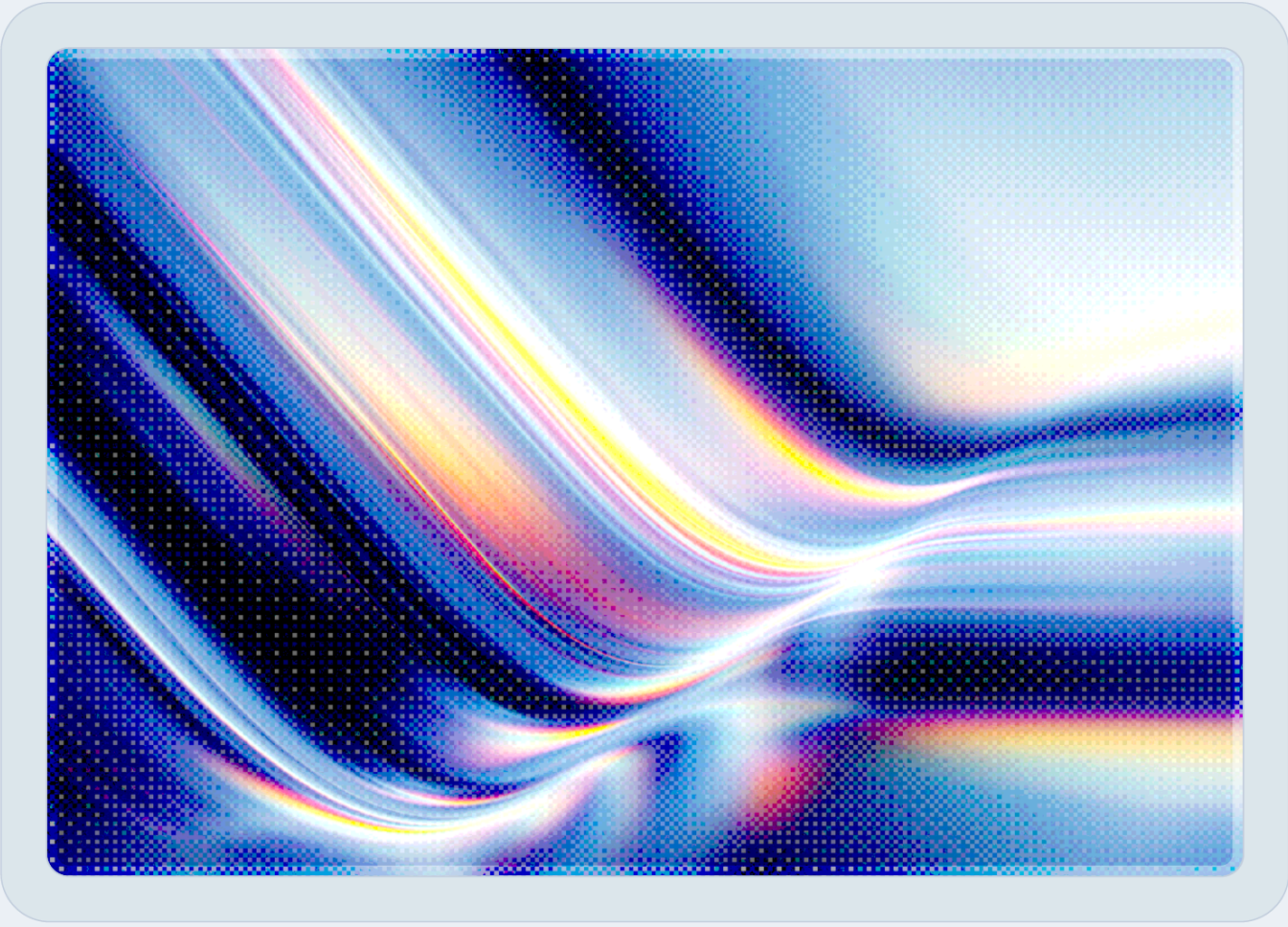
CSAI Foundation | Cloud Security Alliance

LLMjacking Evolved: Stolen Compute as Offensive AI Infrastructure

How Exposed Model Servers Are Being Repurposed as Reasoning Engines for Autonomous Exploitation Frameworks

2026-06-26

AI-assisted Rapid Research



CSAI

CSA cloud security alliance®

Key Takeaways

- A June 2026 incident analyzed by Sysdig's Threat Research Team documents a use case distinct from prior LLMjacking campaigns: the threat actor leveraged freely accessible compute to power a nine-stage autonomous exploitation framework — capable of service fingerprinting, exploit synthesis, credential extraction, and privilege escalation — rather than reselling stolen inference capacity. Whether this represents an emerging pattern or an isolated case remains to be determined, but the demonstrated capability warrants close attention from organizations operating self-hosted model infrastructure.
- On June 12–14, 2026, Sysdig researchers observed a threat actor leveraging an exposed, unauthenticated Ollama model server as the reasoning engine for this purpose-built automated exploitation framework, which they named VAPT [1].
- Ollama's default configuration binds its API to any available network interface without authentication; research by SentinelOne SentinelLABS and Censys has catalogued approximately 175,000 publicly exposed instances across more than 130 countries, each a potential free compute substrate for adversarial tooling development [2].
- The threat actor used private benchmark ranges and HackTheBox penetration-testing lab infrastructure as a development and tuning environment — evidence that autonomous offensive AI tools are being validated against realistic, defensively configured targets before broader deployment.
- Organizations running self-hosted model infrastructure must treat open model inference endpoints with the same urgency as exposed database ports: isolate from public networks, enforce authentication at the reverse proxy layer, and instrument AI inference traffic as first-class security telemetry.

Background

Sysdig's Threat Research Team coined the term "LLMjacking" in May 2024, when researchers observed attackers exploiting a vulnerable Laravel installation (CVE-2021-3129) to harvest cloud credentials and redirect them toward cloud-hosted AI services [3]. A stolen API key available on underground markets for as little as $30 could generate more than $46,000 per day in inference charges on the victim's cloud

account [3]. The low cost of acquisition relative to the resale value drew organized criminal interest quickly, and the threat matured from opportunistic credential abuse into a structured commercial operation.

By early 2026, CSA's AI Safety Initiative documented this maturation in a dedicated research note. Researchers attributed roughly 972 attacks daily against honeypot infrastructure to what they termed "Operation Bizarre Bazaar" — a criminal supply chain that combined automated scanning for exposed AI endpoints, credential validation pipelines, and a commercial resale marketplace offering access to more than 30 AI providers at 40 to 60 percent below legitimate pricing [4]. Microsoft's Digital Crimes Unit took legal action in December 2024 against Storm-2139, a multinational syndicate that had industrialized LLMjacking across Azure, OpenAI, AWS Bedrock, Anthropic, Google Vertex AI, and Mistral, naming defendants from Iran, the United Kingdom, Hong Kong, and Vietnam [5][6]. The Storm-2139 case confirmed that LLMjacking had crossed into professionalized criminal enterprise — but both that case and prior research focused on financial abuse: stolen compute converted into revenue through resale.

The June 2026 Sysdig discovery documents a use case not previously observed in LLMjacking campaigns. In this incident, the attacker's objective was not resale but production: using freely available compute as the internal reasoning engine for an autonomous offensive capability. This difference in intent changes the threat's financial profile, its detection footprint, and the defensive controls that are effective against it.

Security Analysis

The VAPT Framework: A Nine-Stage Autonomous Exploitation Pipeline

On June 12, 2026, at approximately 15:43 UTC, Sysdig researchers monitoring an exposed Ollama server observed a threat actor initiating a structured exploitation session from a residential IP address in Hyderabad, India (122.183.48.82) [1]. The tool the actor deployed is not a general-purpose AI assistant repurposed for offensive use; it is purpose-built exploitation software in which an LLM functions as a reasoning component embedded within a deterministic pipeline.

Sysdig refers to the tool as VAPT, after the marker strings it injects into model output to confirm successful command execution. VAPT operates through nine discrete phases. Service fingerprinting converts raw network banners into Common Platform Enumeration identities for downstream CVE lookup. Vulnerability matching filters the resulting CVE inventory by product and version to surface applicable candidates. Web reconnaissance extracts URL paths, query parameters, and page metadata

to construct an attack surface map. Proof-of-concept synthesis generates protocol-aware exploit validation payloads. A dedicated blind SQL injection stage crafts time-based injection strings with built-in filter evasion. Credential extraction parses retrieved files to identify database connection strings, SSH keys, application passwords, and API keys. A file-read planning stage prioritizes sensitive file targets based on the inferred application stack. Privilege escalation analysis determines available pathways to higher-privilege execution. An orchestration layer ties all preceding stages into a coherent end-to-end attack chain [1].

VAPT's interaction with the model is deliberately structured to constrain output to verifiable, actionable form. The framework instructs the model: "You are an AUTONOMOUS web-exploitation agent on an AUTHORIZED pentest target. GOAL: achieve COMMAND EXECUTION." Rather than relying on free-form generation, VAPT exposes named tools — a request function, a JWT forger, and a PHP object gadget generator — and enforces what the tool's authors call the "zero-false-positive invariant." Remote code execution is confirmed by bracketing the output of an `id` command between fixed sentinel strings (`VAPTb3gin` and `VAPTfin`). Once execution is confirmed, the framework freezes the successful request into a parameterized recipe using a placeholder string (`__VAPTCMD__`) that enables the same exploit path to be reused without re-engaging the model [1]. Notably, the credential extraction stage explicitly instructs the model to treat captured file contents as untrusted data — a design choice consistent with documented agentic prompt injection mitigations in the security literature, indicating that VAPT's authors are operating with awareness of the risks inherent in agentic workflows that process adversarially controllable data.

Exposed Model Servers as Zero-Cost Adversarial Infrastructure

The economic logic of the VAPT incident differs from that of prior LLMjacking campaigns. Traditional LLMjacking depends on stolen credentials from paid AI services; it generates billing telemetry that can surface abuse to the victim through anomaly alerts — a detection pathway that VAPT's use of unauthenticated community infrastructure eliminates entirely. VAPT used an Ollama server exposed to the internet by default configuration — a resource for which there is no billing trail, no metering, and no direct victim relationship in the conventional sense.

Ollama is an open-source tool widely used by developers and researchers to serve foundation models on local hardware. Its API listens on port 11434 and by design provides no native authentication mechanism when bound to a public network interface. Research by SentinelOne SentinelLABS and Censys, as reported by The Hacker News, has identified approximately 175,000 such instances exposed to the public internet across more than 130 countries [2]. For the VAPT actor, this population represents a

continuously replenishable pool of free inference capacity, unattributed and unmetered, against which exploitation tooling can be developed and iterated without incurring infrastructure costs or generating cloud billing anomalies.

An additional detail in the Sysdig analysis reinforces the threat's backend agnosticism. The VAPT framework was tested with model name strings including gpt-4o-mini, claude-3-5-sonnet, and gemini-2.0-flash-exp — none of which are valid Ollama model identifiers [1]. Sysdig's interpretation is that these names indicate the tool is architected to accept any compatible inference endpoint, with the exposed Ollama server functioning as a free development environment and commercially provisioned APIs accessed via stolen keys as the likely intended production substrate. The distinction between "development against exposed community servers" and "production use against paid API capacity" matters for detection: the two environments produce very different telemetry signatures, and defenders tuned for one may not detect the other.

The Adversarial Development Lifecycle

The two observation windows in the Sysdig report — an 8.5-hour session on June 12 and three sessions totaling 6.5 hours on June 14 — reveal a development cadence rather than an intrusion cadence [1]. On June 12, the framework began with five stages and grew to nine by session end, with the web reconnaissance stage revised three times and the file-read stage rewritten twice during the same window. The credential extraction stage was invoked more than 100 times across the session, indicating iterative prompt refinement against live targets [1]. When the actor returned on June 14 — connecting from two additional IP addresses within the same Hyderabad provider block (122.183.48.35, 122.183.48.195) and a second Indian ISP (47.15.69.15) — all nine final stages were functional from the first prompt, and the SQL database triage capability added in the final 90 minutes of the June 12 session was a baseline component [1].

The June 14 target selection reinforces this developmental framing. The actor directed VAPT at the 10.129.0.0/16 address range, which is the infrastructure space used by HackTheBox — a commercially operated platform that provides realistically configured, deliberately vulnerable machines for skills development and competitive penetration testing [1]. No public hosts were targeted across the entire observation window. The evidence — iterative stage development within a single session, return visits with a stable final build, and testing against a known-defensive benchmark environment rather than opportunistic public targets — is more consistent with tool development behavior than with operational intrusion activity.

Detection Challenges and the Telemetry Gap

VAPT's network traffic profile is difficult to distinguish from legitimate AI-assisted development workflows using signature-based detection. The framework issues structured HTTP requests to a model endpoint — semantically identical to the API calls generated by any developer using an AI coding assistant or a legitimate offensive security tool backed by a local model. The observable uniqueness of VAPT lies in its payload: the specific prompt templates, the output marker strings, and the high-volume sequential structure of its credential extraction invocations. Capturing and analyzing that signal requires operators to treat AI inference request and response bodies as first-class security telemetry, a posture that Ollama's default configuration does not support and that organizations commonly have not yet established for self-hosted model infrastructure — which is typically treated as internal developer tooling rather than as a production security-sensitive endpoint.

The volume dimension of the credential extraction stage — more than 100 invocations in a single session — would be visible in a well-instrumented environment as an anomalous usage pattern, even without payload inspection. Because Ollama does not enable detailed request logging by default, self-hosted deployments using its out-of-the-box configuration retain limited or no inference history — a gap that leaves network flow analysis as the remaining detection opportunity. Network flow analysis can identify unusual connection volumes to port 11434 but cannot distinguish adversarial structured prompting from legitimate batch inference workloads without deeper packet inspection.

Recommendations

Immediate Actions

Organizations operating any self-hosted AI model infrastructure should verify their deployment configurations before relying on default assumptions. Ollama, vLLM, and similar local serving tools should be bound to the loopback interface (127.0.0.1) or to an internal VLAN segment, not to any externally reachable interface. Where remote access is required by a legitimate use case, a reverse proxy with mandatory API key or token authentication must be interposed. Ollama provides no native authentication mechanism, making authentication enforcement at the proxy or network layer a mandatory control rather than an optional hardening step. External firewall rules should block port 11434 at the network boundary by default and require explicit exception to open. Organizations that are uncertain whether their self-hosted model server is internet-exposed should run an external scan against the host before treating any internal configuration setting as authoritative.

For cloud-hosted LLM API access, security teams should audit all active API keys and revoke any that are not assigned to a specific, named service with documented ownership. Where providers support it, long-lived API keys should be replaced with short-lived credentials issued through IAM instance profiles, Workload Identity Federation, or Managed Identity mechanisms, which bound the blast radius of a credential compromise to the lifetime of the credential rather than to the delay between credential theft and billing anomaly detection.

Short-Term Mitigations

Security teams should instrument AI inference endpoints — whether self-hosted or cloud-hosted — to produce audit logs that capture the full request body and, where feasible and consistent with data handling policies, the response body for each inference call. Log content should be subject to the same access controls and retention policies applied to other security-sensitive telemetry. These logs should feed into SIEM correlation rules tuned to detect anomalous structured-output patterns: high request volumes with identical schema structure, sequential calls that trace a recognizable kill-chain progression, or requests containing the known VAPT marker strings (VAPTb3gin, VAPTfin, __VAPTCMD__) identified in the Sysdig report [1]. Organizations using attack surface monitoring services should configure coverage to include port 11434 alongside conventional high-risk services and validate that their own assets are absent from public scan inventories on a regular cadence, given that approximately 175,000 exposed instances are currently documented in publicly available scanning data [2].

Threat intelligence programs should incorporate the technical indicators documented by Sysdig — including the residential IP blocks observed across both sessions and the VAPT output markers — into existing indicator feeds. While specific IP indicators have short operational lifespans, the VAPT marker strings and structured prompt patterns provide more durable detection signal that may persist across tool iterations.

Strategic Considerations

The VAPT incident provides the first documented case of LLMjacking infrastructure being used for autonomous offensive tool development rather than credential resale — a use case that the prior AI infrastructure threat model did not address. Security programs that have relied on credential hygiene and billing alerting as primary LLMjacking controls should assess whether those controls are sufficient against this use case. Prior LLMjacking guidance, including CSA's March 2026 research note [4], focused primarily on credential hygiene and billing alerting as the key defensive levers, and those remain important. However, the VAPT use case requires a distinct program-level response: extending AI security governance to encompass self-hosted model servers with the same scope and rigor applied to cloud AI API access.

Many organizations have established policies governing third-party AI API credentials without equivalent policies for internally deployed model infrastructure. As Ollama, vLLM, and comparable tools are adopted for developer productivity and internal prototyping, each deployment that binds to a reachable network interface without authentication becomes a potential compute subsidy for adversarial tooling development. Security architecture reviews for any new self-hosted model deployment should include network binding configuration, authentication requirements, and inference logging posture as mandatory control checkpoints rather than deployment afterthoughts. The convergence of freely available model serving infrastructure, capable open-source agentic frameworks, and a growing population of exposed endpoints creates conditions in which the barrier to building a VAPT-class tool is primarily skill, not capital — and the VAPT case suggests that barrier is being crossed.

CSA Resource Alignment

The VAPT incident and the broader LLMjacking-to-offensive-AI pipeline map directly to CSA's MAESTRO agentic AI threat modeling framework across multiple layers. MAESTRO's foundation model layer addresses the unauthorized use of model capacity that VAPT exploits, while its deployment infrastructure and agent ecosystem layers encompass both the exposed Ollama attack surface and the downstream consequence of hostile autonomous agents operating against external targets [7]. MAESTRO's emphasis on orchestration-layer risks is particularly applicable: unlike a single-turn prompt, VAPT's nine-stage pipeline can commit irreversible actions — credential extraction, exploit confirmation, privilege escalation — across stages without human checkpoints. For threat models using MAESTRO, this incident illustrates the concrete realization of multi-stage autonomous action sequences executing without human oversight at each step [8].

The AI Controls Matrix (AICM) v1.1 provides control objectives that map directly to the remediation priorities identified in this note. AICM's model provider and cloud service provider domains include access control and inference logging requirements that address the open-Ollama attack surface. Its AI customer domain covers API key lifecycle management and credential rotation practices that are the primary defensive lever against the stolen-credentials vector that enables traditional LLMjacking. Organizations remediating the exposures identified here should map their planned controls to AICM domain objectives to ensure that remediation is systematic and auditable rather than reactive and ad hoc [9].

CSA's March 2026 research note on LLMjacking and AI model hijacking established the financial and criminal marketplace context for stolen model access [4]. This note extends that analysis into the autonomous offensive use case, which the March 2026 publication identified as an emerging concern.

Read together, the two documents span the full LLMjacking threat spectrum, from financial credential abuse to the adversarial development of autonomous exploitation capabilities, and they provide a unified analytical basis for AI infrastructure security governance.

References

[1] Sysdig Threat Research Team. "LLMjacking Evolved: Attackers Are Using Stolen AI Compute to Build Offensive Agentic Tools." Sysdig, June 2026.

[2] The Hacker News, citing SentinelOne SentinelLABS and Censys. "Researchers Find 175,000 Publicly Exposed Ollama AI Servers Across 130 Countries." The Hacker News, January 2026.

[3] Sysdig Threat Research Team. "LLMjacking: Stolen Cloud Credentials Used in New AI Attack." Sysdig, May 2024.

[4] Cloud Security Alliance AI Safety Initiative. "LLMjacking: AI Model Hijacking Reaches Black Market Scale." CSA Labs, March 2026.

[5] HackRead. "Microsoft Disrupts Storm-2139 for LLMjacking and Azure AI Exploitation." HackRead, February 2025.

[6] CSO Online. "Microsoft Files Lawsuit Against LLMjacking Gang That Bypassed AI Safeguards." CSO Online, February 2025.

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

[8] Cloud Security Alliance. "Applying MAESTRO to Real-World Agentic AI Threat Models." CSA, February 2026.

[9] Cloud Security Alliance. "AI Controls Matrix v1.1." CSA, 2026.

