [12]. Deploying the toolkit's policy engine in front of agent actions enables sub-millisecond enforcement of capability restrictions, action allowlists, and human-confirmation gates without modifying the underlying model or agent framework. The toolkit's compliance module also supports evidence collection for EU AI Act obligations that become enforceable in August 2026 and Colorado AI Act obligations effective June 2026.

Strategic Considerations

The April 30 CISA advisory establishes a strategic inflection point: organizations that have not incorporated AI agents into their formal threat model now have an authoritative government basis for doing so. Security leaders should incorporate promptware and agentic C2 into their next cycle of threat modeling, red-team exercises, and security awareness training. The AgentDojo framework, used by NIST CAISI for empirical evaluation and available as open-source tooling, enables internal red teams to test agent deployments against documented injection attack patterns across simulated workspace, travel, Slack, and banking environments [11].

The CISA advisory's framing — that agentic AI governance is an extension of zero-trust architecture, not a separate discipline — provides the organizational alignment needed to address this threat without creating a parallel security program. Security teams that have implemented zero-trust network controls, endpoint detection, and identity governance for human users have most of the building blocks needed for agentic AI governance. The gap is primarily in applying those controls to non-human identities: agents need the same access lifecycle management (provisioning, de-provisioning, audit) and the same credential hygiene (short-lived secrets, no static long-lived API keys) that mature zero-trust programs already require for service accounts and cloud workloads.

Organizations should also begin planning for the multi-vendor coordination challenge that the April 2026 coding-agent incident illustrated. When a single prompt injection payload is effective against multiple AI coding assistants simultaneously, responsible disclosure and coordinated patching become a multi-vendor problem without established precedent. Security teams should identify which AI vendors they rely on, map those vendors to their published vulnerability disclosure programs, and establish direct security contacts — not just general support channels — before an incident rather than after.

CSA Resource Alignment

CSA's **MAESTRO (Multi-Agent Environment Security Threat and Risk Overview)** framework, published in February 2025 and updated in February 2026 with real-world CI/CD pipeline applications, provides the primary threat modeling structure for mapping promptware risks across the seven layers of agentic AI architecture [18][19]. MAESTRO classifies prompt injection as a Tier 1 threat at the agent orchestration layer and distinguishes between direct injection via the user interface and environmental injection via retrieved data — the dominant attack vector in the incidents documented in this note. The framework's least-capability guidance aligns directly with the CISA advisory's operational recommendations and provides CSA-standardized terminology for threat modeling artifacts that security teams can use in risk registers and architecture reviews.

The **CSA AI Controls Matrix (AICM)**, which extends the Cloud Controls Matrix with 18 AI-specific domains, addresses the promptware C2 threat surface through multiple control families. The Application and Interface Security (AIS) domain covers prompt injection and output handling controls directly. The Logging and Monitoring (LOG) domain governs the behavioral telemetry that makes detection of anomalous agent activity possible. The Identity and Access Management (IAM) domain provides the control objectives for non-human identity lifecycle management that the CISA advisory identifies as foundational to secure agent deployment. Organizations using AICM as their AI governance framework can map the recommendations in this note to specific AICM control identifiers for audit and evidence purposes.

OWASP's LLM Top 10 (2025) ranks prompt injection as LLM01 — the highest-priority vulnerability category across production AI deployments [20]. Cisco's 2026 State of AI Security report found prompt injection present in over 73% of production AI deployments assessed in security audits, reinforcing the empirical prevalence of this threat class [22]. The companion **OWASP Top 10 for Agentic Applications (December 2025)** addresses the agentic-specific dimensions of the same threat through ASI01 (Agent Goal Hijack), ASI06 (Memory Poisoning), and ASI07 (Insecure Inter-Agent Communication) [13]. Microsoft's Agent Governance Toolkit was designed against this OWASP taxonomy and provides organizations with direct evidence collection for each of the ten agentic risks.

MITRE ATLAS, updated in October 2025 with fourteen new techniques contributed by Zenity Labs covering AI agent and generative AI attack patterns, provides adversary-behavior mappings for Agent Context Poisoning (manipulating the agent's working context to persistently influence decisions across tasks) and Memory Manipulation (altering long-term memory so that injected instructions survive session boundaries) [21]. Organizations that run MITRE ATT&CK-based detection engineering programs can extend their existing detection logic to cover the analogous ATLAS techniques without building a separate AI-specific detection program from scratch.

References

[1] Oleg Brodt, Elad Feldman, Bruce Schneier, and Ben Nassi. "The Promptware Kill Chain: How Prompt Injections Gradually Evolved Into a Multistep Malware Delivery Mechanism." arXiv:2601.09625, January 2026.

[2] VentureBeat. "Three AI coding agents leaked secrets through a single prompt injection. One vendor's system card predicted it." VentureBeat, April 2026.

[3] Anthropic. "Mitigating the risk of prompt injections in browser use." Anthropic Research, November 2025.

[4] OpenAI. "Continuously hardening ChatGPT Atlas against prompt injection attacks." OpenAI, December 2025.

[5] CISA, NSA, ASD ACSC, CCCS, NCSC-NZ, NCSC-UK. "Careful Adoption of Agentic AI Services." CISA, April 30, 2026.

[6] VentureBeat. "AI coding agents breached: attackers targeted credentials, not models." VentureBeat, 2026.

[7] Kai Greshake, Sahar Abdelnabi, Shailesh Mishra, Christoph Endres, Thorsten Holz, and Mario Fritz. "Not what you've signed up for: Compromising Real-World LLM-Integrated Applications with Indirect Prompt Injection." arXiv:2302.12173, February 2023.

[8] OWASP GenAI Security Project. "LLM01:2025 Prompt Injection." OWASP, 2025.

[9] Cloud Security Alliance AI Safety Initiative. "Agent Commander: Promptware-Powered Command and Control Infrastructure for AI Agents." CSA Lab Space, March 17, 2026.

[10] Johann Rehberger. "Agent Commander: Promptware Powered C2 – Your Agent Works for Me Now." embracethered.com, March 2026.

[11] NIST Center for AI Standards and Innovation. "Technical Blog: Strengthening AI Agent Hijacking Evaluations." NIST, January 2025.

[12] Microsoft. "Introducing the Agent Governance Toolkit: Open-source runtime security for AI agents." Microsoft Open Source Blog, April 2, 2026.

[13] OWASP GenAI Security Project. "Top 10 for Agentic Applications for 2026." OWASP, December 2025.

[14] Anon et al. "Kill-Chain Canaries: Stage-Level Tracking of Prompt Injection Across Attack Surfaces and Model Safety Tiers." arXiv:2603.28013, March 2026.

[15] CyberArk. "Poison everywhere: No output from your MCP server is safe." CyberArk Threat Research, 2025.

[16] Anon et al. "MCPTox: A Benchmark for Tool Poisoning Attack on Real-World MCP Servers." arXiv:2508.14925, August 2025.

[17] VentureBeat. "Microsoft patched a Copilot Studio prompt injection. The data exfiltrated anyway." VentureBeat, 2026.

[18] Cloud Security Alliance. "Agentic AI Threat Modeling Framework: MAESTRO." CSA Blog, February 6, 2025.

[19] Cloud Security Alliance. "Applying MAESTRO to Real-World Agentic AI Threat Models: From Framework to CI/CD Pipeline." CSA Blog, February 11, 2026.

[20] OWASP GenAI Security Project. "OWASP Top 10 for Large Language Model Applications 2025." OWASP, 2025.

[21] MITRE. "MITRE ATLAS: Adversarial Threat Landscape for Artificial-Intelligence Systems." MITRE, updated October 2025.

[22] Cisco. "State of AI Security 2026." Cisco, 2026.