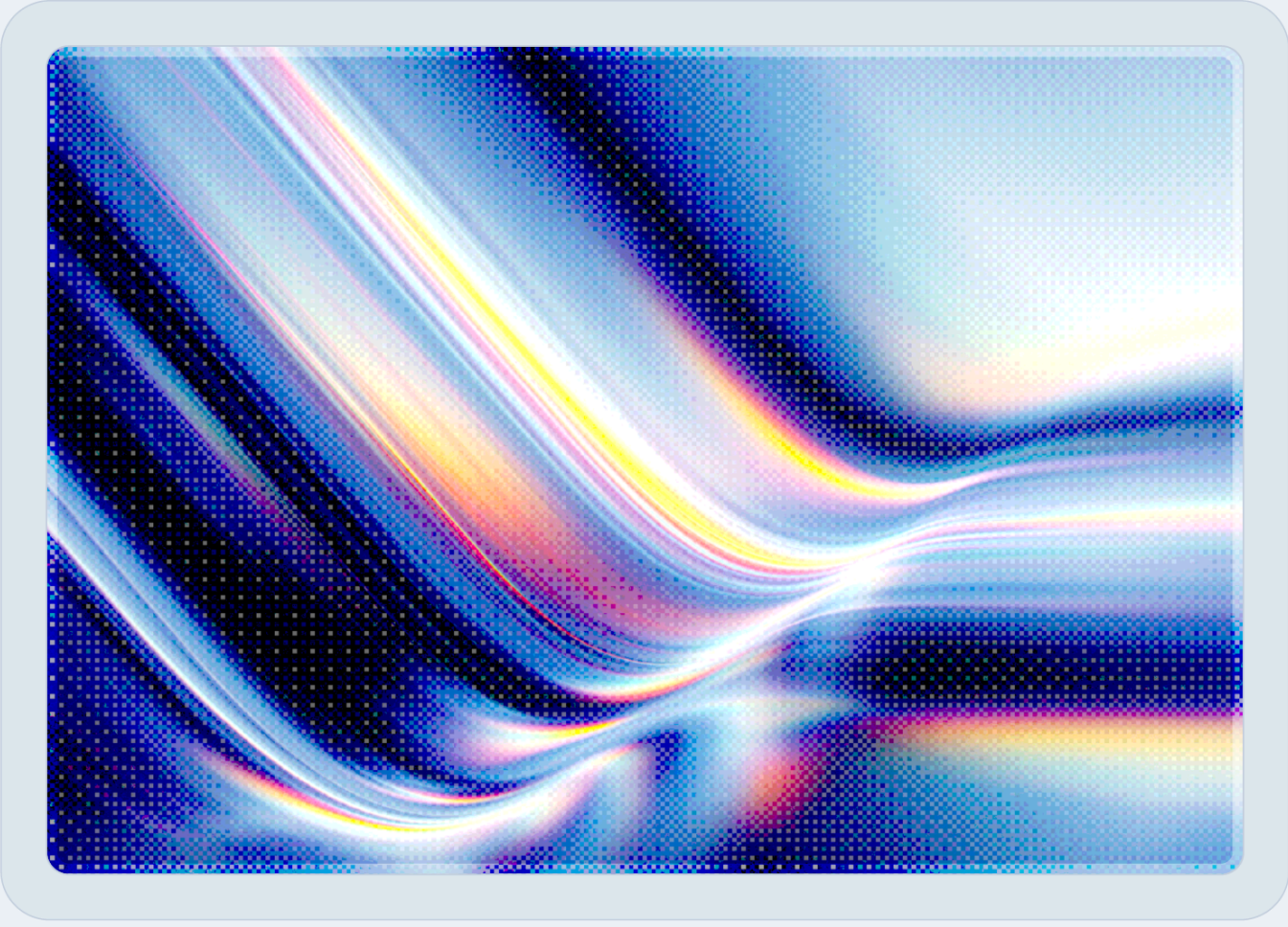
CSAI Foundation | Cloud Security Alliance

ChatGPhish: When the AI Assistant Becomes the Phishing Vector

2026-05-31

AI-assisted Rapid Research



CSAI

CSA cloud security alliance®

Key Takeaways

- Indirect prompt injection — the technique of planting executable instructions inside content that an AI assistant will later read — is now ranked the single highest-priority vulnerability class in the OWASP LLM Top 10 for 2025, and production exploits have escalated from theoretical research to patched CVEs with scores above 9.0 [1][2].
- ChatGPhish, disclosed May 2026, demonstrates that any attacker-controlled or attacker-influenced web page a user asks ChatGPT to summarize can become a phishing delivery surface: attacker-controlled links, spoofed alerts, and credential-harvesting QR codes render inside the trusted chatgpt.com interface without any browser compromise or email delivery required [3].
- At least six named attack techniques have been publicly disclosed in the past twelve months — EchoLeak (CVE-2025-32711, CVSS 9.3), HashJack, CometJacking, Reprompt, CamoLeak, and Comment and Control — each demonstrating a distinct mechanism through which AI assistants integrated into enterprise workflows can be weaponized as phishing or exfiltration channels [4][5][6][7][8][9].
- Traditional email security gateways, network perimeter controls, and conventional endpoint protection have limited or no visibility into several of these attack vectors: URL fragments never reach network inspection layers, image-based exfiltration tunnels through trusted infrastructure, and the AI's own Markdown renderer serves as the delivery mechanism — structural blindspots documented in specific disclosed exploits [5][7].
- Gartner projects that through 2029 more than half of all successful attacks against AI agents will use direct or indirect prompt injection as the initial access vector, and NIST's empirical research found novel agent-targeted injection attacks achieved an 81% task-hijacking success rate compared to 11% for conventional baseline attacks [10][11].

Background

Phishing has always exploited the gap between what a victim perceives as trustworthy and what is actually controlled by an attacker. Email spoofing exploited visual similarity in sender addresses; homograph attacks exploited Unicode rendering; HTML emails hid malicious URLs behind innocuous anchor text. In each case, the attack surface was the victim's own cognitive model of trust — what looks official is treated as official. The emergence of AI assistants integrated into enterprise workflows has opened a structurally new variant of this same exploitation pattern, one for which existing defenses were not designed.

When an employee asks an AI assistant to summarize a web page, draft a reply to an email, or review a document, the AI's context window becomes the conduit through which the task is executed. The AI does not experience the world through a browser with a URL bar displaying a green padlock or a familiar domain; it experiences the world as text and structured data. An attacker who can influence that text — through a crafted web page, a poisoned email, a malicious GitHub issue, or a URL fragment — can issue instructions to the AI that are indistinguishable from legitimate system context. This is the fundamental mechanism that OWASP's LLM Security Project has designated as the top-ranked vulnerability in large language model deployments, listed as LLM01:2025 — Prompt Injection — for the second consecutive edition [1].

The threat has matured with striking speed. In 2023, indirect prompt injection was primarily a research curiosity, documented in academic papers and proof-of-concept blogs; no production incidents at meaningful scale were publicly reported during this period. By 2025, threat actors were executing real-world exploits against commercially deployed AI features in Microsoft 365, GitHub, and Chrome-integrated AI browsers. OpenAI disclosed in its October 2025 threat disruption report that it had disrupted over 40 actor networks abusing its systems to generate phishing lures, and that state-affiliated groups were actively using ChatGPT to streamline spear-phishing personalization [12]. NIST characterized indirect prompt injection as among the most critical vulnerabilities in generative AI systems [27], and the UK NCSC warned in December 2025 that prompt injection breaches could ultimately exceed SQL injection in scope and scale — describing the problem as one that "may never be properly mitigated" in the way SQL injection was [28]. The transition from research curiosity — when academic proof-of-concept papers first appeared in early 2023 — to enterprise threat vector marked by zero-click production exploitation (EchoLeak, mid-2025) took approximately twenty-four months.

Security Analysis

The Mechanics of AI-Mediated Phishing

To understand the ChatGPhish family of attacks, it is necessary to understand what changes when a phishing campaign targets an AI assistant rather than a human inbox. In a conventional phishing attack, the attacker must convince a human to click a link, open a file, or enter credentials — actions that security awareness training, URL reputation services, and browser warnings can all intercept. In an AI-mediated attack, the victim never directly interacts with the malicious content. Instead, the victim asks their AI assistant to perform a routine task, and the AI — acting faithfully on instructions it has been given — performs the phishing action on the victim's behalf.

The attacker's payload is no longer styled HTML designed to look like a bank login page. It is a set of natural-language instructions, embedded invisibly or semi-visibly in a document, web page, email, or URL, that the AI will execute when it processes that content. The instructions might direct the AI to render attacker-controlled links inside its own trusted interface, to exfiltrate data to an external server via an image-loading request, to rewrite a draft email to include malicious content, or to ask the user to re-authenticate through a spoofed form. Each of these outcomes represents a complete phishing delivery without a single malicious pixel reaching the victim's own eyes in a way they would recognize as suspicious.

Named Attack Techniques: A Comparative Overview

The following table summarizes the six most significant publicly disclosed attack techniques as of May 2026, all of which exploit variants of this core mechanism.

Attack	Disclosed	Affected Products	Researcher	CVE / Score	Attack Mechanism
EchoLeak	June 2025	Microsoft 365 Copilot	Aim Security	CVE-2025-32711 / CVSS 9.3	Zero-click: crafted email causes Copilot to exfiltrate inbox content via auto-fetched image request

Attack	Disclosed	Affected Products	Researcher	CVE / Score	Attack Mechanism
HashJack	November 2025	Copilot in Edge, Gemini in Chrome, Perplexity Comet	Cato Networks CTRL	None assigned	Malicious instructions embedded in URL fragment (`#`); client-parsed, server-invisible
CometJacking	October 2025	Perplexity Comet	LayerX Security	None assigned	Single-click: malicious URL query parameters auto-execute against victim's connected Gmail/Calendar
Reprompt	January 2026 (patched)	Microsoft Copilot Personal	Varonis Threat Labs	None assigned	Phishing link with malicious `q` parameter auto-executes prompt, exfiltrating history, files, location
CamoLeak	August 2025 (patched)	GitHub Copilot Chat	Omer Mayraz / HackerOne	CVE-2025-59145 / CVSS 9.6	Invisible Markdown in PR descriptions; exfiltration via pre-signed GitHub Camo proxy image requests

Attack	Disclosed	Affected Products	Researcher	CVE / Score	Attack Mechanism
ChatGPhish	May 2026	ChatGPT (web summarization)	Permiso Security	None assigned	Web page summarization renders attacker links, alerts, QR codes, and remote images inside chatgpt.com

Sources: [1][3][4][5][6][7][8][9]

EchoLeak: The First Zero-Click Production Exploit

EchoLeak, assigned CVE-2025-32711 with a CVSS score of 9.3, is significant not merely because of its severity score but because it was the first publicly confirmed zero-click indirect prompt injection exploit to affect a widely deployed enterprise product in production use. Disclosed by Aim Security in June 2025, the vulnerability required no user interaction beyond ordinary Copilot usage: an attacker sent a single crafted email to the victim's Outlook inbox, and when the victim used Microsoft 365 Copilot to summarize their inbox — a routine task for which the product was designed — Copilot ingested the malicious instructions and silently exfiltrated OneDrive files, SharePoint content, Teams messages, and chat logs to an attacker-controlled server [4].

The attack chained four distinct bypasses. First, the injected instructions were framed as addressed to a human rather than an AI, defeating Microsoft's XPIA (Cross Prompt Injection Attempt) classifier, which was trained to detect instructions addressed to AI systems. Second, a reference-style Markdown link syntax was used to smuggle URLs past Microsoft's link-redaction filter. Third, an auto-fetched image request — permitted under the Microsoft Teams content security policy — served as the exfiltration channel, encoding stolen data in a URL parameter that loaded a 1x1 transparent pixel from the attacker's server. Fourth, the exfiltration traffic traversed Microsoft's own Teams CSP proxy, making it indistinguishable from legitimate image loading. Microsoft patched the vulnerability server-side in June 2025; no customer action was required [4].

The architectural lesson of EchoLeak is that content security policies, link-redaction filters, and classifier-based injection detection can each individually be bypassed when an attacker has the ability to tune their payload against the specific defense in use. The June 2025 Aim Security paper noted that

their adaptive payload bypassed all four defensive layers in sequence — a finding that aligns with late 2025 research testing 12 published defenses against adaptive attackers, which found that all were bypassed with greater than 90% success rates [13].

ChatGPhish and the Trusted Interface Problem

ChatGPhish, disclosed by Permiso Security in May 2026, exploits a different architectural assumption: that the trusted visual context of a known web application conveys authentic information to the user. When a user views content inside chatgpt.com, they are operating in a context they associate with a verified, signed-in session rather than an arbitrary web page. ChatGPhish weaponizes this trust by causing attacker-controlled content to render inside that trusted interface through the AI's own Markdown rendering engine [3].

The mechanism is straightforward. An attacker creates or compromises a web page containing injected Markdown instructions alongside normal-looking content. When a user asks ChatGPT to summarize that page, the AI processes both the legitimate content and the injected instructions, rendering attacker-controlled links as clickable hyperlinks, displaying fake security alerts formatted as system messages, embedding credential-harvesting QR codes, and loading remote images from attacker-controlled servers — the last of which silently leaks the victim's IP address, User-Agent string, and Referer header to the attacker without any further interaction [3]. No special access, account compromise, or prior relationship with the victim is required: any attacker-influenced page that an employee might ask an AI to summarize becomes a potential delivery surface.

The attack exploits ChatGPT's implicit trust in Markdown links and images retrieved from external pages during summarization. Unlike EchoLeak, which exploited a specific set of Microsoft-specific bypass chains, ChatGPhish exploits a more fundamental behavior in how language models process and render mixed content. This distinction matters for remediation: patching a specific CSP bypass leaves the underlying rendering trust model intact. As of the date of this publication, OpenAI has not confirmed a patch for the ChatGPhish vulnerability; readers should monitor OpenAI security advisories for updates.

HashJack and the Perimeter Blind Spot

HashJack, disclosed by Cato Networks CTRL in November 2025, reveals a structural incompatibility between how networks inspect traffic and how AI browser assistants process URLs [5]. URL fragments — the portion of a URL following the `#` character — are processed entirely by the client and never transmitted to the web server. Web servers are therefore unable to log, inspect, or respond to fragment

content. Conventional network security controls — proxies, firewalls, secure web gateways, and content inspection services — similarly cannot see fragment content because it never traverses the network as a query to the destination.

AI browser assistants integrated into Chrome and Edge, however, receive the full URL as supplied by the user or an attacker, including the fragment. Cato's researchers demonstrated that malicious instructions placed in the URL fragment of an otherwise legitimate website were executed by Microsoft Copilot in Edge and Google Gemini in Chrome when those assistants processed the full URL. Six attack scenarios were documented, including callback phishing (injecting fake support phone numbers), credential harvesting, and malware download guidance [5]. The affected products covered two of the three largest browser-integrated AI deployments. Google's public response classified the behavior as "intended" and "social engineering" rather than a security vulnerability, declining to apply a fix [5]. This vendor response is itself a signal: when a major platform vendor classifies a well-documented attack scenario as intended behavior, the defense burden falls entirely on enterprise security teams.

Comment and Control: Agentic Environments as Attack Surfaces

The April 2026 "Comment and Control" disclosure by researchers at Wyze Labs and Johns Hopkins extends the prompt injection threat surface from chat-based AI assistants into agentic CI/CD environments [9]. The researchers demonstrated that GitHub pull request titles, issue bodies, and issue comments — all of which are processed by AI coding agents triggered via GitHub Actions — could contain injected instructions that caused those agents to exfiltrate API keys and GitHub Actions secrets back through PR comments, issue comments, or git commits entirely within GitHub's own infrastructure. Working payloads were demonstrated against Claude Code, Gemini CLI Action, and GitHub Copilot Agent.

The significance of this finding extends beyond the three specific products. GitHub Actions is a standard deployment target for agentic AI workflows, and the pattern of triggering agents via repository events is broadly replicated across enterprise software development pipelines. The "Comment and Control" nomenclature deliberately invokes command-and-control terminology to highlight that the attack transforms a routine development workflow artifact — an issue comment — into an attack instruction that an agent executes with its full set of granted permissions. Unlike reactive prompt injection attacks that require a victim to initiate a query, this is a proactive model: the attacker opens a PR or files an issue, GitHub automation does the rest.

The Statistical Context

Industry data from multiple independent sources has converged on a consistent characterization of the threat's trajectory. HackerOne's 2025 Hacker-Powered Security Report documented a 540% year-over-year increase in valid prompt injection vulnerability reports submitted by security researchers [14]. Google detected a 32% relative increase in malicious prompt injection activity between November 2025 and February 2026 [15]. AttackEval research (arxiv:2604.03598) found that obfuscation combined with emotional manipulation achieved a 97.6% attack success rate against defended language models [16]. NIST's empirical work with the AgentDojo-Inspect toolchain found novel agent-targeted attacks achieving 81% task-hijacking success compared to 11% for conventional baseline attacks [11]. KnowBe4's 2025 Phishing Threat Trends Report found that 82.6% of phishing emails analyzed between September 2024 and February 2025 contained AI-generated content [17].

These figures must be interpreted carefully: they measure different things — researcher-reported vulnerabilities, detected malicious activity, laboratory success rates, and production email classification — and cannot be simply combined. Taken together, however, they consistently indicate that the combination of AI-assisted phishing content generation and AI-assistant exploitation as a delivery mechanism has moved from marginal to mainstream within the threat landscape over the past twelve months.

Embedding and Exfiltration Techniques

A shared technical vocabulary has emerged across the disclosed attacks, centering on the methods attackers use to hide injection payloads from human reviewers while ensuring AI systems read and execute them. The most documented technique is invisible or near-invisible text: white-colored text on a white background, or font sizes below 1 point, which are machine-readable by AI systems but invisible to human reviewers. Research published in July 2025 tested this technique against academic manuscript review workflows and found a 98.6% success rate across tested models, with LLM-generated peer reviews matching injected instructions at a 90% agreement rate [18].

URL fragment injection (HashJack) and reference-style Markdown link smuggling (EchoLeak) represent protocol-layer hiding techniques that exploit specific behaviors in how AI systems parse URLs and format text. Metadata-field injection — placing instructions in EXIF data, PDF document properties, or DOCX core properties — exploits the common pattern of AI assistants reading full document context when summarizing files. Auto-fetched image abuse turns the AI's routine content rendering into an exfiltration beacon by encoding stolen data as URL parameters in image requests that the AI loads automatically as part of display rendering. Each of these techniques exploits the same fundamental asymmetry: the AI reads and acts on content that human reviewers do not see [4][5][16].

Recommendations

Immediate Actions

Security teams should audit which AI assistants in their environment have web summarization, document reading, or inbox-connected features enabled, as these capabilities represent the primary surfaces exploited in ChatGPhish, EchoLeak, and Reprompt-style attacks. For each such feature, teams should verify whether the vendor has applied patches addressing known injection vulnerabilities, and whether the feature's threat model has been reviewed against the attack techniques documented above. Microsoft's patches for CVE-2025-32711 (EchoLeak) and the Reprompt vulnerability were applied server-side and required no customer action [7], but organizations should confirm their environments are running current versions of all AI-integrated products. Where GitHub Actions or similar CI/CD automation triggers AI agent workflows, teams should audit the permissions granted to those agents and verify that contributed code, PR titles, issue bodies, and comments from external contributors are treated as untrusted input rather than trusted instruction sources.

Phishing-resistant MFA should be enforced for all accounts that can access AI assistants with connections to sensitive organizational data, including email, file storage, calendars, and code repositories. CISA's advisory AA23-320A recommends FIDO2 hardware tokens or device-bound passkeys as the baseline for phishing-resistant authentication, and specifically recommends out-of-band identity verification for all password resets and MFA changes — controls that remain relevant even when the phishing vector is an AI intermediary rather than a direct credential form [19]. Phishing resistance is not a complete defense against prompt injection, but it eliminates the credential-harvesting outcome that several of the documented attacks specifically target.

Short-Term Mitigations

Organizations should implement a least-privilege posture for AI agent integrations, granting each agent only the specific OAuth scopes, API permissions, and tool capabilities required for its defined function. CISA's Agentic AI Security Guide and CSA's MAESTRO framework both identify excessive agency — AI agents with broader permissions than their tasks require — as a primary amplifier of prompt injection risk [20][21]. An AI assistant that can read email but not send it, or that can read files but not post to external URLs, limits the blast radius of any successful injection. A read-only research agent that also holds email-send permission represents precisely the excessive agency that OWASP LLM01:2025 identifies as a primary amplifier of injection risk [1].

Human-in-the-loop gates should be implemented for high-consequence AI agent actions, including sending external communications, accessing credential stores, modifying repository contents, and initiating outbound API calls. This architectural control is the closest available mitigation to the structural problem that indirect prompt injection exploits: because AI systems cannot reliably distinguish legitimate instructions from injected ones when both arrive in the same context channel, a human approval gate on consequential actions interrupts the attack chain before damage occurs. CISA's guidance is explicit that agents must not be delegated authority to determine which of their own actions need human sign-off — that boundary must be pre-encoded in system design rather than left to the agent's runtime judgment [20].

Browser-integrated AI features — particularly those marketed as ambient AI assistants that can see and act on everything a user views — should be subject to the same security review as any third-party software integration, including assessment against the HashJack and CometJacking attack models. Organizations that cannot complete that assessment before deployment may find December 2025 Gartner guidance relevant: Gartner issued guidance recommending that CISOs block use of AI browsers entirely pending adequate security evaluation [22].

Strategic Considerations

The deeper strategic challenge these attacks expose is that AI assistants create a new category of insider threat that does not involve a malicious or negligent human. A fully patched, security-aware employee using a fully patched AI assistant can become the source of a credential theft or data exfiltration incident through no action more suspicious than asking their assistant to summarize a web page. This shifts the security model fundamentally: defenses that focus on identifying malicious human behavior, anomalous file access, or suspicious network traffic may not fire on the behavior of a legitimate AI assistant faithfully executing injected instructions on behalf of a legitimate user.

Security teams should update their threat models, detection rules, and incident response playbooks to account for AI-mediated phishing scenarios. MITRE ATLAS v5.1.0 and v5.4.0 provide a documented technique taxonomy covering prompt injection (AML.T0051), jailbreak (AML.T0054), AI agent tool poisoning (AML.T0110), and the newly added "Publish Poisoned AI Agent Tool" for supply chain delivery scenarios [23]. Red team exercises should incorporate indirect prompt injection as a primary initial access technique, using the NIST AI 100-2 E2025 adversarial ML taxonomy and the AgentDojo-Inspect toolchain, which provides 97 injection tasks across 629 test cases as a structured evaluation starting point [11].

Runtime security tooling specifically designed for LLM environments should be evaluated as a complement to traditional endpoint and network controls. Commercial platforms including Lakera Guard (Check Point), Prisma AIRS (Palo Alto Networks), and Thales AI Security Fabric each provide prompt

injection detection at the application layer; open-source options including NVIDIA NeMo Guardrails, LLM Guard, and Garak provide accessible entry points for organizations building evaluation capability. PromptArmor's LLM-based detection has claimed sub-1% false positive and false negative rates on AgentDojo benchmarks in ICLR 2026 presentation materials [24].

CSA Resource Alignment

This research note connects directly to several active CSA frameworks and programs relevant to enterprise AI deployment security. CSA's MAESTRO framework (Multi-Agent Environment Security Threat and Risk Overview) provides the threat modeling foundation most directly applicable to the agentic scenarios described above — specifically HashJack, CometJacking, Comment and Control, and the EchoLeak attack chain, each of which exploits the trust relationship between an AI agent and the data sources it processes. MAESTRO's agent-specific threat taxonomy addresses lateral movement through multi-agent architectures, which the Promptware Kill Chain research has documented as increasing in frequency across the 2023-2026 incident corpus [25].

The AI Controls Matrix (AICM), CSA's control framework for AI system deployments and a superset of the Cloud Controls Matrix, maps directly to the least-privilege, human oversight, input validation, and output monitoring controls recommended in this note. Practitioners deploying AI assistants with enterprise data integrations should map their control environment against AICM categories covering data access governance, agent permission scoping, and content trust boundary enforcement. CSA's STAR (Security, Trust, Assurance, and Risk) certification program provides a third-party assurance mechanism for AI service providers' security claims, and organizations selecting AI assistant vendors should request STAR certification or equivalent independent attestation as a procurement condition.

CSA's Zero Trust Architecture guidance also applies directly to the authentication and authorization recommendations in this note. The phishing-resistant MFA, least-privilege OAuth scoping, and short-lived credential requirements articulated above are each instantiations of Zero Trust principles — verify explicitly, use least privilege, assume breach — applied to the AI integration layer. The practical implication is that AI assistants with access to sensitive organizational data should be treated as privileged principals in the Zero Trust model, subject to the same continuous verification and access control rigor as privileged human users.

Readers seeking additional context should consult CSA's prior research notes addressing adjacent topics: "Institutionalizing AI Safety: CISA's Agentic Guide and CAISI Agreements" [20], "The AI Agent Governance Gap: What CISOs Need Now" [21], and "Prompt Injection in AI-Powered GitHub Actions" [26], each of which provides expanded technical and governance treatment of themes introduced here.

References

[1] OWASP Gen AI Security Project. "LLM01:2025 Prompt Injection." OWASP, 2025.

[2] OWASP. "OWASP Top 10 for LLM Applications v2025 (PDF)." OWASP, 2025.

[3] The Hacker News. "ChatGPhish Vulnerability Turns ChatGPT Into a Phishing Vector." The Hacker News, May 2026.

[4] Aim Security / Checkmarx. "EchoLeak: CVE-2025-32711 – Zero-Click Copilot Vulnerability." Checkmarx, 2025. See also: Aim Security paper at arxiv.org/abs/2509.10540.

[5] Cato Networks CTRL. "HashJack: First Known Indirect Prompt Injection via URL Fragments." Cato Networks, November 2025.

[6] LayerX Security. "CometJacking: How One Click Can Turn Perplexity's Comet AI Browser Against You." LayerX, October 2025.

[7] Varonis Threat Labs. "Reprompt: Single-Click Microsoft Copilot Exploit." Varonis, January 2026.

[8] Dark Reading. "GitHub Copilot CamoLeak Attack Exfiltrates Data (CVE-2025-59145)." Dark Reading, 2025.

[9] Aonan Guan, Zhengyu Liu, Gavin Zhong. "Comment and Control: Prompt Injection Credential Theft via Claude Code, Gemini CLI, GitHub Copilot." Personal research disclosure, April 2026.

[10] Astrix Security. "Beyond the Prompt: Gartner on the Future of AI Agent Security." Astrix Security, 2025. (Gartner strategic planning assumption cited: through 2029, over 50% of successful attacks against AI agents will use direct or indirect prompt injection.)

[11] NIST. "Technical Blog: Strengthening AI Agent Hijacking Evaluations." NIST, January 2025.

[12] OpenAI. "Disrupting Malicious Uses of AI – October 2025." OpenAI, October 2025.

[13] Nasr et al. "The Attacker Moves Second: Stronger Adaptive Attacks Bypass Defenses Against LLM Jailbreaks and Prompt Injections." arXiv:2510.09023, October 2025.

[14] HackerOne. "2025 Hacker-Powered Security Report." HackerOne, 2025. (540% year-over-year increase in prompt injection reports.)

[15] Help Net Security. "Indirect Prompt Injection Is Taking Hold in the Wild." Help Net Security, April 2026. (32% increase in Google-detected malicious prompt injection activity, Nov 2025–Feb 2026.)

[16] AttackEval Research Team. "AttackEval: A Systematic Empirical Study of Prompt Injection Attack Effectiveness Against Large Language Models." arXiv:2604.03598, April 2026.

[17] KnowBe4. "2025 Phishing Threat Trends Report." KnowBe4, 2025. (82.6% of phishing emails contained AI-generated content, Sept 2024–Feb 2025.)

[18] Anonymous authors. "Hidden Prompts in Manuscripts Exploit AI-Assisted Peer Review." arXiv:2507.06185, July 2025.

[19] CISA. "Implementing Phishing-Resistant MFA (AA23-320A)." CISA, 2023.

[20] CSA Labs. "Institutionalizing AI Safety: CISA's Agentic Guide and CAISI Agreements." Cloud Security Alliance, 2026.

[21] CSA Labs. "The AI Agent Governance Gap: What CISOs Need Now." Cloud Security Alliance, April 2026.

[22] Wiz. "Agentic Browser Security: 2025 Year-End Review." Wiz, 2025. (Gartner December 2025 guidance to block AI browsers cited.)

[23] MITRE. "ATLAS – Adversarial Threat Landscape for Artificial-Intelligence Systems." MITRE, v5.4.0, February 2026.

[24] PromptArmor / ICLR 2026. Referenced in: Vectra AI. "Prompt Injection: Detection and Defense." Vectra AI, 2026.

[25] McHugh et al. "Prompt Injection 2.0: Hybrid AI Threats." arXiv:2507.13169, July 2025. (Promptware Kill Chain lateral movement frequency data.)

[26] CSA Labs. "Prompt Injection in AI-Powered GitHub Actions." Cloud Security Alliance, May 2026.

[27] IBM. "How AI Can Be Hacked With Prompt Injection: NIST Report." IBM Think, 2025.

[28] UK NCSC. "Prompt Injection Is Not SQL Injection (It May Be Worse)." NCSC Blog, December 2025.