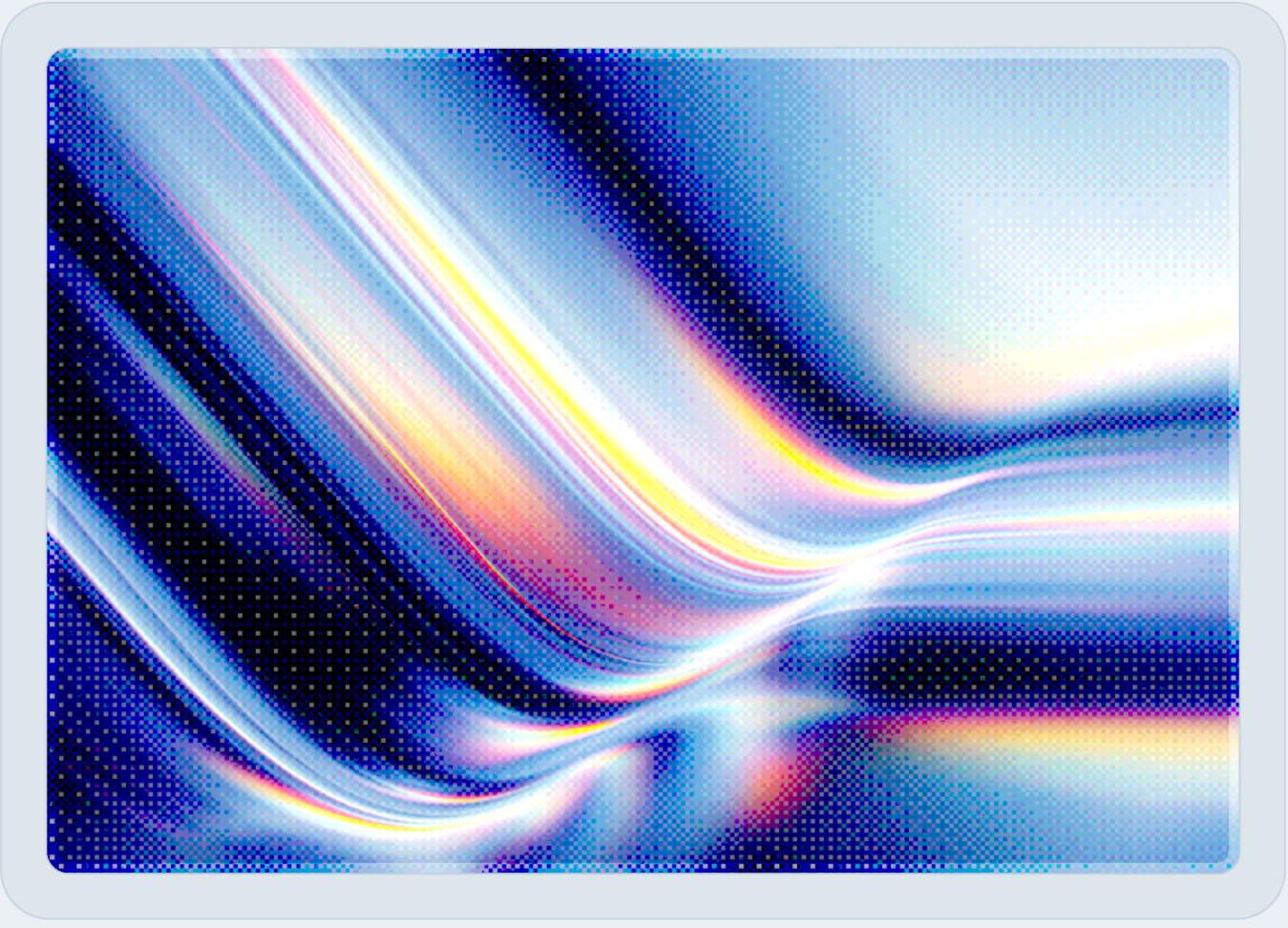
CSAI Foundation | Cloud Security Alliance

WriteOut: Session Isolation Flaw in Writer AI Previews

One-Click Cross-Tenant Account Takeover via Leaked Session Tokens

2026-07-08

AI-assisted Rapid Research



CSAI | CSA cloud security alliance®

Key Takeaways

Security researchers at Sand Security disclosed a now-patched, critical session isolation vulnerability in Writer, an enterprise generative AI platform, codenamed WriteOut, that allowed an unauthenticated outsider to take over a Writer customer's account with nothing more than a single clicked link — researchers characterized the flaw as capable of compromising "any Writer AI organization" [1]. The flaw abused Writer's live agent preview feature: when an attacker shared a preview link for an agent they built, and a logged-in Writer user opened it, the victim's browser attached its Writer session cookie to the request, and the preview proxy forwarded that cookie into the attacker's own sandbox, where code the attacker controlled could read and exfiltrate the token [1]. Because the attacker and victim did not need to share an organization, the vulnerability bypassed tenant isolation — a failure of the boundary that multi-tenant SaaS platforms are expected to maintain between one customer's data and another's [1]. Once an attacker replayed the stolen session token, they could access the victim's private chats, documents, agent configurations, connectors, and large language model credentials, and in some cases escalate to administrative control over the account [1]. Writer's guardrails were also bypassed in the process: input-side filters checked the literal instruction an agent builder submitted rather than the runtime behavior that instruction produced, so researchers evaded detection simply by telling the agent to fetch and execute a remote script instead of embedding the exploit logic directly in the prompt [1]. Writer remediated the issue by no longer forwarding session cookies into sandbox previews at all, instead isolating preview execution to a separate origin [1].

Background

Writer is a San Francisco-based enterprise generative AI company, founded in 2020, that raised $200 million in Series C funding in late 2024 at a $1.9 billion valuation, with customers reported to include Accenture, Intuit, Salesforce, Uber, and Vanguard [2]. Its platform centers on the Palmyra family of large language models alongside a no-code and codeful agent-building environment called Writer Framework, which lets customers construct custom AI agents and workflows and, notably, preview those agents in a live, running state before publishing them [1][2]. That live preview capability — allowing a builder to click a link and interact with a functioning version of an agent inside a hosted sandbox — is precisely the feature

WriteOut exploited, and it reflects a risk that is not unique to Writer: the same infrastructure that makes agent development fast and collaborative can also create an under-scrutinized bridge between a user's authenticated session and code an untrusted third party controls.

Sand Security's research team found that Writer's preview architecture treated the preview proxy as a trusted intermediary without adequately isolating the session state it carried. An attacker needed no prior access to a victim's organization, no compromised credentials, and no social engineering beyond convincing a target to click a single link — a bar so low that Sand Security characterized the flaw as capable of taking over "any Writer AI organization inside industry-leading enterprises" with nothing more than a URL [1]. Sand Security disclosed the vulnerability to Writer through a responsible disclosure process, and Writer patched it before the research became public; the fix arrived in the form of preventing the user's session cookie from being forwarded into preview sandboxes and moving those sandboxes to an isolated origin, closing the specific path the exploit relied on [1]. The Hacker News published an account of the disclosure on July 7, 2026 [1].

WriteOut appears to be an early instance of a broader vulnerability pattern in AI agent platforms, rather than an isolated design mistake — one where a legitimate feature for testing, previewing, or externally connecting an agent inadvertently becomes a channel for forwarding a user's live session credentials to an untrusted destination. A comparable flaw, tracked as CVE-2026-55431, was recently identified in Coder CLI, a developer workspace tool, where a malicious workspace template could define an "external app" URL containing a `$SESSION_TOKEN` placeholder; the CLI performed a literal string substitution of the caller's real session token into that URL before opening it, letting a template author who controlled the destination steal the token the moment a victim ran the command [3]. The two vulnerabilities differ in mechanism — Writer's flaw stemmed from a preview proxy that intentionally bridged sandboxed execution with session state, while the Coder flaw stemmed from unvalidated URL substitution with no scheme or destination allowlisting — but both share the same underlying failure mode: platforms that let one user's constructed artifact (an agent preview, a workspace template) execute in a context that can reach another user's live session token, without a hard boundary preventing that token from crossing into untrusted code [1][3].

Security Analysis

The WriteOut attack chain worked in four steps. First, the attacker created an agent within their own Writer account and enabled its live preview feature, generating a shareable preview link. Second, the attacker distributed that link to a target — through email, a support channel, a shared document, or any other means that would prompt a legitimate, already-authenticated Writer user to click it. Third, when the victim opened the link in a browser where they were already signed in to Writer, the browser's normal

cookie-attachment behavior sent the victim's Writer session cookie along with the request, and Writer's preview proxy — rather than stripping that cookie or scoping it away from the attacker's sandbox — forwarded it into the running preview environment the attacker controlled. Fourth, code inside that sandbox, written by the attacker, read the forwarded cookie value and transmitted it to an attacker-controlled destination, after which the attacker could replay the token directly against Writer's platform and operate as the victim [1].

What makes this attack class particularly consequential for enterprise AI platforms is the scope of access a Writer session token conferred. Because Writer sessions are not scoped narrowly to a single low-sensitivity action, a hijacked token gave the attacker the same access the legitimate user had: private chat histories, stored documents, the configuration of any agents the victim had built, connected third-party integrations, and — critically for an AI platform — credentials and API keys used to authenticate to underlying large language models [1]. Depending on the victim's role within their organization, that access could extend to administrative functions, meaning a single clicked link from an external, unaffiliated attacker could, in the worst case, hand over effective control of an entire customer organization's Writer deployment. This is what distinguishes WriteOut from an ordinary account-compromise bug: the attacker and victim were never required to belong to the same tenant, which means the vulnerability sat inside the platform's multi-tenant isolation boundary rather than within any single customer's own configuration mistake.

The guardrail bypass Sand Security documented is worth examining on its own, because it illustrates a failure mode specific to AI agent platforms rather than traditional web applications. Writer had implemented input-side filtering intended to prevent agent builders from submitting obviously malicious instructions, such as directly asking an agent to read environment variables or execute arbitrary code. That filtering evaluated the instruction text itself, not the behavior the instruction would cause once executed. Sand Security's researchers found they could sidestep the filter entirely by instructing the agent to fetch and run a remote script — a request that reads, on its face, as an unremarkable "download and run" action rather than as an attack. The actual exploit logic never appeared in the prompt the guardrail inspected; it lived in the remote script fetched at runtime, invisible to any filter that only examines what a user typed rather than what code ultimately executes. This gap between instruction-level filtering and runtime behavioral monitoring illustrates a structural weakness worth watching across agent platforms generally: guardrails built to catch malicious prompts may be blind to attacks that defer their payload to a later, dynamically fetched stage.

It is also useful to distinguish what WriteOut demonstrates from what it does not. Sand Security disclosed the flaw responsibly, and there is no public reporting indicating it was exploited against real customers before Writer's fix shipped; this was proof-of-concept research rather than an observed in-the-wild campaign. The vulnerability also required a victim to be already authenticated to Writer and to click a link — it did not bypass Writer's authentication mechanism itself, and a user with no active Writer

session was not directly exposed by this specific technique. The risk nonetheless warrants serious attention because the attack required no privileged access, no malware, and no social engineering more sophisticated than a plausible-looking link, and because Writer's guardrail bypass shows that content filtering alone is an insufficient control for platforms that let users build and run agents that fetch and execute external code.

Recommendations

Immediate Actions

Organizations using Writer should confirm with their account team that the session-isolation fix has been fully deployed across their tenant and should review Writer account activity logs for any anomalous session activity, administrative actions, or credential access originating from unfamiliar preview or agent-sharing links in the period before the patch was confirmed. Security teams should treat any previously shared or clicked Writer agent preview link from an external or unverified source as a potential indicator of compromise and rotate any credentials, connector tokens, or LLM API keys that were accessible to the affected account. More broadly, security teams responsible for any AI agent-building or agent-orchestration platform in active use — not limited to Writer — should ask the vendor directly whether preview, testing, or "try it live" features forward any form of session credential, cookie, or token into sandboxed or externally shared execution contexts, since WriteOut demonstrates this is not a theoretical concern.

Short-Term Mitigations

Enterprises should apply least-privilege scoping to any AI platform account capable of building or previewing agents, limiting which connectors, credentials, and administrative functions are reachable from an individual user's session, so that a single compromised token cannot cascade into organization-wide exposure. Security teams evaluating or renewing contracts with AI agent-building platforms should add specific procurement and assessment questions about preview-sandbox isolation, session-token scoping, and whether guardrails inspect runtime behavior in addition to submitted instructions, using CSA's SaaS Security Capability Framework session-management and identity controls as a baseline for what to require [4]. Organizations should also train employees who build or test AI agents to treat unsolicited preview links — even ones that appear to originate from internal colleagues or familiar vendors — with the same caution applied to unsolicited login links or file-sharing invitations, since this attack class relies entirely on a legitimate-seeming click rather than on any technical compromise.

Strategic Considerations

WriteOut is a useful case study in why the shared responsibility model for AI SaaS platforms needs to explicitly address agent-preview and agent-sandbox features as a distinct attack surface, not as a natural extension of ordinary web application testing tools. Vendors building agent-development platforms should architect preview and testing sandboxes as fully isolated origins from the outset, with no path for a session credential to cross from the builder's authenticated context into code supplied by another user, rather than retrofitting that isolation after a disclosure. The guardrail bypass in this case also points to a structural gap that will recur across agent platforms broadly: instruction-level content filtering cannot substitute for runtime behavioral monitoring of what an agent actually does once it fetches and executes external code, and platforms that allow agents to make outbound network calls should assume that filtering the initial prompt is necessary but not sufficient. As enterprises increasingly rely on multiple agent-building platforms simultaneously, CSA assesses that this pattern of preview-and-session-forwarding vulnerabilities is likely to recur across vendors, and security and procurement teams should build assessment processes that specifically probe for it rather than treating each disclosure as an isolated incident.

CSA Resource Alignment

CSA's SaaS Security Capability Framework (SSCF) speaks directly to the control gap at the heart of WriteOut: the framework's Identity and Access Management domain — the largest of its six domains by control count, with 21 controls — explicitly defines customer-facing expectations around session management, including session blocklisting for revocation and default inactive session timeouts, a category of control that, had it been extended to preview-sandbox contexts, would have reduced the exploitable window for a forwarded session cookie, though the root-cause fix Writer ultimately shipped was architectural — origin isolation for preview execution — rather than session-management controls alone [4]. Because WriteOut is fundamentally a session-handling and tenant-isolation failure inside a SaaS platform rather than a model-behavior problem, the SSCF's customer-facing, vendor-agnostic session controls are a more directly applicable baseline than a general AI governance framework, and organizations assessing any agent-building SaaS vendor should apply SSCF's IAM domain controls as part of that evaluation [4]. CSA's *Enterprise AI Security Starts with AI Agents* survey report reinforces why this incident is representative rather than exceptional: the report found that 53% of organizations report their AI agents exceed intended permissions at least occasionally, that 47% have experienced an AI agent-related security incident, and that 58% indicate detection and response to such incidents takes five hours or longer, meaning most enterprises deploying agent-building platforms like Writer currently lack the visibility and response speed that would have let them detect and contain a WriteOut-style

token replay even after the fact [5]. Finally, CSA's AI Controls Matrix (AICM) v1.1 provides the broader governance baseline for agentic AI security, and its identity-and-access-management domain formalizes the session-scoping and least-privilege principles this research note recommends as short-term mitigations for any organization operating AI agent-building platforms at scale [6].

References

[1] The Hacker News. "Writer AI Flaw Could Let Agent Previews Leak Session Tokens Across Tenants." The Hacker News, July 7, 2026.

[2] BusinessWire. "Writer Raises $200M Series C at $1.9B Valuation to Fuel Leadership in Agentic Enterprise AI." Business Wire, November 12, 2024.

[3] SecureLayer7. "CVE-2026-55431: Coder CLI Session Token Exfiltration via External App URLs." SecureLayer7 Labs, 2026.

[4] Cloud Security Alliance. "SaaS Security Capability Framework (SSCF)." CSA SaaS Working Group, September 2025.

[5] Cloud Security Alliance. "Enterprise AI Security Starts with AI Agents." Cloud Security Alliance, commissioned by Zenity, April 2026.

[6] Cloud Security Alliance. "AI Controls Matrix (AICM) v1.1." Cloud Security Alliance, June 2026.