Hive Pro

Threat Level

Amber

Hiveforce Labs

THREAT ADVISORY

ATTACK REPORT

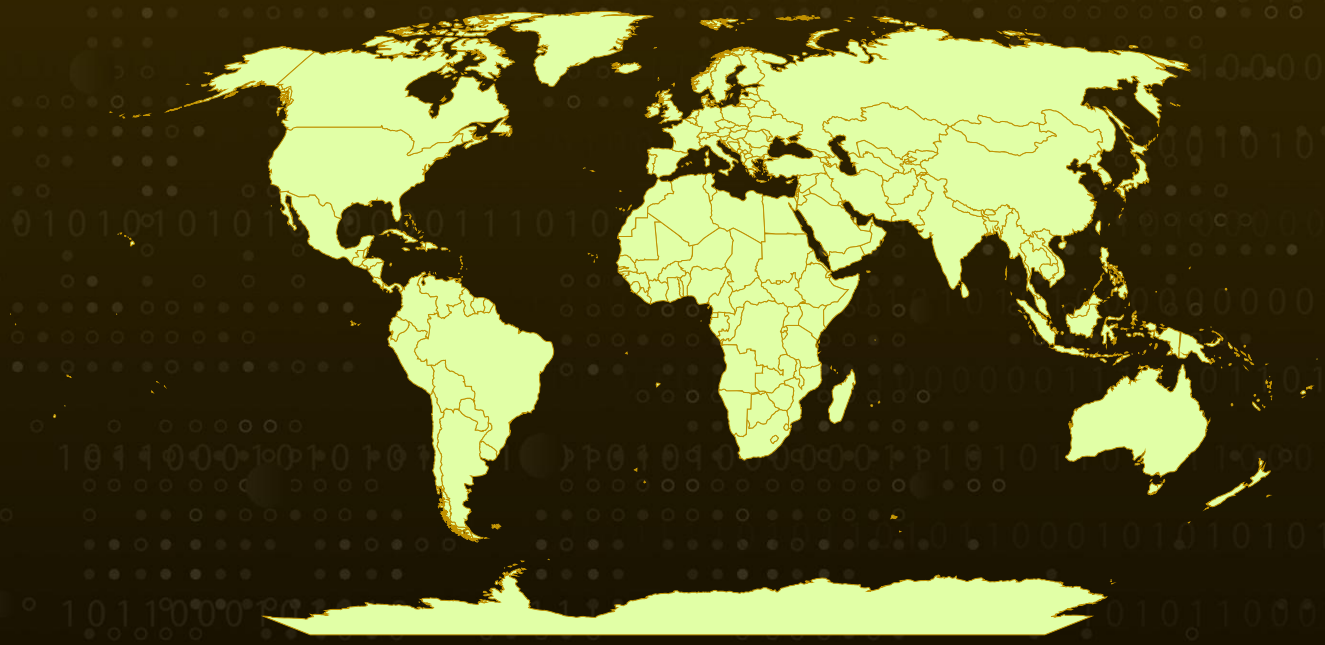
Veil#Drop Abuses Google Blogspot to Deliver Fileless PureLog Stealer

Date of Publication	Admiralty Code	TA Number
July 02, 2026	A1	TA2026184

Summary

First Seen: 2026
Targeted Regions: Worldwide
Targeted Platforms: Windows
Targeted Products: Web browsers (Google Chrome, Microsoft Edge, Mozilla Firefox, Brave, Opera, and other Chromium-based browsers) and cryptocurrency wallets (MetaMask, Exodus, Atomic Wallet, Electrum, Trust Wallet, Coinbase Wallet, Binance Wallet)
Malware: Veil#Drop, PureLog Stealer
Attack: Veil#Drop is a multi-stage, largely fileless delivery framework that ends in the deployment of PureLog Stealer. It begins with a JavaScript file disguised as a document, executed through Windows Script Host, which launches PowerShell download cradles that pull further stages from attacker-controlled Blogspot pages. The chain relies on XOR-obfuscated payloads, dynamically generated URLs, runtime script mutation, and in-memory .NET reflection, with trusted Microsoft-signed binaries used as execution fallbacks. The final PureLog payload harvests browser credentials, cookies, session tokens, cryptocurrency wallet data, and system information before exfiltrating it to remote command-and-control infrastructure.

Attack Regions



Attack Details

#1

There is no exploit here, just a file that lies about what it is. Veil#Drop starts when someone opens a JavaScript file named to look like a document, such as transcript.pdf.js. Because Windows hides known extensions by default, the victim only ever sees "transcript.pdf" and clicks it expecting a harmless PDF. The lures are delivered through legitimate websites that have been quietly compromised, so the operators get to lean on trusted domains and slip past reputation filtering; the delivery itself is assessed, with lower confidence, as either a spear-phishing link or a drive-by download. On that click, Windows hands the file to Windows Script Host, which spawns PowerShell with its execution policy bypassed, and that is where the infection really begins.

#2

From there the chain runs almost entirely in memory. The JavaScript does nothing but launch a PowerShell download cradle, pairing Invoke-RestMethod with Invoke-Expression to pull the next stage straight off an attacker-controlled Blogspot page and borrow Google's reputation as cover. That stage clears the ground: it loads a harmless decoy web page so the victim thinks a document opened, deletes the original .js from Downloads, kills scripting and .NET processes that might interfere, and XOR-decrypts an embedded payload. The decoded loader then rebuilds its next Blogspot URL on the fly, padding the path with a random count of slashes and swapping placeholder strings for random values each run, so no two executions share a hash or a URL signature.

#3

The final loader is where the real payload hides. It carries two XOR-encoded .NET assemblies stored as long runs of decimal values, rebuilds them with a custom Convert-XorDecimalToExe routine (XOR key 47), and runs them directly from memory through Reflection.Assembly::Load() so nothing ever touches disk. If reflection gets blocked, the loader simply works down a list of Microsoft-signed binaries, RegSvcs, InstallUtil, MSBuild, CSC, VBC, ILAsm, CasPol, and AspNet_Compiler, trying each until one succeeds.

#4

At the end of the chain sits PureLog Stealer, a .NET infostealer built to work quietly and take a lot. It pulls saved logins, cookies, session tokens, autofill, browsing history, and card data from Chrome, Edge, Firefox, Brave, Opera, and other Chromium browsers, and goes after wallet material tied to MetaMask, Exodus, Atomic Wallet, Electrum, Trust Wallet, and similar apps. It also fingerprints the host, including machine name, user, OS version, installed software, running processes, hardware, and domain membership, so the operators can tell a throwaway box from something worth their time. The malware never moves through the network itself, but the cookies it lifts can walk straight past MFA, and stolen VPN, Microsoft 365, and cloud credentials are exactly what someone needs to gain a foothold and go deeper later.

#5

Once it has what it wants, PureLog packages the credentials, cookie stores, tokens, wallet data, browser profiles, and system inventory and ships them to command-and-control over encrypted channels. Because the whole thing lives in memory, with no dropped executable, no written DLL, and no leftover script, defenders are mostly left with process lineage, PowerShell script-block logs, memory-resident assemblies, and network telemetry instead of files to scan. The stolen data usually ends up fueling account takeover, cryptocurrency theft, and business email compromise, and it frequently gets resold on criminal markets, pushing the damage well past the first machine.

Recommendations

Alert on Scripting-Engine to PowerShell Process Chains: Treat parent-child relationships such as wscript.exe or cscript.exe spawning powershell.exe as high-fidelity indicators. These chains are uncommon in normal user activity and sit at the very start of the Veil#Drop infection.

Enforce Visible File Extensions and Target User Awareness: Disable the HideFileExt behavior through policy so double-extension lures like transcript.pdf.js are visible, and train users specifically on document-themed JavaScript files delivered from websites.

Scrutinize Blogspot and Blogger Traffic From Scripting Engines: Flag outbound connections to blogspot[.]com and blogger[.]com that originate from powershell.exe, wscript.exe, cscript.exe, or mshta.exe. Legitimate browsing to these domains is normal, but scripting engines reaching them directly should be investigated.

Detect Reflective .NET Assembly Loading: Monitor for PowerShell activity involving Reflection.Assembly::Load, Assembly.Load, Add-Type, dynamic compilation, and ScriptBlock::Create, which are rarely seen in standard workflows and strongly indicate fileless execution.

Baseline and Investigate .NET LOLBIN Abuse: Establish behavioral baselines for RegSvcs, InstallUtil, MSBuild, CSC, VBC, ILAsm, and AspNet_Compiler, and alert when these binaries are spawned by PowerShell, run from user-writable directories, or load assemblies from temporary locations.

Reset Credentials and Invalidate Sessions After Suspected Infection: On any host showing signs of PureLog activity, rotate exposed passwords and forcibly invalidate active sessions, since stolen cookies and tokens can be replayed to bypass standard multi-factor authentication.

Move Toward Phishing-Resistant, Token-Bound Authentication: Because harvested session cookies defeat conventional MFA, adopt phishing-resistant methods and session-binding controls to reduce the value of stolen browser session data.

Restrict Windows Script Host Where Feasible: Disable or tightly constrain the execution of .js files through Windows Script Host in environments that do not require it, removing the framework's primary entry point.

Potential MITRE ATT&CK TTPs

Tactic	Technique	Sub-technique
Initial Access	T1189: Drive-by Compromise	
	T1566: Phishing	T1566.002: Spearphishing Link
Execution	T1059: Command and Scripting Interpreter	T1059.001: PowerShell
		T1059.007: JavaScript
	T1204: User Execution	T1204.002: Malicious File
Persistence	T1547: Boot or Logon Autostart Execution	

Tactic	Technique	Sub-technique
Defense Evasion	T1027: Obfuscated Files or Information	T1027.013: Encrypted/Encoded File
	T1036: Masquerading	T1036.007: Double File Extension
	T1140: Deobfuscate/Decode Files or Information	
	T1218: Signed Binary Proxy Execution	T1218.004: InstallUtil
		T1218.009: Regsvcs/Regasm
		T1218.011: MSBuild
	T1562: Impair Defenses	T1562.001: Disable or Modify Tools
	T1620: Reflective Code Loading	
Credential Access	T1555: Credentials from Password Stores	T1555.003: Credentials from Web Browsers
	T1539: Steal Web Session Cookie	
Discovery	T1082: System Information Discovery	
	T1057: Process Discovery	
Collection	T1005: Data from Local System	
	T1213: Data from Information Repositories	

Tactic	Technique	Sub-technique
Command and Control	T1071: Application Layer Protocol	T1071.001: Web Protocols
	T1105: Ingress Tool Transfer	
	T1573: Encrypted Channel	
Exfiltration	T1041: Exfiltration Over C2 Channel	

Indicators of Compromise (IOCs)

TYPE	VALUE
URLs	hxxps[:]//htlwub00klocate[.]blogspot[.]com/phud[.]dudus[.]docx[.]pdf[.]olp[.]sys, hxxps[:]//cpyzaramay26[.]blogspot[.]com/…/niple[.]docx[.]odp[.]pdf[.]sys
SHA256	b0f550c17a19682ff54bca418ee186ac986d0813e018b317dc0e7aebff5bf054, 6a6a4c29d37732a7af61b2dab8f521306a0cc096974e1a43e0df81cadfffba4a, de6a037dfe2e9f054e8e7695423c0cc388001362e3737271c639fdc1f08f849e, b5396b4034130bbf1fe30234cbd321cac67230b19b620e3f5f6ee9ad8f55dcd3, a048fc039ba6d1e22736c9142998de79445f878136664958f9b11156aaf1b61f, 7fa075ed827095b4531cb35f650ccf6345c3799734e4ed30d9f52e72c0711713, 7e4646d0cf91153653c5e366f98a65aad5ef363e0edeb246c809f53085971453, 3d3342af3608399704d5daf9dc061ad1f8b243531fd9ef8497a10c6a9dd59661

TYPE	VALUE
Filenames	transcript.pdf.js, phud.dudus.docx.pdf.olp.sys, niple.docx.odp.pdf.sys, decoded_1.bin, decoded_2.bin

References

https://www.securonix.com/blog/veildrop-blogspot-hosted-powershell-loader/

What Next?

At **Hive Pro**, it is our mission to detect the most likely threats to your organization and to help you prevent them from happening.

Book a Demo of **HivePro**.

REPORT GENERATED ON

July 02, 2026 • 07:50 AM

More at www.hivepro.com