CSAI Foundation

Cloud Security Alliance AI Safety Initiative

LMDeploy SSRF: AI Inference Infrastructure Weaponized in 13 Hours

CVE-2026-33626 and the Accelerating Exploitation Cycle for AI Serving Tools

Unofficial AI-assisted Research

2026-04-25

Key Takeaways

- CVE-2026-33626 is a CVSS 7.5 (High) Server-Side Request Forgery vulnerability in LMDeploy's vision-language image loader, publicly disclosed on April 21, 2026, and first exploited in the wild just 12 hours and 31 minutes (approximately 13 hours) later [1][3].
- The root cause is an unvalidated `load_image()` function that fetches arbitrary attacker-supplied URLs without checking for private IP ranges, cloud metadata endpoints, or link-local addresses—making the model server a coercible HTTP proxy [3][4].
- No public proof-of-concept code had been published in monitored exploit repositories at time of exploitation; the advisory text provided sufficient technical detail to construct a working exploit, a pattern consistent with an emerging hypothesis about LLM-assisted advisory weaponization, though direct evidence remains circumstantial [1].
- The observed attacker conducted a systematic 8-minute, 10-request reconnaissance campaign targeting AWS IAM credential endpoints, internal Redis and MySQL services, out-of-band DNS channels, and distributed inference control endpoints [1][2].
- Organizations running LMDeploy with vision-language support should immediately upgrade to v0.12.3 or later, enforce IMDSv2 token requirements on all inference nodes, and restrict egress from inference workloads to prevent metadata service access [1][3].

Background

LMDeploy is an open-source inference and serving toolkit developed by the InternLM project at Shanghai AI Laboratory, designed for efficient deployment of large vision-language and text models [7]. The toolkit supports InternVL2, internlm-xcomposer2, Qwen2-VL, and a range of other frontier models, exposing an OpenAI-compatible REST API that allows organizations to self-host powerful multimodal models in cloud or on-premises environments. With approximately 7,800 GitHub stars [1], LMDeploy occupies a growing niche in the enterprise AI infrastructure ecosystem, particularly for teams deploying vision-capable inference pipelines or Chinese-origin frontier models at production scale.

The toolkit's architecture supports multi-GPU parallelism, quantized inference, and distributed serving through an internal system called DistServe, which coordinates model shards across inference nodes via inter-process communication. Its vision-language components follow the convention established by OpenAI's multimodal API: image inputs arrive as URL references within the standard `/v1/chat/completions` request body, and the inference server fetches those URLs to load image data before passing it to the model. This design avoids requiring clients to encode and transmit large image payloads, but it creates an architectural attack surface: the trust boundary of the model server now extends outward to any URL that the underlying operating system can resolve, and by default, LMDeploy binds its API service to `0.0.0.0`, making any internet-accessible deployment directly reachable by external actors without authentication requirements specific to the image loading path.

The category of attack that this design enables is well-understood in web application security: Server-Side Request Forgery allows an attacker to coerce a server into making HTTP requests on their behalf, reaching endpoints not directly accessible from the internet. In the context of cloud-hosted AI inference, SSRF is particularly dangerous because cloud compute instances typically have privileged access to Instance Metadata Services that serve temporary IAM credentials, and because inference deployments frequently co-locate with internal caching layers, databases, and administrative interfaces that assume network adjacency implies trust.

Security Analysis

The Vulnerability

Igor Stepansky of Orca Security discovered CVE-2026-33626 and disclosed it through GitHub's coordinated advisory process, with the public advisory (GHSA-6w67-hwm5-92mq) published on April 21, 2026 [3]. The flaw resides in the `load_image()` function within `lmdeploy/vl/utils.py`. The vulnerable implementation verifies only that a provided URL begins with `http` before executing a network fetch—it performs no hostname resolution, maintains no IP range blocklist, and provides no filtering of link-local addresses (169.254.0.0/16), loopback (127.0.0.1), or RFC 1918 private ranges [3][4]. This means the function will fetch any syntactically valid HTTP URL without any consideration of where that URL resolves.

The advisory was assigned a CVSS 3.1 score of 7.5 (High) [1][4], reflecting the combination of network-accessible attack vector, low attack complexity, no required authentication or privileges, no required user interaction, and high confidentiality impact. All LMDeploy versions prior to v0.12.3 with vision-language

support are vulnerable [2][6]. The patch introduced in v0.12.3 adds an `_is_safe_url()` validation function that blocks fetches to link-local ranges, loopback interfaces, and RFC 1918 private addresses before any network connection is attempted [1][6].

The attack surface is broader than a cursory reading suggests. Because the SSRF is triggered by the `image_url` field in a standard chat completion request—the same field used by legitimate applications to pass image inputs to vision-language models—distinguishing malicious SSRF probes from normal usage at the application layer is difficult without deep inspection of the resolved destination. This pushes effective defense toward infrastructure-layer controls rather than input validation alone.

The Exploitation Timeline

Sysdig's Threat Research Team, operating a honeypot fleet that monitors exploitation of emerging AI infrastructure vulnerabilities, detected the first attack attempt against LMDeploy at 03:35 UTC on April 22, 2026—12 hours and 31 minutes after the advisory appeared on GitHub's public security advisory page at 15:04 UTC the previous day [1][2]. No public proof-of-concept code had been published in any monitored exploit repository at the time the attack was observed — a finding that, given the 12-hour window, suggests the attacker derived a working payload directly from the advisory text, possibly with LLM assistance [1][2]. The advisory's explicit identification of the vulnerable file path (`lmdeploy/vl/utils.py`), the vulnerable function name (`load_image()`), the vulnerable parameter (`image_url`), and the complete absence of validation created what is effectively a functional exploit description, requiring only the construction of a malicious API request to operationalize.

This pattern is consistent with an emerging hypothesis in AI infrastructure security: detailed coordinated disclosure advisories, authored in good faith to enable rapid patching, are increasingly being processed by commercial or customized LLMs to generate working exploits within hours of publication [1]. If this pattern holds across multiple incidents, it would suggest that the speed of weaponization is becoming less a function of attacker expertise and more a function of advisory specificity — a hypothesis worth tracking as AI infrastructure CVEs accumulate.

The Attack Chain

The attacker, originating from IP address 103.116.72.119 (geolocated to Hong Kong and assessed as potentially proxied) [1][2], conducted a methodical three-phase reconnaissance campaign over a single eight-minute window using ten distinct HTTP requests to the honeypot's inference endpoint.

The first phase, spanning approximately two minutes beginning at 03:35 UTC, focused on credential and service reconnaissance. The session opened with a probe of the AWS Instance Metadata Service at `http://169.254.169.254/latest/meta-data/iam/security-credentials/`, the standard endpoint for retrieving temporary IAM credentials associated with an EC2 instance's attached role. A concurrent probe targeted Redis on localhost port 6379, a common high-throughput caching backend in inference deployments that may contain session state, request history, or model metadata. This immediate focus on credential endpoints and internal services indicates the attacker arrived with a specific operational objective—credential exfiltration or lateral movement—rather than exploratory curiosity about the SSRF primitive itself [2].

The second phase, observed around 03:41 UTC, shifted to validation and surface enumeration. The attacker dispatched an out-of-band DNS callback to a requestrepo.com endpoint, confirming both that the server resolved external hostnames and that an exfiltration channel was functional. Requests to `/openapi.json` and the API root path provided a map of available inference endpoints, including details about which model names were loaded and what API routes were exposed—intelligence useful for targeting subsequent requests and for characterizing the deployment's capabilities [2].

The third phase, from approximately 03:42 to 03:43 UTC, targeted infrastructure-specific functionality. A POST request to `/distserve/p2p_drop_connect`—an unauthenticated endpoint in LMDeploy's distributed serving subsystem—probed for the presence of multi-node inference configurations and may have been intended to disrupt inter-node coordination. Additional probes scanned localhost ports 8080 (a common administrative interface), 3306 (MySQL), and 80 (general HTTP). Notably, the attacker alternated between two vision-language model endpoints—internlm-xcomposer2 and InternVL2-8B—during the session, suggesting an effort to identify which model configurations were active and potentially to vary request signatures to reduce the probability of matching simple pattern-based detection rules [1][5].

The entire attack unfolded with a degree of procedural discipline that suggests manual direction or semi-automated execution with human oversight, rather than indiscriminate automated scanning [5]. The attacker did not attempt brute-force enumeration, did not generate high request volumes, and progressed through clearly distinct operational phases. This behavioral profile is consistent with a targeted reconnaissance mission rather than opportunistic scanning.

The Broader Pattern

CVE-2026-33626 is not an isolated event. It represents a point on a clearly visible trajectory: the tools used to host, serve, and scale large language models are high-value targets whose vulnerability-to-exploitation timelines are compressing. Sysdig's researchers explicitly noted that "critical vulnerabilities in

inference servers, model gateways, and agent orchestration tools are being weaponized within hours of advisory publication, regardless of the size or extent of their install base" [1]. LMDeploy's relatively modest community compared to larger ecosystems around vLLM or Ollama did not provide any meaningful protection from targeted exploitation. Attackers increasingly treat AI inference infrastructure—any of it—as a priority attack surface due to the concentration of cloud credentials, sensitive inference workloads, and often-inadequate network isolation it represents.

This vulnerability also illustrates a class problem for AI inference deployments: the multimodal input paradigm, in which model servers fetch external resources at the direction of request parameters, creates SSRF risk by design unless explicit URL validation is built in from the start. As more inference frameworks add vision-language support, image generation capabilities, and tool-calling features that involve outbound HTTP, the attack surface in this category will grow. CVE-2026-33626 should be treated as a prompt to audit all AI serving tools in use for analogous patterns—not merely as a single vulnerability to patch and move on from.

Recommendations

Immediate Actions

Organizations running any version of LMDeploy with vision-language support should treat this as a patch-now vulnerability. Upgrading to v0.12.3 or later is the definitive remediation; this release introduces the `_is_safe_url()` validation function that blocks fetches targeting link-local addresses, loopback interfaces, and RFC 1918 private ranges before any network connection is established [2][3]. For deployments that cannot be patched immediately, an API gateway or reverse proxy positioned in front of the inference endpoint should be configured to inspect and reject or rewrite `image_url` parameters containing internal IP ranges or link-local prefixes. This is a temporary measure, not a substitute for patching.

All AWS inference nodes—whether or not they run LMDeploy—should immediately enforce IMDSv2 with `httpTokens=required`. IMDSv2's session-oriented token exchange cannot be satisfied via SSRF from an application layer, because the token must be obtained through a PUT request with a TTL header that typical SSRF primitives cannot replicate. This control would have blocked credential exfiltration through the IMDS endpoint even without a software patch [1]. Equivalent controls for GCP (disabling legacy metadata APIs) and Azure (restricting IMDS access via instance-level firewall rules) should be verified and enforced across all inference workloads.

Security teams should audit IAM roles attached to inference instances for excessive permissions and rotate any credentials or session tokens that may have been exposed on systems running vulnerable LMDeploy versions between April 21 and the date of patching or mitigation.

Short-Term Mitigations

Network segmentation is an essential and often underimplemented layer of defense for AI inference workloads. Inference servers should not have unrestricted outbound internet access; egress should route through an allowlist-controlled proxy or be restricted to the specific external endpoints required for legitimate operation. VPC security groups, network ACLs, and host-based firewall rules should explicitly block outbound connections from inference processes to 169.254.0.0/16, 127.0.0.0/8, and RFC 1918 ranges except where specific internal service communication is required and documented. The unauthenticated `/distserve` distributed-serving endpoints exposed by LMDeploy should be placed behind network controls and protected with mutual TLS or API token authentication, given that the observed attacker specifically targeted this interface.

Runtime detection capabilities should be extended to cover SSRF exploitation patterns in AI inference environments. Sysdig has published Falco rules that detect outbound connections from container workloads to cloud instance metadata endpoints on AWS, GCP, and Azure [2]. Organizations using endpoint detection, network monitoring, or CSPM tooling should add detection logic for connections originating from model-serving processes to metadata service ranges, and establish alerting for anomalous DNS resolution patterns or high-frequency outbound connection attempts from inference containers. The behavioral profile of the observed attack—methodical, low-volume, multi-phase—means that volume-based detection thresholds would not have caught it; behavioral and destination-based detection is required.

Strategic Considerations

The 12-hour exploitation window for CVE-2026-33626 is a data point that should inform patching urgency for AI inference tools. Whether it represents a floor, a ceiling, or the median for this vulnerability class requires broader comparison, but it is sufficient to conclude that traditional monthly or quarterly patching cycles are structurally incompatible with this threat timeline. AI inference frameworks are updated frequently—often driven by performance benchmarks and model compatibility rather than security—and they occupy a part of the stack that security teams often treat as a developer-managed dependency rather than a patched infrastructure component. That categorization needs to change.

Organizations should establish an AI infrastructure vulnerability management program that treats inference servers, model gateways, and agent orchestration tools as a distinct high-priority asset class, subject to the same emergency patching urgency applied to internet-facing web application components. This program should maintain a monitored list of open-source AI serving frameworks in production use—including LMDeploy, vLLM, Ollama, Text Generation Inference, Ray Serve, and any API compatibility layers or model routers—and subscribe to their security advisory channels. Software composition analysis tools should include these frameworks and their transitive dependencies, with vulnerability feeds tied to remediation SLAs measured in hours for CVSS High and Critical findings, not weeks.

The SSRF class of vulnerability merits particular attention in this context because it is poorly suited to application-layer defenses in AI serving architectures. Users legitimately provide URLs as model inputs in multimodal applications, making malicious URL submission structurally indistinguishable from normal usage at the API layer. Effective defense must therefore be implemented at the infrastructure layer: outbound network controls, metadata service hardening, and runtime process monitoring provide more reliable coverage than input filtering and are not bypassable by attacker-controlled content. Organizations evaluating AI inference frameworks should include SSRF resistance—specifically, whether the framework validates URLs against private IP ranges before fetching—as a security evaluation criterion alongside authentication, authorization, and TLS configuration.

CSA Resource Alignment

MAESTRO. CSA's Agentic AI Threat Modeling Framework [8] provides a layered model for understanding where CVE-2026-33626 operates in an AI deployment's attack surface. The vulnerability is a deployment infrastructure threat: it does not target the model's reasoning, training data, or inference outputs, but rather exploits the serving layer that hosts inference. MAESTRO's guidance on model serving infrastructure emphasizes network isolation, credential segregation, and runtime monitoring for unexpected outbound network behavior—all of which correspond directly to the mitigation layers most relevant to this vulnerability class. Security architects using MAESTRO to model LLM deployment environments should include vision-language image loading as a distinct attack vector in their threat model and map it to the external data fetch category of risks.

AICM. The CSA AI Controls Matrix [9] addresses deployment infrastructure security across several control domains, including API authentication and authorization controls for AI serving endpoints, network access controls for model workloads, and secrets management for credentials accessible to AI systems. Organizations mapping LMDeploy deployments to AICM should examine controls in the AI Deployment and AI Infrastructure domains covering outbound network restriction, runtime process

integrity monitoring, and least-privilege IAM configuration. CVE-2026-33626 represents a failure across multiple AICM control objectives—the absence of URL validation (application control), unrestricted egress from the inference process (network control), and overprivileged IAM roles on inference nodes (identity and access control) all contributed to the risk realized.

STAR for AI. CSA's STAR for AI program [10] provides a structured assurance framework for organizations to assess and communicate the security posture of AI-serving infrastructure. The LMDeploy incident illustrates why STAR for AI's emphasis on deployment-time risk assessment—examining the security of the infrastructure hosting and serving models, not solely the properties of the models themselves—is essential. Self-hosted inference deployments introduce a class of infrastructure risk that model cards and algorithmic evaluations cannot capture; STAR for AI provides the right framework for assessing whether serving infrastructure meets the security baseline required for production AI workloads.

Zero Trust. CSA's Zero Trust guidance applies directly to the network architecture recommendations in this note. Zero Trust principles—assume breach, verify every connection, enforce least-privilege at every layer—are precisely the controls that would have contained CVE-2026-33626 exploitation even without a software patch. An inference node that cannot reach 169.254.0.0/16 cannot exfiltrate IMDS credentials regardless of application-layer vulnerabilities. A model server whose egress is controlled by an allowlist proxy cannot pivot to internal Redis or MySQL services. Zero Trust network architecture for AI inference is not an aspirational posture; it is an immediately applicable set of controls whose logical effect — blocking outbound access to metadata ranges — directly addresses the attack techniques observed in this incident.

References

[1] Sysdig Threat Research Team. "CVE-2026-33626: How Attackers Exploited LMDeploy LLM Inference Engines in 12 Hours." Sysdig, April 2026.

[2] The Hacker News. "LMDeploy CVE-2026-33626 Flaw Exploited Within 13 Hours of Disclosure." The Hacker News, April 24, 2026.

[3] InternLM. "Server-Side Request Forgery (SSRF) in Vision-Language Image Loading — GHSA-6w67-hwm5-92mq." GitHub Security Advisory, April 21, 2026.

[4] GitLab Security Advisories. "CVE-2026-33626: LMDeploy has Server-Side Request Forgery (SSRF) via Vision-Language Image Loading." GitLab, April 2026.

[5] Vulert. "LMDeploy CVE-2026-33626 Exploited Within 13 Hours — SSRF Flaw Exposes AI Infrastructure." Vulert, April 2026.

[6] Cyberpress. "New LMDeploy Vulnerability Exploited in the Wild Just 12 Hours After Public Advisory." Cyberpress, April 2026.

[7] InternLM. "LMDeploy: A Toolkit for Compressing, Deploying, and Serving LLMs." GitHub, 2026.

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

[9] Cloud Security Alliance. "AI Controls Matrix (AICM)." CSA, 2025.

[10] Cloud Security Alliance. "CSA STAR for AI." CSA, 2026.

