INTELLIGENCE BRIEFING
Security Command Center
TLP:CLEAR
2026-07-01 07:10 UTC

Google Chrome 151 Patches Large Volume of Security Vulnerabilities Including Critical UAF Flaws

SECURITY ANALYSIS | CRITICAL | CVSS 8.8

SCC Item ID	SCC-STY-2026-0312
Type	Security Analysis
Severity	CRITICAL
CVSS Base Score	8.8
Affected Products	Google Chrome prior to 151 (stable channel, Windows, Mac, Linux)
Published	3 hours ago
Discovery Source	Serper

Executive Summary

Google has released Chrome 151 to address a significant number of security vulnerabilities, including multiple critical Use-After-Free (UAF) flaws that could allow attackers to execute arbitrary code on affected systems. The exact vulnerability count is disputed across sources: a single aggregated news outlet (Cyberpress) reports 382 flaws, while a Forbes report cites 151 flaws with 22 critical, and neither figure is corroborated by the Google Chrome Releases blog at this time. Regardless of the precise count, the presence of critical UAF vulnerabilities in the world's most widely deployed browser warrants prompt update deployment across enterprise endpoints.

Technical Analysis

Chrome's Use-After-Free vulnerability class (CWE-416) is among the most operationally dangerous browser flaw categories. UAF flaws arise when a program retains a pointer to memory that has already been freed; if an attacker can control the contents of that reclaimed memory region, they can redirect execution flow. In browser contexts, this typically maps to MITRE ATT&CK T1203 (Exploitation for Client Execution) and T1189 (Drive-by Compromise), meaning exploitation requires only that a target visit or be redirected to an attacker-controlled web page, no additional user interaction is required beyond browsing.

The source picture here is notably inconsistent and warrants transparency. Malwarebytes (T1 source) confirms that critical browser security flaws have been patched and recommends updating Chrome, but does not independently corroborate the 382-vulnerability figure. The Forbes report (T3) references 151 flaws with 22 critical, which may reflect an earlier wave or a different counting methodology. The 382 figure originates solely

from Cyberpress (T3), an aggregated news outlet with no direct access to Google's security advisory data. Specific CVE identifiers, affected sub-components, CVSS vectors, and EPSS scores are absent from all provided source material. Security teams should treat the vulnerability count as unresolved and focus on the confirmed operational fact: Chrome has been updated with critical-severity fixes, and unpatched endpoints running prior versions remain exposed to drive-by exploitation chains.

The attack surface is substantial. Chrome holds a dominant share of enterprise and consumer browser deployments across Windows, Mac, and Linux. Drive-by exploitation via UAF has historically been a preferred initial access vector for both financially motivated threat actors and nation-state operators, frequently chained with sandbox escapes or privilege escalation to achieve full system compromise. The absence of CISA KEV listing at this time does not reduce urgency; UAF browser vulnerabilities have moved to active exploitation within days of public disclosure in previous cycles.

Action Checklist

1. Step 1: Assess exposure, confirm whether your organization deploys Google Chrome on Windows, Mac, or Linux endpoints; Chrome prior to version 151 on the stable channel is the affected population
2. Step 2: Deploy the update, push Chrome 151 to all managed endpoints via your endpoint management tooling (SCCM, Intune, Jamf, or equivalent); verify successful update through inventory reporting aligned with CIS 2.2 (Ensure Authorized Software is Currently Supported) and CIS 7.3 (Perform Automated Operating System Patch Management)
3. Step 3: Verify patch coverage, audit software inventory for unmanaged or BYOD endpoints running Chrome; address gaps per CIS 1.1 (Establish and Maintain Detailed Enterprise Asset Inventory) and CIS 2.1 (Establish and Maintain a Software Inventory)
4. Step 4: Review web browsing controls, confirm that proxy-based or endpoint-based web access controls are in place to limit exposure to drive-by compromise via untrusted or unknown domains; review firewall rules per CIS 4.4 and CIS 4.5
5. Step 5: Monitor for exploitation indicators, hunt for anomalous Chrome renderer process behavior, unexpected child process spawning from browser processes, and memory corruption crash telemetry; correlate with AU-6 (Audit Record Review, Analysis, and Reporting) requirements
6. Step 6: Track the advisory, monitor the Google Chrome Releases blog and NVD for official CVE assignments, CVSS scores, and any CISA KEV additions tied to this release cycle, as source-level inconsistencies in this story mean the full scope is not yet confirmed

IR / Forensic Enrichment

Triage Priority	URGENT
Escalation Criteria	Escalate to incident response leadership and legal/compliance if Sysmon or crash telemetry confirms Chrome renderer sandbox escape (child process spawning from chrome.exe) on any endpoint, if CISA adds any Chrome 151-cycle CVE to the Known Exploited Vulnerabilities catalog, or if patch coverage audit reveals more than 10% of endpoints remain on pre-151 Chrome after 72 hours — particularly on endpoints with access to PII, PHI, or financial data triggering breach notification obligations under HIPAA, GDPR, or state privacy statutes.

Recovery Notes	After confirming 100% patch coverage to Chrome 151 stable via software inventory audit, maintain elevated monitoring of Chrome renderer process telemetry (Sysmon Event IDs 1, 8, 10) for a minimum of 14 days post-patch, as UAF exploitation may have established persistence via post-exploitation payloads (scheduled tasks, registry run keys, or browser extension abuse) that survive the browser update. Verify Chrome extension inventory against the pre-incident baseline — attackers exploiting UAF flaws in the renderer may have silently installed malicious extensions that persist independently of the browser version. If any endpoint shows confirmed exploitation indicators, treat it as fully compromised, do not attempt in-place remediation, and reimage from a known-good baseline before returning to production.
Forensic Artifacts	Chrome Breakpad crash minidumps in %LOCALAPPDATA%\Google\CrashReports\ (Windows) or ~/.config/google-chrome/Crash Reports/ (Linux/Mac) — UAF exploitation causes renderer crashes recorded as 0xC0000005 Access Violation faults with heap corruption context; parse with WinDbg or minidump_stackwalk to extract fault address and module offset | Sysmon Event ID 1 (Process Create) logs showing chrome.exe or chrome renderer processes (identifiable by --type=renderer command-line flag) as parent of non-Chrome binaries — the primary forensic indicator of sandbox escape following UAF exploitation | Windows Error Reporting entries (Application Event Log, Event ID 1000, Source: Application Error) for chrome.exe with faulting module chrome.dll — capture the faulting module version and offset to correlate with specific UAF vulnerability location in the Chrome 151 release diff | Chrome renderer process memory image captured via WinPmem or LiME prior to any containment action — UAF heap spray artifacts (NOP sleds, ROP chains, shellcode patterns) may be recoverable from renderer process heap space and are destroyed on process termination or system restart | Web proxy or DNS resolver logs (Sysmon Event ID 22, DNSEvent, filtered on chrome.exe Image) for the 48-hour window preceding detection — UAF drive-by exploitation requires the victim browser to load attacker-controlled content, making the referring domain and full request URI the primary delivery chain evidence; look for requests to domains serving large JavaScript payloads with ArrayBuffer or WebAssembly content, common UAF heap-grooming delivery mechanisms

Per-Action IR Details

Step 1: Assess exposure — confirm whether your organization deploys Google Chrome on Windows, Mac, or Linux endpoints; Chrome prior to version 151 on the stable channel is the affected population

NIST Phase: Detection Analysis

Reference: NIST 800-61r3 §3.2 — Detection and Analysis: scope assessment to identify affected assets before action

Controls: CIS 1.1 (Establish and Maintain Detailed Enterprise Asset Inventory), CIS 2.1 (Establish and Maintain a Software Inventory)

Compensating: Run `wmic product where 'name like "%Chrome%"' get name,version` on Windows endpoints via PSExec or Group Policy startup script; on Linux use `dpkg -l google-chrome-stable` or `rpm -qa google-chrome-stable`; on macOS use `defaults read /Applications/Google\ Chrome.app/Contents/Info CFBundleShortVersionString`. Aggregate results into a CSV for triage. osquery query: `SELECT name, version FROM programs WHERE name LIKE '%Chrome%';`

Evidence: This is an assessment step that does not alter live state. No volatile capture is required before execution. Document the Chrome version string and host list as a baseline artifact for post-patch comparison.

Step 2: Deploy the update — push Chrome 151 to all managed endpoints via your endpoint management tooling (SCCM, Intune, Jamf, or equivalent); verify successful update through inventory reporting aligned with CIS 2.2 (Ensure Authorized Software is Currently Supported) and CIS 7.3 (Perform Automated Operating System Patch Management)

NIST Phase: Eradication

Reference: NIST 800-61r3 §3.4 — Eradication: removing the vulnerability from the environment by applying the vendor-supplied update

Controls: CIS 2.2 (Ensure Authorized Software is Currently Supported), CIS 7.3 (Perform Automated Operating System Patch Management), CIS 7.4 (Perform Automated Application Patch Management), NIST SI-2 (Flaw Remediation)

Compensating: For unmanaged endpoints, push Chrome's built-in updater via registry policy: set `HKLM\SOFTWARE\Policies\Google\Update\AutoUpdateCheckPeriodMinutes` to 60 and `UpdateDefault` to 1. Confirm version post-update with `(Get-Item 'C:\Program Files\Google\Chrome\Application\chrome.exe').VersionInfo.ProductVersion` on PowerShell. For Linux, `apt-get install --only-upgrade google-chrome-stable` or `yum update google-chrome-stable`.

Evidence: Before deploying the patch to any host suspected of active exploitation, capture: (1) a full RAM image using WinPmem or LiME to preserve any in-memory UAF heap spray artifacts or shellcode, (2) `Get-Process chrome* | Select-Object Id,Name,Path,StartTime,MainWindowTitle` output to document all Chrome renderer PID chains, and (3) Chrome's crash report directory (`%LOCALAPPDATA%\Google\CrashReports\` on Windows) which may contain Breakpad minidumps recording the UAF fault address and heap state at time of crash. These are destroyed on browser restart or patch application.

Step 3: Verify patch coverage — audit software inventory for unmanaged or BYOD endpoints running Chrome; address gaps per CIS 1.1 (Establish and Maintain Detailed Enterprise Asset Inventory) and CIS 2.1 (Establish and Maintain a Software Inventory)

NIST Phase: Recovery

Reference: NIST 800-61r3 §3.5 — Recovery: confirming eradication is complete and no vulnerable instances remain before returning systems to full operational status

Controls: CIS 1.1 (Establish and Maintain Detailed Enterprise Asset Inventory), CIS 2.1 (Establish and Maintain a Software Inventory), CIS 2.3 (Address Unauthorized Software)

Compensating: Deploy an osquery scheduled query (`SELECT * FROM programs WHERE name LIKE '%Chrome%' AND version < '151.0.0.0';`) across all enrolled endpoints and export results. For BYOD or network-adjacent devices not enrolled in MDM, use network-based detection: configure your web proxy or DNS resolver to log User-Agent strings and flag requests containing `Chrome/1[0-4][0-9]` (pre-151 version strings) — this surfaces BYOD devices browsing through corporate infrastructure on vulnerable Chrome builds.

Evidence: No live state is altered by an audit step. Preserve the pre-audit software inventory snapshot as a versioned artifact to demonstrate coverage delta before and after patching, supporting post-incident review under NIST 800-61r3 §4.

Step 4: Review web browsing controls — confirm that web proxy or endpoint-based web access mediation (D3-PBWSAM, D3-EBWSAM) is in place to limit exposure to drive-by compromise via untrusted or unknown domains; review firewall rules per CIS 4.4 and CIS 4.5

NIST Phase: Containment

Reference: NIST 800-61r3 §3.3 — Containment Strategy: implementing network and endpoint controls to limit attacker ability to deliver UAF exploit payloads via drive-by download vectors while patching is underway

Controls: CIS 4.4 (Implement and Manage a Firewall on Servers), CIS 4.5 (Implement and Manage a Firewall on End-User Devices), NIST AC-4 (Information Flow Enforcement), NIST SC-7 (Boundary Protection)

Compensating: Without a commercial proxy, configure Chrome enterprise policy to enforce SafeBrowsing at the highest level via GPO: set `HKLM\SOFTWARE\Policies\Google\Chrome\SafeBrowsingProtectionLevel` to 2 (Enhanced). Block uncategorized and newly-registered domains at the DNS layer using Pi-hole or pfSense DNS resolver with a blocklist (e.g., Hagezi or OISD). On Windows hosts, use Windows Defender Application Guard (`winget install Microsoft.ApplicationGuard`) to sandbox Chrome renderer processes for untrusted sites pending patch deployment.

Evidence: Before modifying firewall or proxy rules, capture: (1) current web proxy access logs filtered for the 24-48 hours prior to containment action, specifically request patterns to domains registered within the past 30 days or

domains serving JavaScript-heavy responses with unusual Content-Type mismatches (common UAF heap-spray delivery), and (2) DNS query logs for Chrome renderer processes, obtainable via Sysmon Event ID 22 (DNSEvent) filtered on `Image` containing `chrome.exe`. These logs are the primary evidence of pre-containment drive-by delivery attempts.

Step 5: Monitor for exploitation indicators — hunt for anomalous Chrome renderer process behavior, unexpected child process spawning from browser processes, and memory corruption crash telemetry; correlate with AU-6 (Audit Record Review, Analysis, and Reporting) requirements

NIST Phase: Detection Analysis

Reference: NIST 800-61r3 §3.2 — Detection and Analysis: analyzing telemetry for indicators that UAF vulnerabilities in Chrome were exploited prior to or during the patching window

Controls: NIST AU-6 (Audit Record Review, Analysis, and Reporting), NIST AU-2 (Event Logging), NIST AU-12 (Audit Record Generation)

Compensating: Deploy Sysmon with a configuration that captures Event ID 1 (Process Create) filtered on parent processes matching `chrome.exe` spawning non-Chrome children (e.g., `cmd.exe`, `powershell.exe`, `mshta.exe`, `wscript.exe`, `rundll32.exe`) — this is the primary post-UAF-exploitation process injection/sandbox escape signature. Additionally, monitor `%LOCALAPPDATA%\Google\CrashReports\` for new `.dmp` files via a PowerShell watcher (`Register-ObjectEvent`) and parse minidump exception codes for `0xC0000005` (Access Violation) in Chrome renderer or GPU process context, which is the Windows fault code for UAF memory corruption. For Linux: monitor `/var/crash/` and `~/.config/google-chrome/Crash Reports/` for new Breakpad dumps.

Evidence: Capture before any process termination or isolation: (1) live process tree via `Get-CimInstance Win32_Process | Where-Object {$_.Name -like '*chrome*'} | Select-Object ProcessId,ParentProcessId,CommandLine,CreationDate` to document renderer sandbox escapes, (2) Sysmon Event ID 8 (CreateRemoteThread) logs showing cross-process injection from Chrome renderer into other processes, (3) Sysmon Event ID 10 (ProcessAccess) showing Chrome renderer accessing LSASS or other sensitive processes — a post-exploitation privilege escalation indicator, and (4) Windows Error Reporting entries in the Application event log (Event ID 1000, Source: Application Error) for `chrome.exe` with faulting module `chrome.dll` at a specific offset, which pins the UAF fault location for memory forensics.

Step 6: Track the advisory — monitor the Google Chrome Releases blog and NVD for official CVE assignments, CVSS scores, and any CISA KEV additions tied to this release cycle, as source-level inconsistencies in this story mean the full scope is not yet confirmed

NIST Phase: Post Incident

Reference: NIST 800-61r3 §4 — Post-Incident Activity: integrating updated threat intelligence into organizational detection and response capability as authoritative information is confirmed

Controls: CIS 7.1 (Establish and Maintain a Vulnerability Management Process), CIS 7.2 (Establish and Maintain a Remediation Process)

Compensating: Establish a daily cron job or PowerShell scheduled task that queries the NVD API for new CVEs referencing `google chrome` with a `publishedDate` within the current release window: `Invoke-RestMethod 'https://services.nvd.nist.gov/rest/json/cves/2.0?keywordSearch=google+chrome&pubStartDate=2026-03-01T00:00:00' | ConvertTo-Json`. Subscribe via RSS to `https://chromereleases.googleblog.com/feeds/posts/default` and the CISA KEV JSON feed at `https://www.cisa.gov/sites/default/files/feeds/known_exploited_vulnerabilities.json` — parse the KEV feed daily and alert if any `vendorProject` value of `Google` with `product` of `Chrome` appears with a `dateAdded` after the Chrome 151 release date. Note: these URLs are retrieved from known authoritative sources; validate they resolve correctly before automating.

Evidence: No live state is altered by advisory monitoring. Maintain a version-controlled record of all advisory states (initial report, NVD publication, KEV addition if applicable) with timestamps, as this constitutes the threat intelligence timeline required for post-incident review and potential regulatory disclosure if exploitation is confirmed.

Detection Guidance

Because specific CVE identifiers and confirmed IOCs are not available in the provided source material, detection should focus on behavioral indicators consistent with browser-based UAF exploitation and drive-by compromise chains (T1189, T1203).

Endpoint telemetry: Hunt for Chrome renderer processes (chrome.exe --type=renderer on Windows) spawning unexpected child processes, executing PowerShell or cmd.exe, or writing executables to temp directories. UAF exploitation in browsers frequently manifests as abnormal process lineage.

Crash and stability telemetry: Sudden increases in Chrome crash reports across the fleet, particularly renderer crashes, can indicate in-the-wild exploitation attempts against unpatched instances. Correlate crash volume with endpoint patch level.

Network telemetry: Watch for browser processes initiating outbound connections to new or low-reputation domains immediately following page load, particularly to infrastructure not seen in prior baseline windows. This pattern is consistent with drive-by download staging (T1189).

Log sources to prioritize: EDR process creation logs, Windows Event Log (Sysmon Event ID 1 for process creation, Event ID 3 for network connections), and browser crash reporting telemetry. Reference NIST AU-2 (Event Logging) to confirm these event types are captured.

Asset hygiene audit: Query your software inventory for any endpoint still running a Chrome version prior to 151. Unpatched endpoints are the primary exposure condition. Reference CIS 2.1 and CIS 7.4 (Perform Automated Application Patch Management) for remediation tracking.

No specific IOC values (hashes, C2 domains, payload URLs) were present in the provided source material. Consult the Malwarebytes blog post (https://www.malwarebytes.com/blog/news/2026/06/update-chrome-to-patch-critical-browser-security-flaws) and the Google Chrome Releases blog (https://chromereleases.googleblog.com/) for any published indicators as this situation develops.

Framework Mappings

MITRE-ATTACK

- **T1189** — Drive-by Compromise
- **T1203** — Exploitation for Client Execution

NIST-800-53R5

- **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
- **7.3** — Perform Automated Operating System Patch Management
- **7.4** — Perform Automated Application Patch Management

ISO-27001-2022

- **A.8.8** — Management of technical vulnerabilities
- **A.5.23** — Information security for use of cloud services

MITRE ATT&CK Mapping

Technique ID	Technique Name	Tactic
T1189	Drive-by Compromise	Initial-Access
T1203	Exploitation for Client Execution	Execution

Sources

Source	URL	Tier
Cyberpress	https://cyberpress.org/google-chrome-update/	T3
Update Chrome to patch critical browser security flaws - Malwarebytes	https://www.malwarebytes.com/blog/news/2026/06/update-chrome-to-pat...	T1
151 Chrome Security Flaws, 22 Critical, Fixed In New Google Update	https://www.forbes.com/sites/daveywinder/2026/05/31/151-chrome-secu...	T3

Generated 2026-07-01 07:10 UTC by TJS Security Command Center