INTELLIGENCE BRIEFING
Security Command Center
TLP:CLEAR
2026-05-02 13:42 UTC

Wireshark Security Update Addresses 40+ Vulnerabilities Including Arbitrary Code Execution Flaws

CVE VULNERABILITY | HIGH | CVSS 7.8

SCC Item ID	SCC-CVE-2026-0112
Type	CVE Vulnerability
Severity	HIGH
CVSS Base Score	7.8
Affected Products	Wireshark (multiple versions prior to latest security release; specific version strings not confirmed from available sources)
Published	52 minutes ago
Discovery Source	Serper

Executive Summary

Wireshark, the open-source network protocol analyzer used extensively by security teams and network engineers worldwide, has released a security update addressing multiple vulnerabilities, including flaws that allow arbitrary code execution and denial-of-service via malformed packets. Organizations where analysts run Wireshark against untrusted or live network captures face direct exposure: a crafted packet could compromise the analyst's workstation. Patching to the latest release is the immediate priority; specific CVE identifiers and affected version ranges require confirmation from the official Wireshark advisory at https://www.wireshark.org/security/.

Technical Analysis

Wireshark's protocol dissectors, which parse and decode network traffic, are the affected components. Vulnerabilities include heap-based buffer overflow (CWE-122), out-of-bounds write (CWE-787), uncontrolled resource consumption (CWE-400), and out-of-bounds read (CWE-125). Exploitation paths include injecting malformed packets into a live capture or supplying a crafted capture file, triggering arbitrary code execution (MITRE T1203) or denial-of-service (MITRE T1499) on the analyst's system. CVSS base score is reported at 7.8 (High), consistent with local/adjacent execution context. Specific CVE identifiers and precise affected version ranges require direct review of the Wireshark Security Advisories page (https://www.wireshark.org/security/). Source quality for this item is secondary-tier (cybersecuritynews.com, gbhackers.com); human validation against the official advisory is strongly recommended before finalizing

remediation scope.

Action Checklist

1. Step 1: Containment, Identify all workstations and systems running Wireshark, particularly those used for live capture against untrusted networks or for opening externally sourced capture files. Restrict use of Wireshark on those systems until patched. Do not open .pcap or .pcapng files from untrusted sources in the interim.
2. Step 2: Detection, Query your asset inventory and endpoint management tools (e.g., SCCM, Jamf, osquery) for installed Wireshark versions. Visit https://www.wireshark.org/security/ to identify the latest patched release version and compare against your installed versions. There are no network-based IOCs for this vulnerability class; exposure is determined by software version and usage pattern, not inbound traffic signatures.
3. Step 3: Eradication, Upgrade all Wireshark installations to the patched release confirmed at https://www.wireshark.org/security/. Follow the vendor-provided upgrade path for your operating system (Windows installer, macOS package, Linux package manager). Confirm the installed version post-upgrade.
4. Step 4: Recovery, After upgrade, verify the installed version matches the patched release. Re-enable Wireshark use for analysts. Where analysts routinely open third-party capture files, implement a policy requiring those files be opened in isolated analysis environments (sandboxed VM) as a standing control.
5. Step 5: Post-Incident, Review whether Wireshark and other analyst tools are included in your patch management scope and tracked in your software asset inventory. Analyst workstations running protocol dissectors against untrusted input are a persistent attack surface; add them to your vulnerability management cycle with defined SLAs.

IR / Forensic Enrichment

Triage Priority	URGENT
Escalation Criteria	Escalate to CISO and initiate deeper forensic investigation if any analyst workstation shows evidence of Wireshark spawning unexpected child processes (Sysmon Event ID 1 with wireshark.exe as ParentImage), unexplained crash dumps in WER for Wireshark.exe coinciding with opening an externally sourced .pcap file, or unexpected DLL loads (Sysmon Event ID 7) into the Wireshark process — any of these indicators suggest attempted or successful arbitrary code execution via a malformed packet dissector exploit, which may trigger organizational breach notification obligations if the compromised workstation had access to sensitive network captures containing PII, PHI, or credentials.
Recovery Notes	After all Wireshark installations are confirmed upgraded to the patched release per wireshark.org/security/, monitor analyst workstations for 30 days using Sysmon Event ID 1 (Process Creation) filtered on wireshark.exe as ParentImage — any child process spawned by Wireshark post-patching is anomalous and warrants immediate investigation. Retain all externally sourced .pcap and .pcapng files that were opened during the vulnerability window in quarantined storage (SHA-256 hashed and access-controlled) for potential retrospective forensic analysis if a compromise indicator surfaces later. Verify that the sandboxed VM analysis procedure for untrusted capture files is documented in the analyst runbook and tested by each analyst before the incident is formally closed.

Forensic Artifacts	Windows Error Reporting crash archives at C:\ProgramData\Microsoft\Windows\WER\ReportArchive\ and C:\Users\%USERNAME%\AppData\Local\CrashDumps\ — a Wireshark dissector crash triggered by a malformed packet crafted to exploit one of the 40+ patched flaws would appear here as a fault in a specific dissector module; analyze minidump files with WinDbg to identify the faulting instruction pointer and whether it was redirected to attacker-controlled data | Wireshark MRU registry key at HKCU:\Software\Wireshark\recent_files — records the file paths of all recently opened .pcap and .pcapng files; cross-reference these paths against known-untrusted sources (external shares, email attachments, downloaded files) to identify which files were potential malicious inputs during the vulnerability window | Sysmon Event ID 1 (Process Creation) logs filtered on ParentImage containing wireshark.exe — legitimate Wireshark use produces no child processes; any subprocess (cmd.exe, powershell.exe, sh, python, etc.) spawned by wireshark.exe is a high-confidence indicator of successful arbitrary code execution via a dissector exploit | Wireshark plugin directories for unexpected or unsigned files: Windows at %APPDATA%\Wireshark\plugins\ and the Wireshark install dir plugins subdirectory, Linux at ~/.local/lib/wireshark/plugins/ and /usr/lib/wireshark/plugins/ — a post-exploitation persistence mechanism could involve dropping a malicious .lua script or .so dissector plugin that executes on next Wireshark launch | Windows Prefetch files at C:\Windows\Prefetch\WIRESHARK.EXE-*.pf — provides execution timestamps and the list of DLLs and files accessed during each Wireshark run; parse with Eric Zimmerman's PECmd to reconstruct which .pcap files were loaded in which sessions and correlate with crash events or anomalous child process creation in the same timeframe

Per-Action IR Details

Step 1: Containment — Identify all workstations and systems running Wireshark, particularly those used for live capture against untrusted networks or for opening externally sourced capture files. Restrict use of Wireshark on those systems until patched. Do not open .pcap or .pcapng files from untrusted sources in the interim.

NIST Phase: Containment

Reference: NIST 800-61r3 §3.3 — Containment Strategy: isolate affected systems and restrict the attack vector (malformed packet ingestion via live capture or untrusted .pcap/.pcapng file parsing) before eradication is complete.

Controls: NIST IR-4 (Incident Handling), NIST CM-7 (Least Functionality) — restrict Wireshark execution to patched instances only, CIS 2.3 (Address Unauthorized Software) — treat unpatched Wireshark as unauthorized pending upgrade, CIS 4.4 (Implement and Manage a Firewall on Servers) — where Wireshark runs on servers, apply host firewall rules to block raw socket access until patched

Compensating: For teams without MDM/SCCM: run the following PowerShell one-liner across Windows endpoints via PsExec or a scheduled task to enumerate installations: `Get-ItemProperty HKLM:\Software\Microsoft\Windows\CurrentVersion\Uninstall\* | Where-Object {$_.DisplayName -like '*Wireshark*'} | Select-Object DisplayName, DisplayVersion, PSComputerName`. On Linux, use: `dpkg -l wireshark* 2>/dev/null || rpm -qa | grep wireshark`. Immediately rename or chmod-restrict the Wireshark binary (`chmod 000 /usr/bin/wireshark`) on unpatched Linux systems until the package is upgraded. On Windows, use AppLocker (even in audit mode) or Software Restriction Policies to block wireshark.exe and Wireshark-gtk.exe by hash pending patching.

Evidence: Before restricting Wireshark use, document which analysts opened externally sourced .pcap or .pcapng files within the past 30 days by reviewing: (1) Windows prefetch files at C:\Windows\Prefetch\WIRESHARK.EXE-*.pf for execution timestamps; (2) Windows Security Event Log Event ID 4688 (Process Creation) filtering on wireshark.exe with command-line arguments showing file paths — log source: Microsoft-Windows-Security-Auditing; (3) Recent file lists in the Wireshark MRU registry key at HKCU:\Software\Wireshark\recent_files to identify which .pcap/.pcapng files were opened and from which paths; (4) On macOS, check ~/Library/Application Support/Wireshark/ and Console logs for crash reports (*.crash in ~/Library/Logs/DiagnosticReports/) that could indicate a dissector triggered by a malformed packet.

Step 2: Detection — Query your asset inventory and endpoint management tools (e.g., SCCM, Jamf, osquery) for installed Wireshark versions. Confirm version strings against the patched release listed at wireshark.org/security/. There are no network-based IOCs for this vulnerability class; exposure is determined by software version and usage pattern, not inbound traffic signatures.

NIST Phase: Detection Analysis

Reference: NIST 800-61r3 §3.2 — Detection and Analysis: because this vulnerability class (malformed packet dissector flaws) produces no network-observable IOC, detection must shift to software inventory enumeration and version-string comparison rather than signature-based alerting.

Controls: NIST SI-2 (Flaw Remediation) — identify vulnerable Wireshark versions across the environment, NIST SI-5 (Security Alerts, Advisories, and Directives) — consume the official Wireshark security advisory to define the vulnerable version range, NIST AU-2 (Event Logging) — confirm process creation logging is enabled on analyst workstations to support retrospective analysis, CIS 2.1 (Establish and Maintain a Software Inventory) — Wireshark must appear in the software inventory with version tracking, CIS 7.1 (Establish and Maintain a Vulnerability Management Process) — analyst tool installations must be within vulnerability scan scope

Compensating: Without SCCM or Jamf: deploy the following osquery query to enumerate all endpoints: `SELECT name, version, install_location FROM programs WHERE name LIKE '%Wireshark%';` (Windows) or `SELECT name, version FROM deb_packages WHERE name LIKE '%wireshark%';` (Linux). For a 2-person team with no osquery, run a network-wide PowerShell inventory via: `Invoke-Command -ComputerName (Get-ADComputer -Filter *).Name -ScriptBlock { Get-ItemProperty HKLM:\Software\Microsoft\Windows\CurrentVersion\Uninstall\* | Where {$_.DisplayName -like '*Wireshark*'} | Select DisplayName,DisplayVersion } 2>$null`. Compare resulting version strings against the patched release per wireshark.org/security/ — any version below the patched threshold is in scope regardless of whether the system is used for live capture, since file-open exploitation applies to all installations.

Evidence: Since exploitation via malformed packet dissection leaves no network signature, forensic evidence of potential prior exploitation must be sourced from the endpoint: (1) Windows Application Event Log and Windows Error Reporting (WER) at C:\ProgramData\Microsoft\Windows\WER\ReportArchive\ for Wireshark crash reports — a dissector crash from a malformed packet may appear here as a fault in a specific dissector DLL (e.g., packet-foo.dll); (2) Process crash dumps at C:\Users\%USERNAME%\AppData\Local\CrashDumps\ or configured WER path — analyze with WinDbg for signs of heap corruption or code execution redirection in Wireshark dissector threads; (3) Sysmon Event ID 1 (Process Creation) and Event ID 11 (FileCreate) logs to identify if Wireshark spawned any child processes or wrote unexpected files to temp directories following the opening of a specific .pcap file — unexpected child processes from wireshark.exe are a strong indicator of ACE exploitation.

Step 3: Eradication — Upgrade all Wireshark installations to the latest release confirmed as patched per the official Wireshark security advisory (https://www.wireshark.org/security/). Follow the vendor-provided upgrade path for your operating system (Windows installer, macOS package, Linux package manager). Confirm the installed version post-upgrade.

NIST Phase: Eradication

Reference: NIST 800-61r3 §3.4 — Eradication: remove the vulnerable Wireshark version from all endpoints and replace with the patched release per the official Wireshark security advisory; verify eradication by re-querying software inventory post-upgrade.

Controls: NIST SI-2 (Flaw Remediation) — apply vendor-issued patch from official Wireshark security advisory; test on a representative analyst workstation before broad deployment, NIST CM-3 (Configuration Change Control) — document the version transition from vulnerable to patched Wireshark as a configuration change, CIS 7.3 (Perform Automated Operating System Patch Management) — apply to OS-integrated Wireshark packages (e.g., apt/yum managed installs), CIS 7.4 (Perform Automated Application Patch Management) — apply to standalone Wireshark installer packages on Windows and macOS, CIS 2.2 (Ensure Authorized Software is Currently Supported) — post-upgrade, confirm the patched version is the designated authorized version in the software inventory

Compensating: For teams without automated patch deployment: on Windows, script the silent upgrade using the Wireshark NSIS installer with: `wireshark--x64.exe /S` (silent install flag) deployed via PsExec or a simple batch file distributed via a shared drive. On Debian/Ubuntu Linux: `sudo apt-get update && sudo apt-get install --only-upgrade

wireshark`. On RHEL/CentOS: `sudo yum update wireshark`. On macOS with Homebrew: `brew upgrade wireshark`. After upgrade on each system, confirm the version with: `wireshark --version | head -1` (Linux/macOS) or `Get-ItemProperty HKLM:\Software\Microsoft\Windows\CurrentVersion\Uninstall\Wireshark | Select DisplayVersion` (Windows). Log confirmed patched versions and analyst workstation hostnames in a remediation tracker (a spreadsheet is acceptable for a 2-person team).

Evidence: Before executing the upgrade, preserve the following artifacts from each endpoint in case a prior compromise needs to be investigated: (1) A full copy of the Wireshark MRU registry hive from HKCU:\Software\Wireshark\recent_files documenting all recently opened capture files — export via `reg export HKCU\Software\Wireshark wireshark_mru_backup.reg`; (2) Any .pcap or .pcapng files staged in analyst Downloads, Desktop, or temp directories that were sourced externally — hash these files (SHA-256) and retain copies before the endpoint is upgraded, as they are potential malicious inputs; (3) Wireshark dissector plugin directories (Windows: %APPDATA%\Wireshark\plugins\, Linux: ~/.local/lib/wireshark/plugins/) for any unexpected or unsigned .lua or .so plugin files that could indicate post-exploitation persistence via a malicious Wireshark plugin.

Step 4: Recovery — After upgrade, verify the installed version matches the patched release. Re-enable Wireshark use for analysts. Where analysts routinely open third-party capture files, implement a policy requiring those files be opened in isolated analysis environments (sandboxed VM) as a standing control.

NIST Phase: Recovery

Reference: NIST 800-61r3 §3.5 — Recovery: restore analyst capability by re-enabling patched Wireshark, and implement standing isolation controls for untrusted capture file analysis to prevent recurrence of file-based dissector exploitation.

Controls: NIST IR-4 (Incident Handling) — document recovery actions and verify analyst workstations are returned to known-good state, NIST SC-7 (Boundary Protection) — enforce network isolation for analysis VMs that open untrusted capture files, NIST CM-6 (Configuration Settings) — document the sandboxed analysis VM as a required configuration for analyst workstations processing externally sourced .pcap/.pcapng files, CIS 4.6 (Securely Manage Enterprise Assets and Software) — manage analysis VM configurations under version control or snapshot discipline so a clean baseline can be restored after each analysis session

Compensating: For teams without commercial sandbox solutions: provision a VirtualBox or VMware Workstation Player VM (both free tiers available) on each analyst workstation, install Wireshark inside the VM, and configure the VM network adapter to Host-Only mode to prevent any exploit from reaching the production network. After each untrusted .pcap analysis session, revert the VM to a clean snapshot. Document this procedure in the analyst runbook. Use REMnux (free, purpose-built Linux distro for malware and packet analysis) as the VM OS — it includes Wireshark and network isolation tooling pre-configured. For version verification post-upgrade on all platforms: create a simple cron job or scheduled task that runs `wireshark --version > /var/log/wireshark_version_check.log` daily and alerts if the version string reverts.

Evidence: Before re-enabling analyst access to Wireshark on any workstation, verify: (1) Windows Security Event Log Event ID 4688 showing no unexpected child process spawned by wireshark.exe since the period when the vulnerable version was in use — query with PowerShell: `Get-WinEvent -LogName Security | Where {$_.Id -eq 4688 -and $_.Message -like '*wireshark*'}` and review ParentProcessName and CommandLine fields; (2) File system integrity check on the Wireshark installation directory (Windows: C:\Program Files\Wireshark\) using `Get-FileHash` against a known-good hash baseline from the vendor installer to confirm no DLL has been tampered with post-exploit; (3) Sysmon Event ID 7 (Image Load) logs to verify no unsigned or unexpected DLLs were loaded into the wireshark.exe process space during the vulnerability window.

Step 5: Post-Incident — Review whether Wireshark and other analyst tools are included in your patch management scope and tracked in your software asset inventory. Analyst workstations running protocol dissectors against untrusted input are a persistent attack surface; add them to your vulnerability management cycle with defined SLAs.

NIST Phase: Post Incident

Reference: NIST 800-61r3 §4 — Post-Incident Activity: document lessons learned specific to the gap that allowed untracked Wireshark installations to exist outside patch management scope, and update vulnerability management

processes to include protocol analyzer tools as a defined asset class.

Controls: NIST IR-4 (Incident Handling) — conduct post-incident review and update IR procedures to include analyst tool patching as an explicit scope item, NIST IR-8 (Incident Response Plan) — update the IR plan to include protocol dissector tools (Wireshark, tcpdump, NetworkMiner) as a tracked asset class requiring defined patch SLAs, NIST SI-2 (Flaw Remediation) — establish patch SLA for analyst tools: given CVSS 7.8 (HIGH) with arbitrary code execution, target ≤15 days for patching on internet-connected or network-capture workstations, NIST RA-3 (Risk Assessment) — formally document analyst workstations processing untrusted packet captures as a risk scenario, including the dissector attack surface, in the organizational risk register, CIS 7.1 (Establish and Maintain a Vulnerability Management Process) — add Wireshark and equivalent protocol analyzer tools to the vulnerability scan scope with version tracking, CIS 7.2 (Establish and Maintain a Remediation Process) — define remediation SLA tiers: HIGH severity (CVSS ≥7.0) analyst tools = 15-day patch window, CIS 2.1 (Establish and Maintain a Software Inventory) — ensure Wireshark appears in the software inventory with version, owner, and patching-authority fields populated

Compensating: For teams without a vulnerability scanner: create a monthly osquery scheduled query (deployable via osquery's `osquery.conf` `schedule` block) that reports all installed Wireshark versions across the fleet and outputs to a local log file ingested by any syslog forwarder. Subscribe to the Wireshark security advisory mailing list (announcements@wireshark.org) and the Wireshark-announce RSS feed as a zero-cost early-warning channel. Document in the patch management runbook that any Wireshark advisory rated HIGH or CRITICAL (CVSS ≥7.0) triggers a 15-day mandatory upgrade cycle with named analyst workstation owners accountable for confirmation.

Evidence: For the post-incident lessons-learned record, compile: (1) The complete list of Wireshark versions found in the environment at discovery time versus patched release, with per-system hostnames and owner names — this documents the exposure window and blast radius; (2) A timeline reconstructed from Windows prefetch (C:\Windows\Prefetch\WIRESHARK.EXE-*.pf) and Sysmon Event ID 1 logs showing when Wireshark was last executed on each workstation and whether any externally sourced .pcap files were opened during the vulnerable period — this determines whether any workstations require deeper forensic investigation; (3) The delta between the Wireshark advisory date and the date your team was first aware of it — this gap measurement drives the process improvement recommendation for advisory monitoring.

Detection Guidance

Detection here is asset-based, not behavioral. Query endpoint management or EDR telemetry for Wireshark installations: on Windows, check registry keys under HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall for Wireshark entries and compare version strings; on Linux, run 'wireshark --version' or query the package manager (dpkg, rpm); on macOS, check /Applications/Wireshark.app/Contents/Info.plist for CFBundleShortVersionString. Compare installed versions against the patched release identified at https://www.wireshark.org/security/. Any version prior to the confirmed patched release is affected. No CVE-specific IOC patterns (hashes, signatures, behavioral indicators) are available at this time from the sources reviewed. If your IDS/IPS has a Wireshark dissector vulnerability signature set, verify it is updated, but signature coverage for this class of vulnerability is not confirmed.

Framework Mappings

MITRE-ATTACK

- **T1499** — Endpoint Denial of Service
- **T1203** — Exploitation for Client Execution

NIST-800-53R5

- **SC-5** — Denial-of-Service Protection

- **SC-7** — Boundary Protection
- **SI-2** — Flaw Remediation
- **SI-3** — Malicious Code Protection
- **SI-4** — System Monitoring
- **SI-16** — Memory Protection

CIS-V8

- **16.10** — Apply Secure Design Principles in Application Architectures
- **13.8** — Deploy a Network Intrusion Prevention Solution
- **7.3** — Perform Automated Operating System Patch Management
- **7.4** — Perform Automated Application Patch Management

ISO-27001-2022

- **A.5.23** — Information security for use of cloud services

MITRE ATT&CK Mapping

Technique ID	Technique Name	Tactic
T1499	Endpoint Denial of Service	Impact
T1203	Exploitation for Client Execution	Execution

Sources

Source	URL	Tier
	https://cybersecuritynews.com/wireshark-vulnerabilities-code-execut...	T3
Critical Wireshark Flaws Let Attackers Execute Arbitrary Code Via ...	https://x.com/The_Cyber_News/status/2050112484879843421	T3
Critical Wireshark Vulnerabilities Let Attackers Execute Arbitrary ...	https://www.linkedin.com/posts/dlross_critical-wireshark-vulnerabil...	T3
Multiple Wireshark Vulnerabilities Allow Arbitrary Code Execution ...	https://gbhackers.com/multiple-wireshark-vulnerabilities/amp/	T3
Wireshark Vulnerabilities Let Attackers Crash by Injecting ... - Siembiot	https://siembiot.eu/cyber-security-news/wireshark-vulnerabilities-l...	T3

DISCLAIMER

Generated 2026-05-02 13:42 UTC by TJS Security Command Center