INTELLIGENCE BRIEFING
Security Command Center
TLP:CLEAR
2026-04-09 13:33 UTC

Microsoft Driver Signing Suspensions Expose Open Source Patch Delivery Gap in Enterprise Environments

GOVERNANCE | MEDIUM | CVSS 5.0

SCC Item ID	SCC-GOV-2026-0013
Type	Governance
Severity	MEDIUM
CVSS Base Score	5.0
Affected Products	WireGuard (Windows driver), VeraCrypt (Windows driver), MemTest86 (Windows driver), Windscribe VPN (Windows driver), Microsoft Windows Hardware Program / Partner Center
Published	2026-04-09T02:46:26
Discovery Source	Rss

Executive Summary

Microsoft's automated account verification process suspended driver-signing credentials for WireGuard, VeraCrypt, MemTest86, and Windscribe VPN without notifying affected maintainers, halting signed kernel-mode driver delivery for these tools on Windows. Enterprises relying on these projects for VPN connectivity, disk encryption, and memory diagnostics may face unsigned driver failures on systems with Driver Signature Enforcement and Secure Boot enabled. The incident exposes a structural supply chain dependency: a single administrative action by Microsoft can fully block update and patch delivery for widely deployed open-source security tooling, with no viable self-service escalation path.

Technical Analysis

Microsoft's Windows Hardware Program (Partner Center) suspended developer accounts for WireGuard (wireguard-windows), VeraCrypt, MemTest86, and Windscribe VPN through an automated verification process. The suspensions revoked Extended Validation (EV) code signing certificates and WHCP signing authority required to produce kernel-mode drivers loadable on Windows 10/11 systems with Driver Signature Enforcement (DSE) and Secure Boot enabled. No CVE is assigned; this is an operational supply chain disruption, not a vulnerability exploitation event. CWE-693 (Protection Mechanism Failure) applies: the signing pipeline that guarantees driver integrity and delivery failed as an administrative control. CWE-1357 (Reliance on Insufficiently Trustworthy Component) applies: affected projects depend entirely on a single vendor's administrative process for software integrity and distribution continuity. MITRE ATT&CK T1553.006 (Subvert

Trust Controls: Code Signing Policy Modification) applies as a risk context, organizations may attempt to weaken DSE or disable Secure Boot as a workaround, creating kernel attack surface. This is not an observed tactic in the incident but a known risk reaction to patch delivery disruption. T1195.002 (Supply Chain Compromise: Compromise Software Supply Chain) contextualizes the broader supply chain risk. Affected maintainers had no functional support escalation path; resolution required press coverage and executive-level intervention at Microsoft. No exploitation is involved. No patch is pending. The risk is operational: enterprises may be running older signed driver versions and cannot receive updates until signing access is restored or re-established.

Action Checklist

1. Containment: Identify all Windows systems in your environment running WireGuard, VeraCrypt, MemTest86, or Windscribe VPN kernel-mode drivers. Confirm which driver versions are currently installed and whether those versions carry valid WHCP signatures. Do not loosen Driver Signature Enforcement (DSE) or disable Secure Boot as a workaround; doing so increases kernel attack surface.
2. Detection: Query your asset inventory or endpoint management platform (SCCM, Intune, or equivalent) for installed versions of WireGuard, VeraCrypt, MemTest86, and Windscribe VPN across all Windows endpoints. Cross-reference installed driver versions against the last known signed release from each project's official repository. Flag any endpoints where automatic update mechanisms may have stalled.
3. Verification: No compromise or malicious payload requires removal. The threat vector is supply chain disruption, not compromise. If unsigned or self-signed driver variants are detected in your environment, treat them as unverified and remove or quarantine until a WHCP-signed version is confirmed available from the official project maintainer.
4. Recovery: Monitor official project repositories (WireGuard GitHub, VeraCrypt SourceForge/GitHub, MemTest86 PassMark, Windscribe official site) for announcements confirming restored signing access and new signed driver releases. Validate signature chains on any new driver packages before deployment. Confirm Secure Boot and DSE remain enforced post-remediation.
5. Post-Incident: Document organizational dependencies on open-source tools that rely on Microsoft's Windows Hardware Program for driver signing. Evaluate whether your software supply chain risk assessment accounts for single-vendor administrative control points. Consider adding monitoring for WHCP signing status changes for critical open-source dependencies as an ongoing third-party risk indicator.

IR / Forensic Enrichment

Triage Priority	STANDARD
Escalation Criteria	Escalate to urgent if any endpoint is found running an unsigned or self-signed variant of wireguard.sys, veracrypt.sys, or the Windscribe kernel driver that was not sourced directly from the official project maintainer, as this indicates a potentially tampered driver entered the environment through an unofficial channel and requires immediate integrity investigation under NIST IR-4 (Incident Handling).

Recovery Notes	Recovery is gated entirely on each upstream project maintainer regaining WHCP signing access and publishing a new signed release — monitor WireGuard GitHub, VeraCrypt GitHub/SourceForge, PassMark MemTest86 portal, and Windscribe release channels at minimum weekly intervals until signed releases are confirmed. Before redeploying any new driver package, validate the full Authenticode chain to 'Microsoft Windows Hardware Compatibility Publisher' using signtool.exe and confirm DSE and Secure Boot remain enforced on all target systems post-deployment. Maintain heightened driver integrity monitoring via Sysmon Event ID 6 for at least 30 days post-recovery to detect any unsigned driver load attempts that may indicate a gap in your redeployment coverage.
Forensic Artifacts	Windows System Event Log — Source: Microsoft-Windows-CodeIntegrity, Event IDs 3001, 3002, 3003, 3004: these events log driver signature validation failures at load time and are the primary indicator that DSE blocked or flagged wireguard.sys, veracrypt.sys, or windscribe kernel driver during boot, directly evidencing the unsigned driver exposure window. | Sysmon Event ID 6 (Driver Loaded) with Hashes field: captures SHA-256 of every kernel-mode driver loaded since Sysmon deployment — filter on ImagePath for wireguard.sys, veracrypt.sys, memtest86 driver, or Windscribe .sys files and compare hashes against official release hashes from project repositories to identify any non-official builds. | driverquery /FO CSV /SI output snapshot: the /SI flag exposes the IsSigned and Manufacturer fields for all loaded drivers — preserve a timestamped CSV from each affected host as the forensic baseline showing signature status at the time of discovery. | Registry key HKLM\SYSTEM\CurrentControlSet\Services\: captures the driver service registration including ImagePath, Start type, and any parameters — a self-signed or tampered driver substituted for the legitimate one would appear here with an anomalous ImagePath or binary not matching the official install directory. | Get-AuthenticodeSignature output on each affected .sys file: the SignerCertificate property and its Issuer field will show whether the driver carries a valid WHCP chain, a self-signed certificate, or no signature at all — this is the definitive artifact distinguishing a legitimately unsigned official build (signing suspended by Microsoft) from a potentially malicious substitution, and must be preserved before any remediation action.

Per-Action IR Details

Containment — Identify all Windows systems in your environment running WireGuard, VeraCrypt, MemTest86, or Windscribe VPN kernel-mode drivers. Confirm which driver versions are currently installed and whether those versions carry valid WHCP signatures. Do not loosen Driver Signature Enforcement (DSE) or disable Secure Boot as a workaround; doing so increases kernel attack surface.

NIST Phase: Containment

Reference: NIST 800-61r3 §3.3 — Containment Strategy

Controls: NIST IR-4 (Incident Handling), NIST SI-7 (Software, Firmware, and Information Integrity), NIST CM-7 (Least Functionality — restrict unsigned kernel-mode drivers), CIS 1.1 (Establish and Maintain Detailed Enterprise Asset Inventory), CIS 4.6 (Securely Manage Enterprise Assets and Software)

Compensating: Run the following PowerShell command on each endpoint to enumerate installed driver signing status for the four affected products: Get-WindowsDriver -Online | Where-Object {$_.ProviderName -match 'WireGuard|VeraCrypt|Windscribe|MemTest'} | Select-Object Driver,ProviderName,Version,Date,Inbox. To verify Authenticode signature chain on the .sys file directly: Get-AuthenticodeSignature 'C:\Windows\System32\drivers\wireguard.sys' (adjust path per product). For fleet-wide coverage without SCCM/Intune, deploy a lightweight osquery query: SELECT name, version, publisher, install_date FROM programs WHERE name LIKE '%WireGuard%' OR name LIKE '%VeraCrypt%' OR name LIKE '%Windscribe%'; Cross-reference output against WHCP signature validity using signtool.exe verify /v /kp from the Windows SDK.

Evidence: Before any changes, capture: (1) output of 'driverquery /FO CSV /SI' on affected hosts — the /SI flag shows signing status and will expose unsigned or self-signed WireGuard (wireguard.sys), VeraCrypt (veracrypt.sys), or Windscribe driver files; (2) Windows System Event Log entries under Source 'Microsoft-Windows-CodeIntegrity' (Event IDs 3001, 3002, 3003, 3004) which log driver signature verification failures at boot — these will confirm whether DSE blocked an unsigned driver load; (3) registry key HKLM\SYSTEM\CurrentControlSet\Services for each affected driver service entry, capturing ImagePath and any Start/Type values that indicate the driver is set to load at boot.

Detection — Query your asset inventory or endpoint management platform (SCCM, Intune, or equivalent) for installed versions of WireGuard, VeraCrypt, MemTest86, and Windscribe VPN across all Windows endpoints. Cross-reference installed driver versions against the last known signed release from each project's official repository. Flag any endpoints where automatic update mechanisms may have stalled.

NIST Phase: Detection Analysis

Reference: NIST 800-61r3 §3.2 — Detection and Analysis

Controls: NIST SI-2 (Flaw Remediation), NIST SI-5 (Security Alerts, Advisories, and Directives), NIST AU-2 (Event Logging), NIST RA-5 (Vulnerability Monitoring and Scanning), CIS 2.1 (Establish and Maintain a Software Inventory), CIS 2.2 (Ensure Authorized Software is Currently Supported), CIS 7.1 (Establish and Maintain a Vulnerability Management Process)

Compensating: Without SCCM or Intune, use osquery with the following query against your fleet: SELECT h.hostname, p.name, p.version, p.install_date FROM programs p JOIN system_info h WHERE p.name LIKE '%WireGuard%' OR p.name LIKE '%VeraCrypt%' OR p.name LIKE '%Windscribe%' OR p.name LIKE '%MemTest86%'; For individual hosts, use: Get-ItemProperty HKLM:\Software\Microsoft\Windows\CurrentVersion\Uninstall* | Where-Object {$_.DisplayName -match 'WireGuard|VeraCrypt|Windscribe|MemTest'} | Select-Object DisplayName,DisplayVersion,InstallDate. Compare returned versions against last WHCP-signed releases: WireGuard 0.5.3 (signed, pre-suspension), VeraCrypt 1.26.7, MemTest86 v10.x (check PassMark portal), Windscribe (check Windscribe GitHub releases). Flag version mismatches or installs dated after the suspension event (reported circa late 2024 / early 2025).

Evidence: Before querying, preserve: (1) Windows Application Event Log entries from Source 'MsiInstaller' and 'Windows Installer' showing install/update history for the four affected packages — Event IDs 1033 (install completed) and 1034 (product removed) with timestamps to establish a timeline of when versions were deployed or auto-updated; (2) Sysmon Event ID 6 (Driver Loaded) logs filtered on ImagePath containing wireguard.sys, veracrypt.sys, or equivalent — Sysmon captures the Hashes field which allows comparison against known-good hash values from official signed releases; (3) scheduled task or service definitions that may trigger auto-update behavior for WireGuard or Windscribe (check HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Schedule\TaskCache\Tasks for entries referencing these products).

Eradication — No malicious component requires removal. The threat vector is supply chain disruption, not compromise. If unsigned or self-signed driver variants are detected in your environment, treat them as unverified and remove or quarantine until a WHCP-signed version is confirmed available from the official project maintainer.

NIST Phase: Eradication

Reference: NIST 800-61r3 §3.4 — Eradication

Controls: NIST IR-4 (Incident Handling), NIST SI-3 (Malicious Code Protection), NIST SI-7 (Software, Firmware, and Information Integrity), NIST CM-11 (User-Installed Software), CIS 2.3 (Address Unauthorized Software), CIS 7.2 (Establish and Maintain a Remediation Process)

Compensating: To disable a suspect unsigned driver service without full uninstall, run: sc config wireguard start= disabled (repeat for veracrypt, windscribe service names as appropriate) then reboot to confirm the driver does not load — verify via 'driverquery /SI' post-reboot. For VeraCrypt, before disabling, confirm no encrypted volumes are mounted that would become inaccessible; document all mounted VeraCrypt volumes first with: Get-WmiObject Win32_Volume | Where-Object {$_.Label -match 'VeraCrypt'}. To quarantine the .sys file without deletion: move wireguard.sys, veracrypt.sys, or windscribe driver .sys to a password-protected zip with evidence tagging before removal. Use signtool.exe verify /pa /v to confirm absence of valid WHCP chain before quarantine decision.

Evidence: Before removal or quarantine, collect: (1) a full copy of the unsigned or self-signed .sys file (wireguard.sys, veracrypt.sys, windscibevpn.sys or equivalent) with SHA-256 hash documented via: Get-FileHash -Algorithm SHA256 \ — this establishes a forensic record of exactly what was running and enables later comparison if the file reappears; (2) the certificate chain embedded in the driver using: (Get-AuthenticodeSignature).SignerCertificate | Format-List — captures whether a self-signed or revoked certificate was used, which would indicate a non-official build entered the environment; (3) Windows Security Event Log Event ID 7045 (A new service was installed) to determine if and when an unsigned driver service was registered, and by which account.

Recovery — Monitor official project repositories (WireGuard GitHub, VeraCrypt SourceForge/GitHub, MemTest86 PassMark, Windscribe official site) for announcements confirming restored signing access and new signed driver releases. Validate signature chains on any new driver packages before deployment. Confirm Secure Boot and DSE remain enforced post-remediation.

NIST Phase: Recovery

Reference: NIST 800-61r3 §3.5 — Recovery

Controls: NIST IR-4 (Incident Handling), NIST SI-2 (Flaw Remediation), NIST SI-7 (Software, Firmware, and Information Integrity), NIST CM-3 (Configuration Change Control), CIS 7.3 (Perform Automated Operating System Patch Management), CIS 7.4 (Perform Automated Application Patch Management)

Compensating: Before deploying any newly released signed driver, validate the full WHCP Authenticode chain using: signtool.exe verify /v /kp /tw 0 — confirm the root certificate resolves to 'Microsoft Code Verification Root' and the intermediate shows 'Microsoft Windows Hardware Compatibility Publisher'. To confirm DSE is still enforced post-deployment, run: bcdedit /enum {current} and verify 'testsigning' is Off and 'nointegritychecks' is Off. For Secure Boot status: Confirm-SecureBootUEFI (returns True if enforced). Set up a lightweight RSS or GitHub release watch on the four repositories using a free tool like Newsfeed Monitor or a cron-driven curl against the GitHub releases API: curl https://api.github.com/repos/WireGuard/wireguard-windows/releases/latest to detect new signed release announcements without manual polling.

Evidence: Before restoring production drivers, capture: (1) the full Authenticode signature details of the new signed installer/driver package — document the signing timestamp, certificate serial number, and WHCP cross-certificate chain as baseline evidence that signing was restored; (2) a post-deployment 'driverquery /FO CSV /SI' snapshot confirming all four drivers now show valid signatures across remediated endpoints — retain this as a recovery validation artifact; (3) Windows System Event Log Source 'Microsoft-Windows-CodeIntegrity' confirming no new Event ID 3001/3002/3004 errors after redeployment, establishing that Secure Boot and DSE accepted the new signed drivers cleanly.

Post-Incident — Document organizational dependencies on open-source tools that rely on Microsoft's Windows Hardware Program for driver signing. Evaluate whether your software supply chain risk assessment accounts for single-vendor administrative control points. Consider adding monitoring for WHCP signing status changes for critical open-source dependencies as an ongoing third-party risk indicator.

NIST Phase: Post Incident

Reference: NIST 800-61r3 §4 — Post-Incident Activity

Controls: NIST IR-4 (Incident Handling), NIST IR-8 (Incident Response Plan), NIST RA-3 (Risk Assessment), NIST SA-12 (Supply Chain Protection), NIST SI-5 (Security Alerts, Advisories, and Directives), CIS 7.1 (Establish and Maintain a Vulnerability Management Process), CIS 2.1 (Establish and Maintain a Software Inventory)

Compensating: Create a standing inventory entry for each WHCP-dependent open-source driver in your software inventory (CIS 2.1 requirement) tagging it with 'driver-signing-dependency: WHCP' and the last verified signed release version and date. Establish a biweekly manual review task: query each project's GitHub releases API (curl https://api.github.com/repos/WireGuard/wireguard-windows/releases/latest) and compare the published signing certificate serial against your documented baseline. For the lessons-learned document, record the operational gap exposed: WireGuard handles kernel-mode VPN tunneling, VeraCrypt handles full-disk encryption, MemTest86 handles memory diagnostics — all three affect security-critical functions, and loss of signed driver delivery for any of them during patch cycles creates a measurable compliance and continuity risk.

Evidence: For the post-incident record, retain: (1) the full driverquery /SI output snapshots taken at the start of the incident and after recovery as before/after baseline documentation; (2) a timeline log cross-referencing Microsoft's WHCP suspension notice dates (from official Microsoft Partner Center communications or public disclosure), the dates each affected project publicly announced the suspension, and when your organization first detected the operational impact — this gap analysis is the primary artifact for supply chain risk process improvement; (3) the signed driver certificate serial numbers and validity periods documented for each affected product pre-incident, establishing what 'known-good' looks like for future deviation detection.

Detection Guidance

No adversarial IOCs are associated with this incident. Detection focus is operational. Query endpoint management platforms (Intune, SCCM, or equivalent) for installed driver versions of WireGuard (wireguard.sys), VeraCrypt (veracrypt.sys), MemTest86 drivers, and Windscribe VPN drivers. Use Windows Event Log, System channel, Event ID 7045 (new service installed) and Event ID 219 (driver load failure), to identify any unsigned or improperly signed driver load attempts. In environments with Windows Defender Application Control (WDAC) or AppLocker driver rules, review blocked driver events in the CodeIntegrity operational log (Microsoft-Windows-CodeIntegrity/Operational, Event ID 3077 or 3089) for failures tied to these driver binaries. Flag any internal deployment pipelines that pull driver packages from these projects and alert if expected update cadences stall beyond your established baseline.

Framework Mappings

MITRE-ATTACK

- **T1553.006** — Code Signing Policy Modification
- **T1195.002** — Compromise Software Supply Chain

NIST-800-53R5

- **CM-7** — Least Functionality
- **SA-9** — External System Services
- **SR-3** — Supply Chain Controls and Processes
- **SI-7** — Software, Firmware, and Information Integrity
- **SR-2** — Supply Chain Risk Management Plan

CIS-V8

- **7.3** — Perform Automated Operating System Patch Management
- **7.4** — Perform Automated Application Patch Management
- **15.1** — Establish and Maintain an Inventory of Service Providers

ISO-27001-2022

- **A.8.8** — Management of technical vulnerabilities
- **A.5.21** — Managing information security in the ICT supply chain

NIST-CSF-2

- **GV.SC-01** — Cybersecurity supply chain risk management program

SOC2-TSC

- **CC9.2** — Manages risks associated with vendors and business partners

MITRE ATT&CK Mapping

Technique ID	Technique Name	Tactic
T1553.006	Code Signing Policy Modification	Defense-Evasion
T1195.002	Compromise Software Supply Chain	Initial-Access

Sources

Source	URL	Tier
Security News	https://www.bleepingcomputer.com/news/microsoft/microsoft-suspends-...	T3
Microsoft suspends dev accounts for high-profile open source projects	https://www.bleepingcomputer.com/news/microsoft/microsoft-suspends-...	T3
Microsoft BANNED WireGuard, VeraCrypt & Windscribe With Zero ...	https://www.youtube.com/watch?v=fTui3CQuL9I	T3
VeraCrypt, WireGuard maintainers locked out of their Microsoft ...	https://cybernews.com/security/microsoft-suspends-veracrypt-wiregua...	T3
Microsoft Mysteriously Freezes Accounts for VeraCrypt, WireGuard ...	https://me.pcmag.com/en/vpn/36463/microsoft-mysteriously-freezes-ac...	T3

Generated 2026-04-09 13:33 UTC by TJS Security Command Center