INTELLIGENCE BRIEFING
Security Command Center
TLP:CLEAR
2026-03-30 18:44 UTC

Unconfirmed No-Click Telegram Flaw Scores 9.8 CVSS, Vendor Denial Leaves Security Teams in Limbo

SECURITY ANALYSIS | CRITICAL | CVSS 7.5

SCC Item ID	SCC-STY-2026-0037
Type	Security Analysis
Severity	CRITICAL
CVSS Base Score	7.5
Affected Products	Telegram (messaging application), all platforms, specific versions unconfirmed
Published	2026-03-30T11:01:59
Discovery Source	Rss

Executive Summary

A researcher has claimed a critical zero-interaction vulnerability in Telegram, self-scoring it at 9.8 CVSS, with the alleged mechanism being a maliciously crafted sticker file capable of triggering exploitation without any user action. Telegram has publicly denied the vulnerability exists, no CVE has been officially assigned, and no vendor patch is available, leaving security teams with no authoritative remediation path. The dispute itself is the strategic signal: organizations relying on Telegram for internal communications or threat intelligence sharing must weigh continued use against unresolved and unverified risk while awaiting independent verification.

Technical Analysis

The claimed vulnerability targets Telegram's media processing pipeline, specifically the handling of sticker files. The researcher's assertion is that a specially crafted or corrupted sticker file can trigger exploitation without any user interaction, classifying it as a zero-click attack. The associated CWEs point to memory safety weaknesses: CWE-787 (out-of-bounds write), CWE-119 (improper restriction of operations within memory buffer bounds), CWE-416 (use-after-free), and CWE-20 (improper input validation). These are consistent with parser-level exploitation in media-handling libraries, a class of vulnerability with documented precedent in messaging platforms.

The MITRE ATT&CK techniques mapped to this claim include T1566.003 (Spearphishing via Service), T1059 (Command and Script Interpreter), T1204.001 (Malicious Link), and T1203 (Exploitation for Client Execution). If the vulnerability were real and weaponized, the attack chain would likely begin with the delivery of a crafted

sticker through Telegram's messaging infrastructure, triggering memory corruption at the parsing layer and potentially enabling arbitrary code execution on the target device, all without the target interacting beyond receiving the message.

Critical caveats govern everything above. Confidence in this vulnerability's technical validity is LOW. Telegram has explicitly denied the flaw's existence. No CVE has been assigned by MITRE or any CNA. The researcher-assigned score of 9.8 conflicts with a separate reported base score of 7.5, suggesting either scoring methodology disagreement or reporting inconsistency across secondary sources. All five sources in the feed are Tier 3: secondary reporting, aggregators, or community content. No NVD entry, no CISA advisory, and no independent technical proof-of-concept has been confirmed as of 2026-03-04.

The broader pattern here is instructive regardless of this specific claim's validity. Telegram's architecture presents genuine and documented risk for enterprise use: end-to-end encryption applies only to Secret Chats, not default cloud-based chats; the platform's media auto-download features have been exploited in prior confirmed incidents; and the closed-source nature of Telegram's client-server protocol limits independent security auditing. This unverified claim lands on top of a risk profile that security teams should already have evaluated.

Action Checklist

1. Step 1: Assess exposure, inventory whether Telegram is in use across your organization, including informal use on personal devices accessing corporate networks, and whether it is used for any internal communications, threat intelligence sharing, or customer-facing channels
2. Step 2: Review controls, if Telegram is in use, verify whether auto-download of media (photos, stickers, files) is disabled in client settings across managed endpoints; confirm your MDM or UEM policy enforces this setting where possible
3. Step 3: Hold on patch deployment, no vendor patch or official advisory exists; do not act on unverified remediation guidance circulating in secondary sources; monitor MITRE NVD, CISA, and Telegram's official security channel for a formal response or CVE assignment
4. Step 4: Update threat model, log this as an open, unverified risk item in your threat register; flag it for reassessment if independent technical verification, a CVE assignment, or a CISA advisory emerges; do not treat the researcher's CVSS score as authoritative
5. Step 5: Communicate findings, brief relevant stakeholders on the unresolved status: the claim is unverified, the vendor denies it, no patch is available, and the appropriate response is heightened monitoring rather than emergency action; avoid amplifying unconfirmed severity to leadership without this context
6. Step 6: Monitor developments, track MITRE NVD, CISA KEV, and Telegram's official announcements at https://t.me/durov for any acknowledgment, patch release, or CVE assignment; also watch for proof-of-concept publication that would shift the risk calculus significantly

IR / Forensic Enrichment

Triage Priority	DEFERRED

Escalation Criteria	Escalate immediately to active incident response (urgent or immediate priority) if any of the following occur: a CVE is officially assigned by MITRE/NVD, CISA adds a Telegram entry to the KEV catalog, an independent researcher publishes a verified PoC demonstrating no-click sticker exploitation, Telegram issues a security advisory or emergency update, or internal monitoring detects anomalous Telegram client behavior (unexpected process spawning, outbound connections to non-Telegram infrastructure, or tdata cache modifications) on endpoints where auto-download was confirmed disabled — the last condition would suggest an alternate exploitation vector not dependent on the claimed sticker mechanism.
Recovery Notes	If exploitation is later confirmed and an affected endpoint is identified, recovery requires full reimaging of the Telegram client installation rather than in-place repair, as a no-click exploit with no user interaction provides no reliable behavioral indicator of when compromise occurred — the infection window is unknown. Post-recovery, verify the reinstalled Telegram client version matches the latest release in Telegram's official changelog (https://desktop.telegram.org/changelog) and confirm auto-download remains disabled in the fresh installation before reconnecting to corporate resources. Monitor the recovered endpoint for 30 days post-reimaging using Sysmon Event ID 1 (Process Create) and Event ID 3 (Network Connect) filtered on the Telegram process name, watching for any child process spawning or unexpected outbound connections that were not present in the pre-incident baseline.
Forensic Artifacts	Telegram Desktop tdata directory (%APPDATA%\Telegram Desktop\tdata\ on Windows; ~/Library/Application Support/Telegram Desktop/tdata/ on macOS; ~/.local/share/TelegramDesktop/tdata/ on Linux) — contains cached sticker packs, downloaded media blobs, and local message database; a malicious sticker payload would be written to this cache directory upon receipt if auto-download was enabled at time of delivery | Telegram Desktop local SQLite message database within tdata — stores message metadata including sender identifiers, message timestamps, and content type flags; forensic analysis can confirm receipt of sticker-type messages from external or unknown contacts in the window preceding any anomalous behavior | Windows Sysmon Event ID 1 (Process Create) logs filtered on the Telegram.exe process as parent — a no-click sticker exploit achieving code execution would manifest as unexpected child processes (cmd.exe, powershell.exe, rundll32.exe, or a shellcode-injected process) spawned by Telegram.exe; absence of Sysmon requires manual review of Windows Security Event Log Event ID 4688 with process creation auditing enabled | Network flow logs (firewall, proxy, or DNS) for Telegram client connections — legitimate Telegram traffic uses MTProto protocol to known DC IP ranges (91.108.4.0/22, 91.108.56.0/22, 149.154.160.0/20, 149.154.164.0/22); any Telegram process making outbound connections to IPs outside these ranges, particularly to single-use or bulletproof hosting infrastructure, would indicate post-exploitation C2 activity | Mobile device Telegram cache directories (iOS: Library/Application Support/Telegram/ within the Telegram app sandbox; Android: /data/data/org.telegram.messenger/files/ and /sdcard/Telegram/) — on mobile platforms where the claimed vulnerability may affect the sticker rendering engine, cached sticker files with anomalous file sizes, unexpected file types masquerading as .webp sticker format, or modification timestamps inconsistent with user activity patterns would be primary forensic indicators

Per-Action IR Details

Step 1: Assess exposure — inventory whether Telegram is in use across your organization, including informal use on personal devices accessing corporate networks, and whether it is used for any internal communications, threat intelligence sharing, or customer-facing channels

NIST Phase: Preparation

Reference: NIST 800-61r3 §2 — Preparation: Establish IR capability and asset awareness before an incident is confirmed

Controls: NIST IR-4 (Incident Handling) — preparation component requires knowing which assets and applications are in scope, NIST RA-2 (Security Categorization) — categorize risk of Telegram-dependent workflows (e.g., threat intel sharing channels) against the unverified no-click sticker exploitation claim, CIS 1.1 (Establish and Maintain Detailed Enterprise Asset Inventory) — extend asset inventory to include messaging applications, with specific attention to Telegram on BYOD and unmanaged endpoints, CIS 2.1 (Establish and Maintain a Software Inventory) — Telegram clients across Windows, macOS, Linux, iOS, and Android must appear in software inventory; shadow installs on personal devices accessing corporate Wi-Fi or VPN are the primary blind spot here

Compensating: For teams without MDM visibility into personal devices: run a network query to identify Telegram traffic by checking DNS logs or firewall logs for connections to Telegram's known API domains (api.telegram.org, *.telegram.org, DC IP ranges 91.108.4.0/22, 91.108.56.0/22, 149.154.160.0/20). On Windows endpoints you do control, run: `Get-ItemProperty HKLM:\SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall\* | Where-Object DisplayName -like '*Telegram*'` and the equivalent for HKCU. On Linux/macOS, run `find / -name 'Telegram' -type f 2>/dev/null`. Cross-reference results with HR-provided BYOD registration records if available.

Evidence: Before scoping, preserve a point-in-time snapshot of network flow data showing Telegram API connections (api.telegram.org, MTProto protocol on TCP 443 and TCP 80) from both managed and guest/BYOD network segments. Capture firewall or proxy logs showing source IPs, timestamps, and data volumes for Telegram traffic in the 30 days preceding discovery. This baseline establishes which devices were active Telegram clients and could be relevant if exploitation is later confirmed.

Step 2: Review controls — if Telegram is in use, verify whether auto-download of media (photos, stickers, files) is disabled in client settings across managed endpoints; confirm your MDM or UEM policy enforces this setting where possible

NIST Phase: Containment

Reference: NIST 800-61r3 §3.3 — Containment Strategy: Implement controls to limit the impact of a potential incident while preserving the ability to operate

Controls: NIST SI-3 (Malicious Code Protection) — the alleged exploit mechanism is a maliciously crafted sticker file processed at the client layer; disabling auto-download prevents the Telegram client from automatically fetching and rendering sticker/media payloads without user intent, NIST CM-7 (Least Functionality) — restrict Telegram client functionality to the minimum required; auto-download of media is non-essential for most enterprise use cases and expands the attack surface for this claimed no-click vector, NIST SC-18 (Mobile Code) — sticker rendering in Telegram clients involves client-side processing of remotely sourced content; this control directly applies to restricting execution of untrusted mobile/remote content, CIS 4.6 (Securely Manage Enterprise Assets and Software) — enforce Telegram client configuration via MDM (Jamf, Intune, SCCM) policy to set auto-download to disabled; document the configuration baseline

Compensating: For teams without MDM/UEM: distribute a step-by-step configuration guide to end users for each platform — Telegram Desktop: Settings > Notifications and Sounds > uncheck automatic media download; Telegram mobile (iOS/Android): Settings > Data and Storage > set Auto-Download Media to 'Never' for all connection types. For managed Windows endpoints without MDM, use a Group Policy logon script or PowerShell remediation: check the Telegram Desktop config file at `%APPDATA%\Telegram Desktop\tdata\` for media download settings and alert if not set to disabled. Validate compliance manually by spot-checking 10–15 endpoints per network segment.

Evidence: Before modifying any Telegram client settings, preserve a forensic copy of the Telegram Desktop local data directory (`%APPDATA%\Telegram Desktop\tdata\` on Windows, `~/.local/share/TelegramDesktop/tdata/` on Linux, `~/Library/Application Support/Telegram Desktop/tdata/` on macOS). This directory contains cached media files, downloaded sticker packs, and configuration state. If exploitation occurred via a malicious sticker prior to this step, sticker cache files and downloaded media blobs in this directory would be the primary on-disk artifacts. Preserve directory contents and timestamps before any configuration changes overwrite or purge the cache.

Step 3: Hold on patch deployment — no vendor patch or official advisory exists; do not act on unverified remediation guidance circulating in secondary sources; monitor MITRE NVD, CISA, and Telegram's official security channel for a formal response or CVE assignment

NIST Phase: Preparation

Reference: NIST 800-61r3 §2 — Preparation: Maintain accurate situational awareness and authoritative source discipline as part of IR readiness; acting on unverified guidance can introduce new risk

Controls: NIST SI-5 (Security Alerts, Advisories, and Directives) — explicitly requires receiving alerts from defined external organizations (CISA, NVD); this step enforces source authority by prohibiting action on non-authoritative secondary sources pending official confirmation, NIST RA-3 (Risk Assessment) — the absence of a CVE, vendor acknowledgment, or independent technical verification means the risk cannot be quantified; document this as an open assessment item with explicit reassessment triggers, NIST IR-8 (Incident Response Plan) — the IR plan should define authoritative sources for vulnerability intelligence; this step exercises that definition against a contested claim, CIS 7.1 (Establish and Maintain a Vulnerability Management Process) — the vulnerability management process must distinguish between unverified researcher claims and confirmed vulnerabilities; applying emergency patch procedures to an unconfirmed finding with no CVE violates this process

Compensating: Configure free RSS monitoring for the following authoritative feeds: NVD New CVE feed (https://nvd.nist.gov/feeds/xml/cve/misc/nvd-rss.xml), CISA Known Exploited Vulnerabilities catalog (https://www.cisa.gov/sites/default/files/feeds/known_exploited_vulnerabilities.json), and Telegram's official announcements channel (t.me/durov). Use a free RSS aggregator (Feedly free tier, NewsBlur, or a local Miniflux instance) to alert on keywords: 'Telegram', 'sticker', 'zero-click', 'CVE-202*-*'. Set a calendar-based reassessment reminder at 7-day intervals. Log the monitoring initiation date and assigned owner in your risk register entry for this item.

Evidence: Preserve all secondary-source remediation guidance encountered (URLs, screenshots, social media posts, blog articles) as dated artifacts in your risk register entry. This documentation establishes what unverified guidance was circulating and confirms your team did not act on it — relevant if a regulator or auditor later asks about your decision-making process during the disputed window. Archive using a tool like SingleFile browser extension or wget for static page capture.

Step 4: Update threat model — log this as an open, unverified risk item in your threat register; flag it for reassessment if independent technical verification, a CVE assignment, or a CISA advisory emerges; do not treat the researcher's CVSS score as authoritative

NIST Phase: Post Incident

Reference: NIST 800-61r3 §4 — Post-Incident Activity: Lessons learned and continuous improvement apply to unresolved threat intelligence items, not only closed incidents; maintaining a living threat register is part of this function

Controls: NIST IR-5 (Incident Monitoring) — track and document this open item as a potential precursor event; the disputed claim and vendor denial are themselves intelligence data points requiring documented status, NIST RA-3 (Risk Assessment) — formally assess likelihood and impact given current evidence state: unverified researcher claim, CVSS 9.8 self-scored (not NVD-assigned), vendor denial, no PoC, no CVE; document the assessment rationale explicitly, NIST PM-16 (Threat Awareness Program) — integrate this dispute into your threat awareness program as a case study in contested vulnerability disclosure; the no-click sticker vector for a widely-used messaging application is a plausible threat class even if this specific instance is unverified, CIS 7.2 (Establish and Maintain a Remediation Process) — the remediation process must accommodate risk items with no available remediation path; document the item with explicit reassessment triggers rather than leaving it in an ambiguous 'waiting' state

Compensating: Use a free risk register template (NIST's risk register guidance in SP 800-39 Appendix G, or a simple spreadsheet) with the following fields for this entry: Item ID, Date Logged, Threat Description (no-click Telegram sticker exploit claim), Affected Asset (Telegram clients on managed and BYOD endpoints), Current Evidence State (unverified researcher claim / vendor denial / no CVE), Self-Reported CVSS (9.8 — researcher-assigned, not authoritative), Organizational CVSS Assessment (pending independent verification), Compensating Controls Applied (auto-download disabled per Step 2), Reassessment Triggers (CVE assignment, CISA advisory, independent PoC, vendor acknowledgment), Reassessment Date (7-day cadence), Owner (named individual). Flag the CVSS field explicitly as 'researcher self-scored' to prevent the 9.8 figure from propagating without context.

Evidence: Preserve the original researcher disclosure post, any technical analysis published by independent researchers, Telegram's public denial statement, and the NVD search result showing no CVE assignment — all with timestamps. These artifacts establish the evidentiary basis for your risk rating at the time of logging. If a CVE is later assigned with a different CVSS score, the preserved artifacts document that your team's risk assessment was

calibrated to available evidence, not to an unverified self-scored number.

Step 5: Communicate findings — brief relevant stakeholders on the unresolved status: the claim is unverified, the vendor denies it, no patch is available, and the appropriate response is heightened monitoring rather than emergency action; avoid amplifying unconfirmed severity to leadership without this context

NIST Phase: Detection Analysis

Reference: NIST 800-61r3 §3.2 — Detection and Analysis: Communicate incident status accurately to authorized staff; the 9.8 CVSS figure without context constitutes a misleading severity signal that could drive disproportionate response

Controls: NIST IR-6 (Incident Reporting) — reporting to stakeholders must accurately represent the current state of analysis; presenting an unverified CVSS 9.8 to leadership without the researcher self-score caveat and vendor denial violates accurate reporting requirements, NIST IR-7 (Incident Response Assistance) — if stakeholder pressure results in demands for emergency action that outpace the evidence, escalate to your IR support resource or legal/compliance team to contextualize the unverified claim, NIST AU-6 (Audit Record Review, Analysis, and Reporting) — the reporting component of this control applies to communicating the results of your analysis process, including the finding that the current evidence does not support emergency response classification, CIS 7.1 (Establish and Maintain a Vulnerability Management Process) — stakeholder communication is part of the vulnerability management workflow; the process should include a defined communication template that separates confirmed from unconfirmed findings

Compensating: Prepare a one-page situational brief using the following structure: (1) What was claimed — no-click Telegram sticker exploit, researcher CVSS 9.8; (2) What is verified — Telegram is in use in our environment [yes/no/partial], auto-download settings reviewed [status]; (3) What is NOT verified — the vulnerability mechanism, any CVE, any vendor acknowledgment; (4) Vendor position — Telegram publicly denies the vulnerability exists; (5) Our response — auto-download disabled as precaution, monitoring authoritative sources, reassessment triggers defined; (6) What would change our response — CVE assignment, CISA advisory, or independent PoC. Distribute via your existing incident communication channel (email, ticketing system, or Slack/Teams) with a clear 'UNVERIFIED — Monitoring' classification header.

Evidence: Before the stakeholder brief, document the internal analysis process: who reviewed the researcher's claim, what sources were consulted, what the NVD search returned (no results), and what Telegram's official position states. This creates an audit trail showing due diligence was performed and that the 'heightened monitoring' recommendation was a reasoned conclusion, not inaction. Retain the brief itself as a dated artifact in your incident tracking system.

Step 6: Monitor developments — track MITRE NVD, CISA KEV, and Telegram's official announcements at https://t.me/durov for any acknowledgment, patch release, or CVE assignment; also watch for proof-of-concept publication that would shift the risk calculus significantly

NIST Phase: Detection Analysis

Reference: NIST 800-61r3 §3.2 — Detection and Analysis: Continuously monitor threat intelligence sources and integrate new information to update the estimated scope and impact of the potential incident

Controls: NIST SI-5 (Security Alerts, Advisories, and Directives) — maintain active monitoring of CISA, NVD, and vendor channels as defined authoritative sources; PoC publication is a categorical escalation trigger that changes the threat from theoretical to actionable, NIST IR-5 (Incident Monitoring) — the open risk register item from Step 4 requires active status tracking; this step operationalizes that tracking with specific sources and escalation criteria, NIST DE.AE-07 (Cyber Threat Intelligence Integration) — from NIST 800-61r3 knowledge base: 'Rapidly acquire and analyze vulnerability disclosures for the organization's technology'; PoC publication for a no-click messaging app exploit would require immediate re-evaluation of the Step 2 compensating controls as sufficient, CIS 7.1 (Establish and Maintain a Vulnerability Management Process) — the vulnerability management process must define how new intelligence (CVE assignment, PoC release, CISA KEV addition) triggers reassessment and escalation to active remediation workflow

Compensating: Configure the following no-cost monitoring stack: (1) NVD CVE search alert — bookmark https://nvd.nist.gov/vuln/search with keyword 'Telegram' and check on the 7-day reassessment cadence; (2) CISA KEV JSON feed — use a daily cron job or scheduled task to pull https://www.cisa.gov/sites/default/files/feeds/known_exploited_vulnerabilities.json and grep for 'Telegram'; (3) GitHub

PoC monitoring — set a GitHub search alert or use grep.app to watch for new repositories tagged 'telegram', 'sticker', 'zero-click', or 'no-click' in the description; (4) Threat intel community monitoring — watch VulnDB, Packet Storm, and Exploit-DB for Telegram-related submissions. If a PoC is published: immediately escalate to active containment, consider emergency Telegram client removal from managed endpoints, and reassess whether the Step 2 auto-download disable is sufficient or whether full application removal is required.

Evidence: If a PoC is published or a CVE is assigned, immediately collect: (1) Telegram client version numbers from all managed endpoints (check registry at `HKLM:\SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall\` for Windows Telegram Desktop version string, or `mdm query` via Jamf/Intune for mobile); (2) Telegram Desktop tdata cache contents (`%APPDATA%\Telegram Desktop\tdata\`) for forensic preservation before any removal; (3) Network logs for the 72-hour window surrounding PoC publication showing Telegram API connections, specifically any unusual data transfer volumes that could indicate sticker payload delivery; (4) Any Telegram message history accessible via the client's local SQLite database (located within tdata) for evidence of receipt of sticker content from unknown senders.

Detection Guidance

Given the zero-click nature of the alleged exploit and the lack of confirmed IOCs, detection cannot be scoped to a specific artifact. Security teams should focus on behavioral and telemetry-based hunting.

Endpoint telemetry: Monitor for anomalous process execution spawned from Telegram client processes (e.g., Telegram.exe on Windows, Telegram on macOS/Linux, or the Telegram app on mobile). Any unexpected child process, outbound network connection, or file write originating from the Telegram process should be investigated. EDR platforms should alert on process injection or memory anomalies associated with the Telegram process.

Network telemetry: Flag unexpected outbound connections from Telegram client processes to non-Telegram infrastructure. Telegram's legitimate CDN and server ranges are documented; connections outside those ranges from the Telegram process warrant review.

Media processing: If your environment permits, consider restricting or logging Telegram media auto-download through network proxy or DLP controls. Unusual inbound sticker or animated media files processed at high frequency could indicate probing activity if this vulnerability moves toward active exploitation.

Log sources to review: EDR process tree logs, proxy/firewall logs filtered on Telegram client user-agent or process origin, and mobile device management (MDM) logs for anomalous app behavior on managed devices.

Hunting hypothesis (labeled as hypothetical): If this vulnerability were exploited in the wild, initial indicators might include Telegram client crashes correlated with receipt of media content, followed by anomalous process execution or network beaconing. Hunting for Telegram process crashes alongside subsequent outbound connections in the same 60-second window is a low-cost hypothesis to run against existing EDR data.

Baseline caveat: No confirmed exploitation has been reported. These detection approaches are precautionary and appropriate to the unverified threat level. Reassess and tighten guidance if a CVE is assigned or proof-of-concept code becomes public.

Framework Mappings

MITRE-ATTACK

- **T1566.003** — Spearphishing via Service
- **T1059** — Command and Scripting Interpreter

- **T1204.001** — Malicious Link
- **T1203** — Exploitation for Client Execution

NIST-800-53R5

- **CM-7** — Least Functionality
- **SI-3** — Malicious Code Protection
- **SI-4** — System Monitoring
- **SI-7** — Software, Firmware, and Information Integrity
- **SC-7** — Boundary Protection
- **SI-2** — Flaw Remediation
- **SI-10** — Information Input Validation
- **SI-16** — Memory Protection

OWASP-TOP10-2021

- **A03:2021** — Injection

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

SOC2-TSC

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

MITRE ATT&CK Mapping

Technique ID	Technique Name	Tactic
T1566.003	Spearphishing via Service	Initial-Access
T1059	Command and Scripting Interpreter	Execution
T1204.001	Malicious Link	Execution
T1203	Exploitation for Client Execution	Execution

Sources

Source	URL	Tier
Security News	https://www.darkreading.com/application-security/storm-brews-critic...	T3
Telegram's Critical Security Vulnerability Exposes Users to ...	https://www.mexc.com/news/987930	T3
Telegram CVEs and Security Vulnerabilities - OpenCVE	https://app.opencve.io/cve/?product=telegram&vendor=telegram	T3
Telegram Vulnerability Exposed: Secret Chats at Risk	https://www.linkedin.com/posts/enigma-security_cybersecurity-vulner...	T3
Is Telegram safe to use? Learn why cybercriminals...	https://www.kelacyber.com/academy/cti/is-telegram-safe-to-use/	T3

Generated 2026-03-30 18:44 UTC by TJS Security Command Center