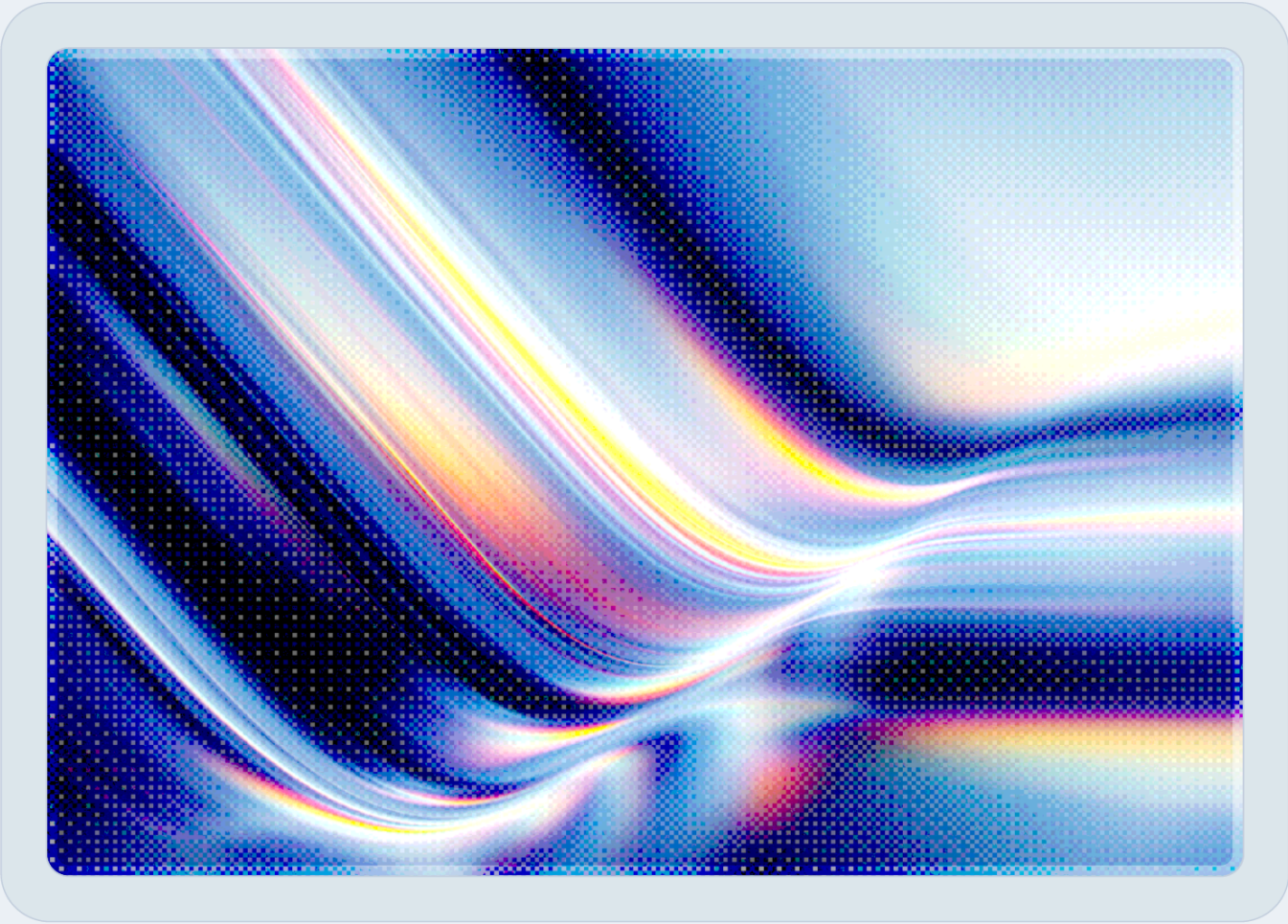
CSAI Foundation | Cloud Security Alliance

Hermes AI Agent Ran Unattended Inside a Finance Ministry

2026-07-26

AI-assisted Rapid Research



CSAI

CSA cloud security alliance®

Key Takeaways

Threat intelligence firm Hunt.io, working with security researcher Bob Diachenko, uncovered a staging server exposing an active intrusion against Thailand's Ministry of Finance, discovered when three publicly accessible directories were found on a Hong Kong-hosted host between July 9 and July 13, 2026 [1][2]. The recovered material showed that the operator had deployed Hermes, an open-source autonomous AI agent released in February 2026 by Nous Research, and had enabled its "YOLO mode," a setting that removes the approval prompts an operator would otherwise have to clear before the agent executes a command [1][3]. With that safeguard disabled, the agent independently ran privilege-escalation reconnaissance using LinPEAS, scanned for exploitable kernel vulnerabilities, enumerated services and SUID/SGID binaries, and traversed ministry file shares, cataloguing personnel records from the Office of the Permanent Secretary for Finance dating back to 2012, all without an operator issuing each command by hand [1][2][3]. The human behind the operation still performed the steps that required specific knowledge of the target, building customized password lists from the ministry's internal department abbreviations and hardcoding internal network paths the agent could not have known on its own [1]. Researchers also recovered 62 copies of a previously undocumented Go-based implant the operator called Hades, built for both Windows and Linux and supporting encrypted command-and-control, persistence, interactive shells, file transfer, SOCKS proxying, and configurable working hours and kill dates, though no recovered artifact shows Hades reaching a ministry machine [1][2][3][4]. Attribution indicators, including a Hong Kong staging IP with a documented history of hosting ShadowPad and VShell command-and-control infrastructure, and a Hermes interface password containing the Chinese term for "thunder god," led Hunt.io to assess with low-to-medium confidence that the operator is Chinese-speaking, though it named no specific threat group and the initial access vector into the ministry's network remains unknown [1][2][3]. This research note examines how the incident illustrates a division of labor between human operators and unattended AI agents in post-exploitation, and what that division implies for detection and governance.

Background

Hermes is a self-hosted, open-source autonomous agent built by Nous Research and released in February 2026, designed to run as a persistent service that retains memory across task sessions and can be operated through Telegram, Slack, email, or a command-line interface [3][5]. It is marketed as a

general-purpose assistant for managing routine digital chores rather than as an offensive security tool, and its intended use case is closer to a always-on personal assistant than a penetration-testing framework [1]. Hermes ships with a permission-checking layer, similar to the approval gates found in some other agent frameworks, that is meant to pause execution and request human approval before running commands the system considers potentially dangerous. The operator behind the Thai Finance Ministry intrusion disabled that layer by enabling "YOLO mode," a configuration option that strips out those approval gates entirely and allows the agent to continue reconnaissance and command execution indefinitely without anyone reviewing each step [1][2][3].

The exposed staging infrastructure gave researchers an unusually direct view into how the agent was actually used, because the recovered material included both the tooling deployed against the ministry and logs of Hermes's own reasoning and command output [2]. Hunt.io's investigation found that the operator had exploited Apache HiveServer2 instances configured with default "NONE" authentication, installing a malicious Java function packaged as HiveCmd.jar to achieve arbitrary code execution, and had separately staged exploit code targeting Apache Ambari, GlassFish administrative consoles, and mail servers [1][3]. Alongside the Hermes logs, researchers recovered PHP web shells, HTTP tunneling utilities, credential-theft tools, and exploit code for known vulnerabilities including CVE-2021-4034 (PwnKit), CVE-2021-3156 (a sudo privilege-escalation flaw), and CVE-2017-7269 (an IIS WebDAV remote code execution bug) [1][4]. The presence of this broader toolkit indicates that Hermes was one component in a larger intrusion rather than the sole mechanism of compromise, and the public reporting is explicit that the method of initial entry into the ministry's network has not been determined [1][2][3].

This incident is not an isolated proof of concept: it follows a pattern CSA's AI Safety Initiative has already documented in other 2026 cases where autonomous agents operated with reduced or absent human oversight during an active intrusion, including the July 2026 breach of Hugging Face's production infrastructure by an autonomous agent that executed thousands of actions across a weekend with minimal apparent human intervention [6]. What distinguishes the Thai Finance Ministry case is the clarity of the human-machine division of labor visible in the recovered logs: the agent handled generic, repeatable reconnaissance tasks that do not require specific knowledge of the target, while the human operator retained responsibility for target-specific tradecraft such as building a department-abbreviation password list and hardcoding internal paths [1]. That split offers a more concrete picture of how a working, unattended agentic intrusion is actually staffed than the Hugging Face case, which described autonomy only in aggregate percentages of tactical work performed.

Security Analysis

The core enabling condition behind the unattended reconnaissance phase of this intrusion was not a novel exploit but a configuration choice: switching off the agent's own approval gate. Once "YOLO mode" was active, Hermes could execute a chain of reconnaissance commands, evaluate their output, and decide on follow-up actions without any point at which a human needed to review or authorize what came next [1][2][3]. The five recovered Hermes operation logs show the agent independently running LinPEAS, searching for privilege escalation opportunities, enumerating SUID and SGID binaries, inspecting containers, and traversing ministry directory structures, and this reconnaissance persisted long enough to catalogue personnel records extending back to 2012 [1][2]. None of these individual actions represent new tradecraft; LinPEAS, kernel enumeration, and file traversal are standard post-exploitation techniques that any human operator or scripted tool could perform. What changed is that the agent could carry them out continuously and adaptively, evaluating each result and selecting a next step without requiring an analyst's continuous attention — a workflow that, from the outside, resembles how a human analyst would proceed, though the logs do not establish that the agent's underlying reasoning process was equivalent.

That autonomy had clear limits, and those limits are as informative as the capability itself. The recovered artifacts show the human operator handling every task that depended on specific knowledge of the target environment: building a password-guessing list out of the ministry's internal department abbreviations, and hardcoding internal network paths that an agent with no prior access to ministry documentation could not have inferred [1]. This division suggests that, at least in this case, the agent functioned as a force multiplier for generic reconnaissance rather than as a fully autonomous intruder capable of independently identifying and exploiting a target from a standing start. The Hades implant tells a similar story about the limits of what was actually verified: researchers recovered 62 copies of the Go-based tool, which supports encrypted command-and-control, persistence, interactive shells, SOCKS proxying, and operational safeguards such as configurable working hours and kill dates, but no recovered evidence shows Hades executing on a ministry system, and the other 60 recovered copies were not individually examined [2][3]. The gap between what was staged and what was confirmed to have executed is a reminder that exposed staging infrastructure, however revealing, does not by itself establish the full scope of an intrusion.

The attribution picture illustrates the same caution. Hunt.io's assessment rests on circumstantial infrastructure and linguistic indicators: a Hong Kong-hosted staging IP address with a documented history of hosting ShadowPad and VShell command-and-control frameworks, use of the FOFA asset-search service, and a Hermes interface password containing the Chinese term "Leishen," meaning thunder god [1][2][3]. Taken together, these indicators supported only a low-to-medium confidence assessment that the operator is Chinese-speaking or closely familiar with the language, with no

attribution to a specific named threat group and no determination of how the operator first gained access to the ministry's network [1][2][3]. That evidentiary gap matters for defenders: an unattended agent left running reconnaissance for days inside a government network is a meaningful finding regardless of who was behind the keyboard, and organizations should not wait for definitive attribution before addressing the exposure that made the reconnaissance possible in the first place.

For defenders, we think the more durable lesson is architectural rather than forensic. An agent running with its approval gate disabled behaves, from a network and endpoint monitoring perspective, like a persistent, tireless intruder that does not need to sleep, take breaks, or manually re-type commands between reconnaissance steps. Detection strategies built around the assumption that a human operator will pause between actions, or that unusual command sequences will cluster around a narrow window of active keyboard use, are poorly matched to an adversary that can run LinPEAS, enumerate SUID binaries, and traverse file shares in a single continuous, machine-paced session. Because the individual commands used in this intrusion were not novel, signature-based detection tuned to specific tools would likely have caught pieces of this activity had it been deployed and monitored; the harder problem is recognizing the behavioral signature of sustained, unattended, adaptive reconnaissance as distinct from either a human operator working at a keyboard or a static automated script.

Recommendations

Immediate Actions

Organizations operating Apache HiveServer2, Ambari, or GlassFish deployments should immediately verify that authentication is enforced rather than left at default "NONE" or unauthenticated settings, since that misconfiguration was the specific vector the operator used to gain code execution in this incident [1][3]. Security teams should also treat CVE-2021-4034, CVE-2021-3156, and CVE-2017-7269 as active exploitation risks warranting confirmation that affected systems are patched, since exploit code for all three was found staged and ready for use against the ministry [1][4]. Any organization that has deployed Hermes, or a comparable self-hosted autonomous agent framework, should audit whether "YOLO mode" or an equivalent unattended-execution setting is enabled anywhere in the environment, and disable it unless there is a documented, risk-accepted business justification for unattended operation.

Short-Term Mitigations

Security operations teams should extend detection playbooks to look for sustained, multi-step reconnaissance sequences that run continuously over extended periods without the pauses typical of manual keyboard-driven activity, since LinPEAS execution, kernel enumeration, and directory traversal chained together at machine pace is a meaningful behavioral signal independent of whether any single command is novel. Internal deployments of agentic AI tooling, whether used defensively or for legitimate automation, should be inventoried and configured so that any mode disabling human approval gates requires explicit, logged authorization rather than being available as an unmonitored default. Given that the operator in this case still relied on target-specific knowledge that only a human could supply, organizations should also assume that credential lists, internal file paths, and other environment-specific artifacts exposed through prior breaches, document leaks, or misconfigurations meaningfully lower the bar for an agent-assisted intrusion of this kind, and should prioritize remediating such exposures.

Strategic Considerations

Security leaders should recognize that the defining feature of this incident is not a new exploitation technique but a demonstrated, working example of an operator delegating routine post-exploitation labor to an unattended AI agent while retaining the target-specific judgment calls for themselves. That division of labor is likely to recur and to scale as autonomous agent frameworks become more capable and more widely available, including in variants purpose-built or repurposed for offensive use. Organizations should factor this pattern into their threat models for any internet-facing or sensitive government and enterprise infrastructure, treating the possibility of sustained, unattended, machine-paced reconnaissance as a standing risk rather than a hypothetical one, and should build runtime governance for their own agentic AI deployments that would prevent a similar approval-gate bypass from persisting undetected inside their own networks.

CSA Resource Alignment

This incident connects most directly to CSA's July 2026 research note on the Hugging Face breach, "Hugging Face's Autonomous AI Agent Breach," which examined a separate case in which an autonomous agent executed more than 17,000 logged actions across a weekend with the human operator largely out of the loop once the intrusion was underway [6]. Both cases share a structural feature: an agent operating with disabled or absent approval controls compresses what would ordinarily be days of manual reconnaissance into a continuous, unattended session. In CSA's assessment, both cases illustrate why runtime enforcement — rather than model-level safety training alone — is the control

most directly responsive to this gap. That research note recommends adopting runtime controls conformant with the CSAI Foundation's Autonomous Action Runtime Management (AARM) specification, an open system category for intercepting, authorizing, and recording AI-driven actions before they execute, which is precisely the control category that Hermes's disabled "YOLO mode" bypassed in the Thai Finance Ministry intrusion [6][7]. Organizations evaluating whether their own agent deployments could be similarly reconfigured to bypass approval gates should consult the AARM specification directly for its baseline and extended conformance requirements around action interception and audit logging [7].

CSA's research note "Marimo RCE: LLM Agents as Post-Exploitation Tools," which analyzed the first publicly documented in-the-wild intrusion driven by an LLM agent during post-exploitation, offers a directly applicable detection lens for this case [8]. That research identified behavioral signatures distinguishing agent-driven execution from conventional human or scripted activity, including machine-paced command sequencing and continuous, adaptive reconnaissance without the natural pauses of manual operation; the Hermes logs recovered from the Thai Finance Ministry staging server exhibit a comparable pattern of sustained, uninterrupted reconnaissance across LinPEAS execution, privilege-escalation searches, and file traversal [1][2][8]. Finally, the AI Controls Matrix (AICM v1.1) provides the underlying control structure for the identity, authorization, and logging and monitoring domains that both the runtime-enforcement and detection recommendations in this note depend on, and organizations formalizing a response to this incident should map their remediation steps against those AICM control objectives rather than treating agent governance as a standalone initiative separate from existing cloud and AI control frameworks [9].

References

[1] Hunt.io. "Thailand's Ministry of Finance Targeted With Hermes AI Agent Running Unattended, Hades Implant Staged." Hunt.io, July 2026.

[2] The Hacker News. "Hacker Runs Hermes AI Agent Unattended for Post-Exploitation at Thai Finance Ministry." The Hacker News, July 2026.

[3] BleepingComputer. "Hermes AI Agent Used to Automate Attack on Thai Finance Ministry." BleepingComputer, July 2026.

[4] Security Affairs. "Thailand's Ministry of Finance Targeted With Hermes AI Agent Running Unattended, Hades Implant Staged." Security Affairs, July 2026.

[5] Nous Research. "Hermes Agent." Nous Research, 2026.

[6] Cloud Security Alliance. "Hugging Face's Autonomous AI Agent Breach." Cloud Security Alliance AI Safety Initiative, July 2026.

[7] Cloud Security Alliance. "Autonomous Action Runtime Management (AARM)." Cloud Security Alliance, 2026.

[8] Cloud Security Alliance. "Marimo RCE: LLM Agents as Post-Exploitation Tools." Cloud Security Alliance AI Safety Initiative, June 2026.

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

