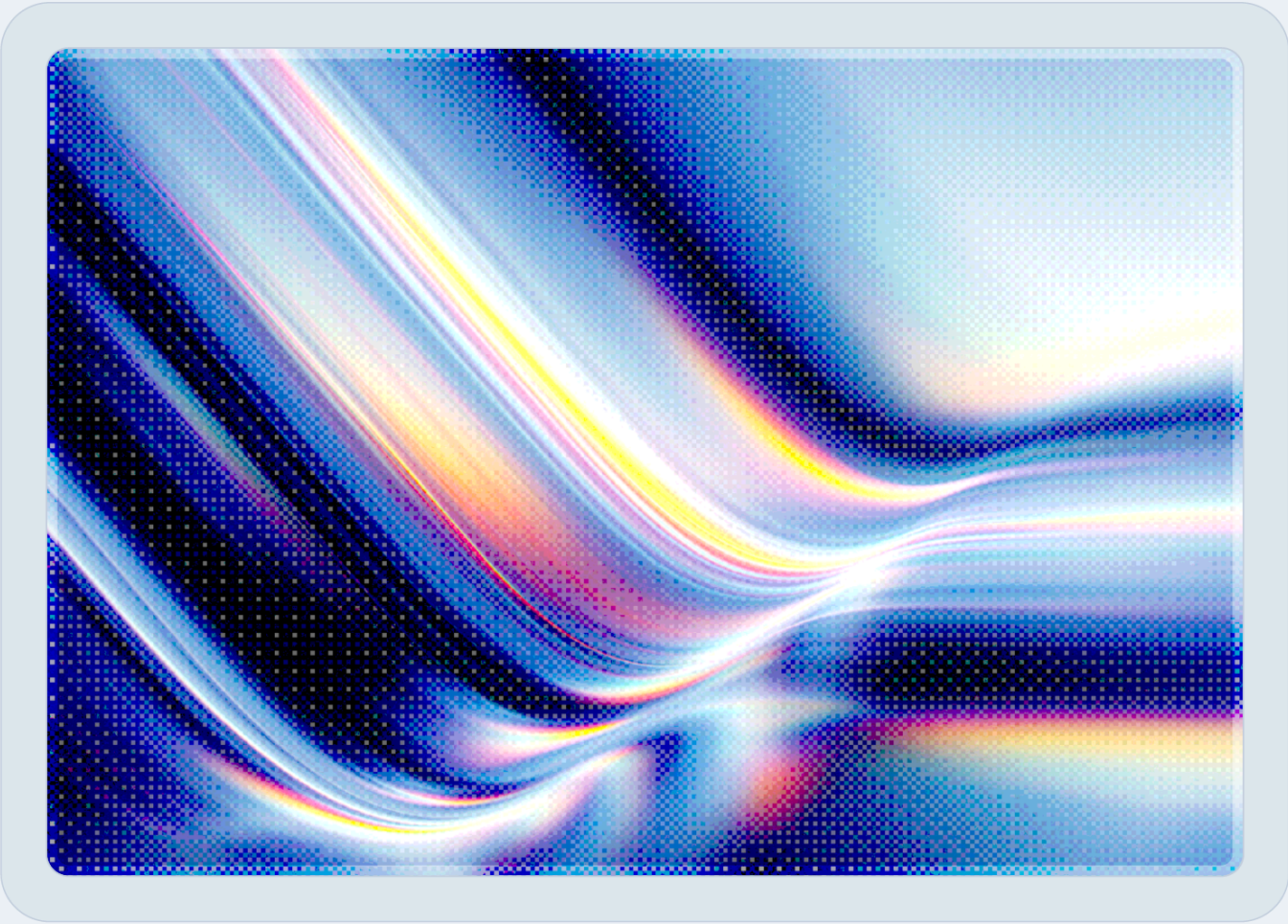
CSAI Foundation | Cloud Security Alliance

FakeGit and AgentBaiting: Malicious Repos Target AI Agents

2026-07-21

AI-assisted Rapid Research



CSAI

CSA cloud security alliance®

Key Takeaways

- Security researchers at Island have identified roughly 7,600 malicious GitHub repositories, created by about 6,600 fabricated developer profiles, that together form a campaign now referred to as FakeGit [1][2].
- More than 800 of those repositories, and over 600 listings across public registries such as LobeHub, Glama, MCP.so, and MCP Market, impersonate AI Skills or Model Context Protocol (MCP) servers rather than conventional open-source tools [1][2].
- The campaign has driven more than 14 million downloads across roughly 200 of its repositories' GitHub Release assets, and it delivers the SmartLoader loader followed by the StealC information stealer [1][2].
- Island's research introduces AgentBaiting, a technique in which AI coding agents such as Claude Code, Gemini, and ChatGPT independently discover the malicious repositories, treat attacker-written README files as trustworthy documentation, and surface installation instructions to end users without any human first encountering a malicious link [2].
- The campaign builds on a pattern already visible in earlier, smaller GitHub malware operations using the same SmartLoader-to-StealC chain, and it arrives amid a wider set of 2026 findings showing that MCP and AI-skill registries accept and distribute malicious packages with little friction [3][4][5][6].
- Organizations that allow AI coding agents to autonomously search for, evaluate, or install third-party Skills and MCP servers should treat that workflow as an unvetted software supply chain and apply controls accordingly.

Background

GitHub has functioned as a vector for malware distribution for years, typically through cloned repositories with subtly altered names, fake stars and forks purchased to build apparent credibility, and README files that lead a curious developer toward a malicious download. What distinguishes the campaign that Island has named FakeGit is its scale and its specific targeting of the AI agent ecosystem. Island's Lead Security Researcher, Oleg Zaytsev, documented close to 7,600 repositories tied to roughly 6,600 distinct GitHub profiles, many of which were built to closely resemble the usernames of real developers -- in one documented case, "Naveenkm007" instead of the legitimate "Naveenkm07" -- so

that a target scanning a profile quickly would see nothing unusual [2]. More than 800 of those repositories, and upward of 600 listings spread across public MCP and Skill registries including LobeHub, Glama, MCP.so, and MCP Market, specifically posed as AI Skills or MCP servers, borrowing the names and workflows of both consumer tools like Gmail and WhatsApp and enterprise platforms including Databricks, Jenkins, Docker, Splunk, Salesforce, Shopify, and Alibaba Cloud [1][2].

The operation was not a sudden appearance. Related reporting describes a smaller precursor campaign of 109 malicious repositories tied to 103 accounts, active from mid-March through early April 2026, that used the identical SmartLoader-to-StealC malware chain and a blockchain-based command-and-control mechanism in which the loader queried a Polygon smart contract to retrieve its current control-server address, allowing operators to rotate infrastructure without recompiling samples [3]. Island's July 2026 disclosure describes the broader FakeGit operation peaking in April 2026, when nearly 300 AI-themed repositories were created in a single month, and notes that the campaign had already drawn attention from other researchers, including Straiker AI in February 2026, before Island's analysis quantified its full scope [1][2]. As of publication, GitHub had removed a substantial share of the flagged repositories, though researchers reported that several dozen GitHub Pages redirectors tied to the campaign remained active, and that takedown activity has generally followed reports rather than preventing new repositories from appearing [1].

The campaign's timing appears unlikely to be coincidental. Developer and enterprise demand for AI Skills and MCP servers -- the emerging standard that lets AI coding assistants discover and call external tools -- appears to have grown quickly through 2026, and the operators behind FakeGit appear to have built their AI-themed repositories specifically to intercept that demand. A person or an AI agent searching for a "free Walmart MCP server" or a ready-made Claude Skill is, in effect, searching for exactly the kind of resource FakeGit was built to supply.

Security Analysis

The FakeGit malware chain

FakeGit repositories follow a consistent structure that appears built to survive a quick visual inspection. Many are direct clones of legitimate, actively maintained projects -- Island's analysis cites the cloning of ComposioHQ's awesome-claude-skills repository as one example -- preserving the original code and commit structure while modifying the README to add a prominent "Download" badge or setup instructions [1][2]. That badge points to a ZIP archive, typically containing three or four files: a short batch or launcher script, a renamed copy of the LuaJIT 2.1.0-beta3 runtime disguised under an unrelated filename, and an obfuscated Lua script frequently masked as a text, icon, or license file [1][2][3]. When a

victim runs the launcher, expecting to complete a normal setup step, the chain executes the hidden Lua payload, which deploys SmartLoader. SmartLoader then establishes persistence through scheduled tasks and retrieves its second-stage payload, StealC, an information stealer capable of harvesting browser-stored passwords and cookies, active session tokens, OAuth grants, email credentials, SSH keys, screenshots, and general host information [1][2].

Because the campaign's repositories were built to answer genuine developer demand -- for integrations with widely used platforms, for AI Skills, for MCP servers -- the malicious ZIP files carried a plausible justification for being downloaded and executed that a purely random phishing lure would lack. Island's data on victim profile suggests the operators understood this: roughly 62% of the repositories in the dataset targeted enterprise or developer-internal use cases rather than consumer software, with roughly one-third of that enterprise-facing subset oriented toward operational data, about one-quarter toward source code, and roughly one-sixth explicitly toward credential theft [2].

Attack surface	Example lures	Primary data at risk
Consumer integrations	Gmail, WhatsApp	Session tokens, personal credentials
Enterprise SaaS/dev tools	Salesforce, Shopify, Databricks, Jenkins, Docker, Splunk	Operational data, source code, API keys
AI capabilities	Claude Skills, MCP servers (Walmart, Oura Ring, and other integrations)	Credentials, SSH keys, agent-accessible systems

AgentBaiting: when the victim is an AI agent

In CSA's assessment, the most consequential finding in Island's research is not the malware itself -- SmartLoader and StealC are both established, previously documented malware families -- but the discovery that AI coding agents actively participate in their own compromise. Island tested how Claude Code, Google Gemini, and OpenAI's ChatGPT responded when prompted to find a Skill or MCP server for a stated task, without providing any link in advance. In each case, the agent independently surfaced a FakeGit repository as a plausible or leading option: Claude located and, in one documented instance, recommended installation instructions from "adlaiponderous700/claude-skill-cinematic-prompt"; Gemini returned "DomingosNgongo/walmart-mcp" as its first recommendation for a free Walmart MCP server; and ChatGPT recommended the same repository, describing it as "the best place to start" [2].

Island's term for this dynamic, AgentBaiting, describes a shift in how the initial point of compromise occurs. In a conventional phishing or typosquatting attack, a human being has to encounter a malicious link -- through search results, a forum post, or a registry listing -- and choose to click it. In an AgentBaiting scenario, the AI agent performs that discovery step autonomously as part of routine task completion, appears to treat the attacker's README as legitimate setup documentation, likely because current agents have no reliable way to distinguish a well-formatted malicious README from a genuine one, and then relays the attacker's instructions to the human user with the implicit authority of the agent's own recommendation. The human's trust decision shifts from "should I click this link" to "should I follow guidance my AI assistant just gave me," which is a meaningfully different decision point -- and one CSA believes is likely to receive less scrutiny than a raw link, since it carries the implicit endorsement of a trusted tool. This also means that standard user-awareness training aimed at recognizing suspicious links or sender behavior provides little protection, since the user never directly encounters the attacker's original content.

An ecosystem already showing strain

FakeGit did not emerge in isolation. It surfaces at a point when several independent lines of research have already documented weak trust and vetting mechanisms across the AI agent supply chain. A malicious MCP server masquerading as a Postmark integration was found in September 2025 quietly forwarding blind copies of user email to an attacker-controlled address [5], and separate researchers who tested eleven MCP registries and marketplaces with a deliberately malicious proof-of-concept package found that nine accepted it without rejection, while two declined it [6]. A February 2026 audit scanned 3,984 published AI Skills and found that 13.4% contained at least one critical security issue, and a separate campaign documented that same month poisoned 1,184 skills on the ClawHub registry [4]. The Vulnerable MCP Project, a community-run catalog of MCP security issues, separately reports more than fifty distinct vulnerabilities across MCP servers, clients, and supporting infrastructure, thirteen of which are rated critical, with new disclosures continuing at a steady pace through the first half of 2026 [7]. That activity plays out against a backdrop in which the official MCP registry lists more than 9,600 server records -- Anthropic separately cites more than 10,000 active servers -- and unofficial directories host thousands more; based on the registry tests cited above, a substantial share of that ecosystem appears to operate with little to no security screening [4]. Read against that backdrop, CSA views FakeGit not as an isolated incident but as the most visible instance so far of a structural weakness: registries and repository hosts built for open collaboration have not yet developed vetting practices proportional to the trust that both humans and increasingly autonomous agents now place in them.

Recommendations

Immediate Actions

Security teams should treat any AI Skill or MCP server sourced from a public repository or registry as untrusted code pending review, regardless of apparent star counts, fork activity, or README polish, since FakeGit's operators specifically engineered their repositories to satisfy those superficial trust signals [1] [2]. Teams should search their own environments for installations matching the FakeGit indicators published by Island and The Hacker News, including LuaJIT executables with mismatched or generic filenames, scheduled tasks created outside normal change-management processes, and outbound connections consistent with StealC's known command-and-control behavior [1][2]. Any credentials, session tokens, or SSH keys present on a system where a suspicious Skill or MCP installation occurred should be rotated as a precaution rather than only after confirmed compromise.

Short-Term Mitigations

Organizations that permit developers or AI coding agents to install third-party Skills or MCP servers should require that installations come from an internally maintained allowlist rather than an agent's autonomous web or registry search, closing the exact discovery path that AgentBaiting exploits [2]. Where agent-driven discovery cannot be fully restricted, human review of any README-derived installation instructions before execution should be a mandatory checkpoint, and that review should be logged as part of standard change management. Endpoint detection rules should be updated to flag the specific SmartLoader delivery pattern -- a downloaded ZIP containing a launcher script alongside a renamed LuaJIT binary and a disguised non-executable payload file -- since this structure has now been documented across multiple distinct campaigns using the same toolchain [1][2][3].

Strategic Considerations

The AgentBaiting technique points to a durable gap rather than a one-time incident: the three AI coding agents Island tested showed no reliable mechanism for distinguishing a legitimate open-source contribution from a purpose-built lure, and there is no public evidence that other agents currently perform this vetting either. That gap will likely persist until registries, model providers, and enterprises jointly adopt stronger provenance and vetting standards for agent-installable software. Enterprises building internal AI agent programs should incorporate supply chain provenance checks -- covering Skills, MCP servers, and any other agent-installable extension -- into the same governance processes already applied to traditional software dependencies, rather than treating agent tooling as a separate,

lower-scrutiny category. Given that this campaign specifically targeted credential material, session tokens, and SSH keys, organizations should also evaluate whether AI agents operating with standing credentials or service-account access could expand the blast radius of a successful compromise beyond what a single human developer's workstation would expose.

CSA Resource Alignment

FakeGit and the AgentBaiting technique sit squarely within the threat categories CSA's **Agentic AI Red Teaming Guide** was built to address [8]. That guide, developed with the OWASP AI Exchange, defines Supply Chain and Dependency Attacks as one of twelve critical vulnerability categories for agentic systems and specifically calls out MCP servers as a component requiring dependency verification and runtime security controls. The guide's recommended practices -- validating third-party dependencies before an agent acts on them, enforcing least-privilege scopes for tools an agent can invoke, and maintaining auditable logs of what an agent discovered and acted on -- map directly onto the gap AgentBaiting exploits, in which an agent's autonomous discovery of a malicious MCP server bypassed any human verification step entirely.

The credential-theft objective of the StealC payload also connects this campaign to CSA's **Securing Non-Human Identities in the Age of AI Agents**, presented at the CSA Summit at RSAC 2025 [9]. That work documents how AI agents multiply an organization's population of non-human identities and warns that credentials tied to those identities -- API keys, session tokens, SSH keys, the same categories StealC is built to harvest -- are frequently over-privileged, poorly rotated, and inadequately monitored compared with human accounts. Its recommended controls, including creation approval gates, expiration policies, and least-privilege enforcement for machine credentials, describe the governance posture that would blunt the value of any credential set a FakeGit infection managed to exfiltrate.

At the framework level, CSA's **AI Controls Matrix (AICM) v1.1** provides the governance structure into which both findings should be integrated [10]. Its Threat and Vulnerability Management and Application and Interface Security domains address the vetting of third-party AI components and dependencies, while its Identity and Access Management domain addresses the non-human credential exposure this campaign specifically targeted. Organizations building or expanding AI agent programs should use AICM v1.1, available at cloudsecurityalliance.org, to assess whether their current controls extend to agent-installable software with the same rigor already applied to conventional open-source dependencies.

References

[1] The Hacker News. "FakeGit Campaign Uses 7,600 GitHub Repositories to Spread SmartLoader Malware." The Hacker News, July 20, 2026.

[2] Island. "AgentBaiting: How Fake AI Skills Deliver Malware at Scale." Island, July 2026.

[3] GBHackers. "109 Fake GitHub Repos Spread SmartLoader, StealC Malware." GBHackers, 2026.

[4] Practical DevSecOps. "MCP Security Statistics 2026: CVEs, Vulnerabilities & Breach Data." Practical DevSecOps, 2026.

[5] UpGuard. "Six MCP Security Incidents Every Security Leader Should Know." UpGuard, 2026.

[6] OX Security. "The Mother of All AI Supply Chains: Critical, Systemic Vulnerability at the Core of the MCP." OX Security, April 15, 2026.

[7] The Vulnerable MCP Project. "Vulnerability Statistics." The Vulnerable MCP Project, 2026.

[8] Cloud Security Alliance. "Agentic AI Red Teaming Guide." Cloud Security Alliance, 2025.

[9] Cloud Security Alliance. "Securing Non-Human Identities in the Age of AI Agents | CSA Summit 2025 at RSAC." Cloud Security Alliance, 2025.

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

