CSAI Foundation

Cloud Security Alliance AI Safety Initiative

PHANTOMPULSE: Blockchain C2 RAT via Obsidian Plugin Abuse

REF6598 Campaign Exploits Legitimate Plugin Ecosystem and On-Chain C2 to Target Finance and Crypto Professionals

Unofficial AI-assisted Research

2026-04-16

Key Takeaways

- Elastic Security Labs has documented campaign REF6598, a targeted social engineering operation that abuses the Obsidian note-taking application's community plugin ecosystem to deliver PHANTOMPULSE, a previously undocumented remote access trojan (RAT) with blockchain-based command-and-control (C2) resolution, against individuals in the financial and cryptocurrency sectors [1].
- The attack does not exploit a software vulnerability in Obsidian. Instead, it weaponizes legitimate application functionality: attackers pre-stage a malicious shared vault, instruct victims to enable community plugin sync, and the Shell Commands plugin executes arbitrary PowerShell or shell payloads automatically upon vault open — with no further interaction required from the victim [1][2].
- PHANTOMPULSE resolves its C2 endpoint by querying the Ethereum, Base, and Optimism blockchains through public Blockscout APIs, decoding an XOR-encrypted URL from on-chain transaction input data tied to a hardcoded wallet address. Because the malware trusts any sender's transaction without verification, defenders who extract the wallet address and XOR key from the binary can sinkhole all infected hosts by posting a competing, more recent transaction [1].
- PHANTOMPULSE shows multiple indicators consistent with LLM-assisted development — unusually verbose step-numbered debug strings and a polished "Phantom Panel" C2 interface — though these individually are not definitive. Their combination aligns with a pattern observed in other recently analyzed AI-assisted malware families [1][3].
- Both Windows and macOS systems are targeted: the Windows chain deploys PHANTOMPULSE via a reflective loader (PHANTOMPULL); the macOS chain uses an obfuscated AppleScript dropper and uses a public Telegram channel as a dead-drop resolver to retrieve a backup C2 domain when the primary hardcoded infrastructure is unavailable [1].
- Organizations in finance, cryptocurrency, and adjacent sectors should treat third-party productivity applications — particularly those with community plugin ecosystems and cloud-sync capabilities — as meaningful attack surface and apply the same supply-chain scrutiny to synced vault configurations as to software packages [2][4].

Background

On or around mid-April 2026, Elastic Security Labs published a detailed analysis of REF6598, a campaign discovered during active endpoint detection response on a customer environment [1]. The campaign targets professionals in the financial and cryptocurrency sectors, with attackers impersonating venture capital firms to initiate contact via LinkedIn before shifting conversations to Telegram, where multiple fabricated personas bolster the illusion of a legitimate firm [1][5]. Victims are asked to use Obsidian — a popular Markdown-based note-taking and knowledge management application — as the firm's "management database," and are provided with credentials to an attacker-controlled cloud vault framed as a shared business dashboard [1][2].

Obsidian is an Electron-based desktop application with a rich community plugin marketplace that allows users to add code-execution capabilities to their note-taking environment. The Shell Commands plugin is among the most operationally significant of these plugins from a security perspective, enabling users to bind arbitrary shell or PowerShell commands to Obsidian triggers, including vault-open events. This functionality is intentional and documented; it makes Obsidian a powerful personal automation environment. It also, as REF6598 illustrates, creates a category of risk that many conventional security controls are not configured to detect — endpoint detection that evaluates Obsidian as a trusted application without inspecting its child process tree will miss this entirely, though behavioral platforms that monitor process lineage may flag it [1][2]. The attack embeds its payload in an application's configuration file, where it executes through the application's own runtime and produces child processes that appear to originate from a trusted Electron binary.

The broader threat context adds urgency to this finding. REF6598 is not the first campaign to route malware delivery through a productivity application or AI-adjacent tool ecosystem. The GlassWorm campaign, reported in March 2026, used compromised maintainer accounts on npm, PyPI, GitHub, and the Open VSX marketplace to distribute stealers whose C2 infrastructure resolved via Solana blockchain transactions used as a dead-drop mechanism [6]. North Korean-linked group UNC5342 adopted EtherHiding — embedding malicious JavaScript payloads within Ethereum and BNB Smart Chain smart contracts — as part of the ongoing Contagious Interview social engineering campaign since at least February 2025, with Google Threat Intelligence Group documenting the technique's adoption by the actor [7]. What distinguishes REF6598 is the convergence of these trends: blockchain-based C2 evasion, AI-assisted malware development, and the abuse of a legitimate consumer productivity application's extension model, all orchestrated through a methodical social engineering operation targeting high-value individuals in asset-rich sectors.

Security Analysis

The Obsidian Vault as Attack Surface

The abuse of Obsidian in REF6598 exposes a structural security gap that is neither unique to Obsidian nor easily resolved by the application's developers. The Obsidian model separates the application itself — distributed through standard channels and reviewed at the binary level — from community plugins, which are independently developed and published to a community marketplace without the same review controls that govern, for example, browser extension stores. When a user enables community plugin sync, Obsidian automatically replicates the vault's plugin configuration across devices from the synced cloud source [1][2]. In the REF6598 scenario, this means the malicious Shell Commands configuration file that the attacker pre-staged in the vault travels to the victim's machine as soon as sync is enabled, with no visible installation prompt and no secondary confirmation that the synced configuration includes execution-capable plugin settings.

The Shell Commands plugin's data file contains the commands it will execute on each configured trigger, including a vault-open event. Once the attacker's pre-staged configuration is synced, opening the vault causes the plugin to run Base64-encoded PowerShell (`powershell.exe -ExecutionPolicy Bypass`) that contacts the attacker's staging server and downloads the first-stage script [1]. The victim receives no visual cue distinguishing this from normal vault loading behavior. This sequence — shared credentials, cloud sync, plugin configuration propagation, silent execution — uses nothing but Obsidian's intended feature set. There is no exploit in the conventional sense.

Based on Elastic's analysis [1], behavioral detection of PowerShell spawning from Electron runtimes is the most readily actionable technical control at the delivery layer — though training, vault-sharing governance, and egress filtering can interrupt earlier stages of the attack chain. Elastic reports that its Elastic Defend product blocked the attack at the PowerShell execution stage during the victim engagement from which this campaign was discovered [1].

PHANTOMPULSE Technical Capabilities

The Windows attack chain proceeds through two additional stages after the initial PowerShell execution. The first-stage PowerShell script (`script1.ps1`) uses Windows BitsTransfer to download a 64-bit Windows PE executable identified by Elastic researchers as PHANTOMPULL, which sends status telemetry to a C2 endpoint using a coded prefix system (`GFILE FOUND ON PC`, `RDOWNLOAD ERROR`, `GLAUNCH SUCCESS`) that tracks delivery success in near real time [1]. PHANTOMPULL then decrypts an embedded payload using AES-256-CBC with a hardcoded key and initialization vector, loads it entirely into memory via reflective PE injection through `DllRegisterServer`, and uses timer-queue callbacks with a 50-millisecond delay to complicate sandbox detection. The final payload never touches disk in its decrypted form, and runtime API resolution uses a djb2 hashing algorithm throughout to resist static analysis [1].

The payload — PHANTOMPULSE — is a fully featured 64-bit backdoor that communicates over HTTPS using a structured REST API. Its command set spans the primary capabilities an operator would require for sustained, covert access: `inject` (shellcode, DLL, or PE process injection via module stomping); `drop` (file staging and execution); `screenshot` (screen capture with upload); `keylog` (keyboard logging with start/stop control); `elevate` (UAC bypass via COM elevation moniker); `downgrade` (SYSTEM to elevated administrator demotion for specific operations); and `uninstall` (complete persistence and artifact removal) [1]. The C2 REST endpoints follow a consistent telemetry-style naming convention (`/v1/telemetry/report`, `/v1/telemetry/tasks/<id>`, `/v1/telemetry/upload/`, `/v1/telemetry/keylog/`) that, combined with the verbose debug strings present throughout the binary, suggests the operator used an LLM to generate or substantially augment the codebase — a development pattern that is increasingly visible in recently analyzed malware families [1][3].

On macOS, the attack path diverges after the initial Obsidian trigger. An obfuscated AppleScript dropper creates a persistent LaunchAgent at `~/Library/LaunchAgents/com.vfrfeufhtjpwgray.plist` with both `KeepAlive` and `RunAtLoad` directives enabled, ensuring execution survives reboots. The AppleScript itself uses character-level string obfuscation (`ASCII character` and `character id` calls with decoy variables and fragmented string concatenation) to evade static detection. When the primary hardcoded C2 domain (`0x666[.]info`) is unavailable, the macOS implant falls back to scraping a public Telegram channel (`t[.]me/ax03bot`) for a backup domain, adding a second layer of C2 resilience without requiring any server-side infrastructure changes [1].

Blockchain-Based C2: Architecture and Exploitable Design Flaw

PHANTOMPULSE applies the technique the security community has designated "EtherHiding" — the use of public blockchain infrastructure as a resilient, censorship-resistant dead-drop resolver for malware C2 endpoints [12]. Rather than encoding payloads or full instructions on-chain (as earlier EtherHiding implementations did), PHANTOMPULSE uses the blockchain specifically for C2 URL discovery: the malware queries the Blockscout API across three Ethereum-compatible networks — Ethereum L1 (`eth.blockscout[.]com`), Base L2 (`base.blockscout[.]com`), and Optimism L2 (`optimism.blockscout[.]com`) — and retrieves the most recent inbound transaction to a hardcoded wallet address (`0xc117688c530b660e15085bF3A2B664117d8672aA`, XOR-encrypted in the binary) [1]. The input data of that transaction is hex-decoded and XOR-decrypted using the wallet address as the key; if the result begins with "http," PHANTOMPULSE treats it as the live C2 URL and initiates communication [1].

This mechanism offers the operator genuine resilience advantages. Blockchain transactions are immutable and globally replicated, making the resolution mechanism impossible to take down through conventional infrastructure disruption. Rotating the C2 endpoint requires only posting a new transaction to the hardcoded wallet. Elastic observed two rotations during its investigation: a Cloudflare Tunnel endpoint (`thoroughly-publisher-troy-clara[.]trycloudflare[.]com`) active on February 12, 2026, and a dedicated panel domain (`panel.fefea22134[.]net`) observed from February 19, 2026, with a valid Let's Encrypt certificate and Cloudflare-fronted hosting on infrastructure belonging to Polish provider MEVSPACE [1].

However, the implementation contains a design flaw with significant defensive implications: PHANTOMPULSE selects the most recent transaction to the wallet address without verifying that the transaction was sent by the wallet's controller. Any third party that extracts the wallet address and XOR key from a PHANTOMPULSE binary — both of which are recoverable through standard binary analysis — can submit a competing transaction to the same address, pointing all active PHANTOMPULSE infections to an analyst-controlled sinkhole. The most recently mined transaction wins, meaning a sinkhole operator who monitors the wallet and submits a counter-transaction whenever the actor rotates C2 can maintain persistent redirection of the botnet [1]. This flaw is structurally similar to the "blockchain squatting" technique used against early EtherHiding implementations and represents a meaningful defender-side leverage point that security teams and law enforcement should consider in active response scenarios.

AI-Assisted Development Indicators

Elastic Security Labs researchers noted several characteristics in PHANTOMPULSE that are consistent with LLM-assisted code generation. The binary's debug strings are unusually verbose and follow a structured step-numbering pattern (`[STEP 1]`, `[STEP 1/3]`, `[STEP 2/3]`) throughout the codebase, with self-documenting comments embedded in what would normally be silent operational code [1]. The "Phantom Panel" web interface exposed by the C2 server exhibits a design consistency and polish atypical of custom criminal tooling, where interfaces are more commonly minimal or cobbled together from open-source components [1][3]. While these indicators are not individually definitive, their combination — particularly the structured debug verbosity — aligns with a pattern that has been observed in other AI-assisted malware samples analyzed in 2025 and 2026, including PowerShell-based initial-access tools where modular layout and in-line documentation were assessed as markers of LLM generation rather than traditional operator craft [8].

The significance extends beyond attribution curiosity. AI-assisted malware development accelerates the production of feature-complete, well-structured malicious tools by operators who may lack the software engineering depth to build them from scratch. PHANTOMPULSE's combination of reflective loading, module stomping, blockchain-based C2, and cross-platform targeting represents a level of technical breadth that, if assembled manually, would indicate a well-resourced adversary. That this capability may now be within reach of actors with strong social engineering skills but only moderate technical depth is a material shift in the threat landscape that security teams should factor into their adversary modeling.

Recommendations

Immediate Actions

Organizations in the financial services, investment, and cryptocurrency sectors — the primary targets of REF6598 — should communicate immediately to staff that Obsidian vault-sharing schemes used as part of unsolicited business introductions via LinkedIn or Telegram are a confirmed social engineering lure. Security awareness programs should be updated to include this pattern, with explicit guidance that accepting cloud vault credentials from external parties and enabling community plugin sync carries material malware delivery risk, regardless of how legitimate the accompanying business context appears.

Security operations teams should hunt for evidence of PHANTOMPULSE activity in their environments using the following indicators documented by Elastic Security Labs [1]:

Type	Value	Purpose
SHA-256	`70bbb38b70fd836d66e8166ec27be9aa8535b3876596fc80c45e3de4ce327980`	PHANTOMPULL loader (syncobs.exe)
SHA-256	`33dacf9f854f636216e5062ca252df8e5bed652efd78b86512f5b868b11ee70f`	PHANTOMPULSE final payload
IP	`195.3.222[.]251`	Staging server (initial PowerShell/loader)
Domain	`panel.fefea22134[.]net`	PHANTOMPULSE C2 panel (active Feb 2026)

Type	Value	Purpose
Domain	`0x666[.]info`	macOS implant primary C2
Ethereum wallet	`0xc117688c530b660e15085bF3A2B664117d8672aA`	On-chain C2 resolver (XOR-encrypted in binary)
Ethereum wallet	`0x38796B8479fDAE0A72e5E7e326c87a637D0Cbc0E`	Funding wallet (~50 transactions observed)

Endpoint detection platforms should be checked for alerts on the following behavioral patterns, which Elastic has released as detection logic [1][12]; SOC Prime has published additional community detection content for this campaign [11]:

- Suspicious child process creation from `Obsidian.exe` or `Obsidian` (shell interpreters: `sh`, `bash`, `zsh`, `powershell.exe`, `cmd.exe`)
- Presence of `obsidian-shellcommands` in file paths on monitored endpoints
- Network connections originating from Obsidian or its child processes to public Blockscout API endpoints
- LaunchAgent creation at `~/Library/LaunchAgents/com.vfrfeufhtjpwgray.plist` on macOS

Short-Term Mitigations

Endpoint security policies should explicitly monitor Electron-based applications for anomalous child process spawning. This is a general pattern — not specific to Obsidian — because the Electron runtime is commonly used in developer tools, communication applications, and productivity software, all of which may have community plugin ecosystems or scripting capabilities that can be abused through a similar vault-sync-style delivery model. Process creation rules that flag shell interpreter invocations descending from Electron parent processes, particularly those accompanied by Base64 command-line arguments or external network connections, are well-positioned to detect this class of attack across applications, though determined actors with knowledge of detection logic may adapt delivery to evade specific rules.

Most enterprise software governance policies address software installation but may not extend to the installation of plugin configurations that arrive via cloud sync within an already-approved application — a gap REF6598 exploits directly. The REF6598 campaign illustrates this precisely: the application was approved; the configuration sync was a feature of the approved application; and the malicious component — the plugin configuration — was not separately reviewed. Governance frameworks should be extended to address configuration-level trust assumptions within approved cloud-synced applications, treating vault configurations that include execution-capable plugin settings with the same scrutiny as software package installations.

For organizations in high-risk sectors (finance, cryptocurrency, asset management), proactive network controls should block outbound connections to all three Blockscout API endpoints (`eth.blockscout[.]com`, `base.blockscout[.]com`, `optimism.blockscout[.]com`) as well as any other public blockchain explorer APIs that are not required for business operations. This degrades the blockchain-based C2 resolution mechanism and forces the malware to fall back to hardcoded infrastructure, which is more easily blocked through traditional indicator-based controls.

Strategic Considerations

The REF6598 campaign illustrates a supply-chain trust assumption that organizations have not systematically addressed: community plugin ecosystems for productivity applications. While developer tooling and AI assistant ecosystems have attracted significant security scrutiny recently — package manager security, MCP server vetting, VS Code extension reviews — community plugin ecosystems in

consumer productivity applications have received comparatively less systematic attention from enterprise security programs. Applications such as Obsidian and Logseq expose users to community-developed plugins that can execute shell commands and access local file systems, representing the most direct analog to REF6598. Other productivity platforms with third-party integration ecosystems — including those relying on API-based connections — present related but distinct risks that warrant separate evaluation. Security programs should formally enumerate these applications in their environments and assess the degree to which plugin installation and configuration sync policies are defined and enforced.

At a strategic level, the AI-assisted development dimension of PHANTOMPULSE warrants attention from threat modeling programs. Security teams have generally treated the presence of advanced tradecraft — reflective loading, blockchain-based C2, cross-platform implants — as a signal of nation-state or well-resourced criminal actors, and calibrated their response posture accordingly. PHANTOMPULSE's apparent AI-assisted construction suggests that this calibration may warrant revisiting: if advanced technical capabilities are increasingly accessible through LLM tooling, capability alone may be a less reliable proxy for resourcing level than it once was. Organizations that are not high-value targets from a nation-state perspective may still face adversaries equipped with these techniques through AI-assisted development.

CSA Resource Alignment

REF6598 maps directly to several threat categories and control domains addressed by CSA's security frameworks, and security teams responding to this campaign should consult the relevant guidance in each.

The MAESTRO framework for agentic AI threat modeling addresses the Layer 7 (Agent Ecosystem) risk of malicious plugin or tool installation, a threat category that REF6598 instantiates in a consumer productivity context. MAESTRO's taxonomy of ecosystem-layer threats — compromised plugin registries, malicious skill injection, tool configuration manipulation — applies directly to the Obsidian Shell Commands abuse vector. Organizations deploying AI agents with plugin or skill ecosystems (including MCP servers, VS Code extensions, and agent tool registries) should treat REF6598 as a concrete threat model case study for the risks MAESTRO identifies at this layer [9].

The CSA AI Controls Matrix (AICM) provides 243 control objectives across 18 security domains, with specific applicability to the social engineering and supply-chain dimensions of REF6598. AICM's domains addressing data integrity, supply chain provenance, and deployment security are directly relevant to the vault-sync delivery mechanism, which exploits the assumption that synced configuration data from a trusted source is safe to execute. Organizations implementing AICM controls should extend their supply chain coverage to include third-party plugin configurations and cloud-synced application settings [10].

CSA's Zero Trust guidance is relevant to the vault credential sharing vector at the center of REF6598's social engineering lure. The campaign depends on victims trusting externally provided credentials and treating the resulting synced environment as benign — an assumption that Zero Trust architecture explicitly rejects. Applying least-privilege and continuous verification principles to productivity application configurations, including treating community plugin sync as a privileged action requiring explicit authorization, operationalizes Zero Trust at the endpoint in a way that directly mitigates this attack vector.

The STAR (Security Trust Assurance and Risk) program provides a framework for assessing the security posture of cloud service providers, which is relevant here as organizations evaluate whether third-party cloud vault and sync infrastructure for productivity applications meets their risk thresholds. As REF6598 demonstrates, the risk of cloud-synced productivity tool configurations includes not only data exposure but active payload delivery — a dimension that STAR assessments for these services should explicitly address.

References

[1] Elastic Security Labs. "Phantom in the vault: Obsidian abused to deliver PhantomPulse RAT." Elastic Security Labs, April 2026.

[2] CyberSecurityNews. "Hackers Weaponize Obsidian Shell Commands Plugin to Launch Cross-Platform Malware Attacks." CyberSecurityNews, April 2026.

[3] Penligent AI. "Obsidian Shell Commands Abuse Shows a New Malware Playbook." Penligent AI, April 2026.

[4] GBHackers. "Hackers Exploit Obsidian Plugin to Deploy Cross-Platform Malware." GBHackers, April 2026.

[5] CoinTelegraph. "Crypto Users Warned of Scam on Notes App Obsidian." CoinTelegraph, April 2026.

[6] The Hacker News. "GlassWorm Malware Uses Solana Dead Drops to Deliver RAT and Steal Browser, Crypto Data." The Hacker News, March 2026.

[7] Google Cloud Blog / GTIG. "DPRK Adopts EtherHiding: Nation-State Malware Hiding on Blockchains." Google Cloud Blog, October 2025.

[8] BleepingComputer. "Konni Hackers Target Blockchain Engineers with AI-Built Malware." BleepingComputer, 2025.

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

[10] Cloud Security Alliance. "AI Controls Matrix." CSA, 2025.

[11] SOC Prime. "PhantomPulse RAT via Malicious Obsidian Vaults." SOC Prime Active Threats, April 2026.

[12] Elastic Security. "Potential EtherHiding C2 via Blockchain Connection." Elastic Security Detection Rules, 2025.

