INTELLIGENCE BRIEFING
Security Command Center
TLP:CLEAR
2026-05-19 06:43 UTC

SHub Reaper Bypasses Apple's Terminal Lockdown with AppleScript Delivery, Wallet Hijacking and Backdoor Included

THREAT CAMPAIGN | **HIGH** | CVSS 7.5

SCC Item ID	SCC-CAM-2026-0334
Type	Threat Campaign
Severity	HIGH
CVSS Base Score	7.5
Affected Products	macOS (targeted, Apple Tahoe 26.4 mitigation bypassed); Browsers: Google Chrome, Mozilla Firefox, Brave, Microsoft Edge, Opera, Vivaldi, Arc, Orion; Crypto Wallets: MetaMask, Phantom, Exodus, Atomic Wallet, Ledger Live, Electrum, Trezor Suite; Password Managers: 1Password, Bitwarden, LastPass; Other: iCloud, Telegram; Lure vectors: WeChat (fake installer), Miro (fake installer)
Published	2026-05-18T17:42:20
Discovery Source	Rss

Executive Summary

SHub Reaper is a macOS infostealer campaign that bypasses Apple's March 2026 Terminal restriction by delivering malicious code through the AppleScript URL scheme, requiring only a single user click on a social-engineered lure. The malware targets employees on macOS systems, harvesting browser credentials, password manager vaults, iCloud and Telegram sessions, and cryptocurrency wallet seed phrases. The highest business risk is wallet binary replacement, which enables persistent financial theft and backdoor access well beyond the initial infection event.

Technical Analysis

SHub Reaper is a new variant of the SHub macOS infostealer family. Confidence: Medium (sourced from SentinelOne threat research; independent verification recommended before defensive action). It pivots from Terminal-based ClickFix delivery (partially blocked in macOS Tahoe 26.4, March 2026) to the applescript:// URL scheme (T1059.002), which executes AppleScript payloads without spawning a visible Terminal window. The attack chain opens with social engineering lures (T1566) impersonating Apple security updates, Google, Microsoft, WeChat, and Miro installers to induce a single click. Upon execution: (1) Browser credential, cookie, and autofill data is harvested across Chromium and Gecko-based browsers including Chrome, Firefox, Brave, Edge, Opera, Vivaldi, Arc, and Orion (T1555.003, T1539). (2) Password manager vault data is targeted for

1Password, Bitwarden, and LastPass (T1555). (3) iCloud and Telegram session tokens are exfiltrated. (4) Crypto wallet binaries for Exodus, Atomic Wallet, Ledger Live, Electrum, and Trezor Suite are located and replaced with trojanized versions, enabling persistent seed phrase and signing key theft (T1195.002, CWE-506). (5) A LaunchAgent plist is written to ~/Library/LaunchAgents for login persistence (T1543.004), establishing a C2 beacon over HTTP/S (T1071.001) that supports remote code execution. Gatekeeper bypass is achieved via xattr -cr and ad hoc signing (T1553.001, CWE-693). Obfuscation uses ASCII art and dynamic script construction (T1140, T1027). Sandbox evasion performs system checks before payload execution (T1497.001). Relevant CWEs: CWE-506 (embedded malicious code), CWE-427 (uncontrolled search path, wallet binary replacement), CWE-693 (Gatekeeper bypass), CWE-312 (cleartext credential stores targeted), CWE-494 (payload fetched without integrity check).

Action Checklist

1. Containment. Block applescript:// URL scheme execution at the macOS system level via MDM policy (restrict AppleScript execution or disable applescript:// scheme handler). Prevent applescript:// links from rendering in browsers via managed configuration profiles. Isolate any macOS endpoint where Reaper execution is suspected. Source: SentinelOne SHub Reaper research.
2. Detection. Search endpoint telemetry for: LaunchAgent plist creation events in ~/Library/LaunchAgents from non-standard parent processes; osascript or applescript:// invocations spawned from browser processes; xattr -cr and codesign --force executions targeting wallet application directories; curl or wget fetching scripts from non-corporate domains post-browser interaction. Review EDR process trees for osascript → curl → plist write sequences.
3. Eradication. Remove malicious LaunchAgent plists from ~/Library/LaunchAgents on affected hosts. Reinstall all crypto wallet applications (Exodus, Atomic Wallet, Ledger Live, Electrum, Trezor Suite) from official vendor sources after verifying binary integrity. Rotate all credentials stored in targeted browsers and password managers on affected machines. Revoke iCloud and Telegram sessions from the account security dashboards.
4. Recovery. Verify wallet binary hashes against official vendor-published checksums before restoring financial operations. Confirm LaunchAgent persistence paths are clean. Monitor C2-indicative outbound traffic patterns from previously affected endpoints for at least 30 days post-remediation. Require re-authentication from affected user accounts across all enterprise SSO-connected services.
5. Post-Incident. Evaluate MDM policy coverage for applescript:// URL scheme blocking on all managed macOS endpoints. Assess whether macOS Tahoe 26.4 is deployed across the fleet and enforce the update. Review employee security awareness content to include AppleScript lure recognition alongside existing ClickFix guidance. Audit which users have locally installed crypto wallet or financial applications, and determine whether policy controls are warranted.

IR / Forensic Enrichment

Triage Priority	IMMEDIATE

Escalation Criteria	Escalate to CISO and legal counsel immediately if any affected user held crypto wallet seed phrases or private keys on the compromised endpoint, if iCloud or SSO credentials were confirmed exfiltrated (indicating potential PII/financial data breach triggering state breach notification obligations), or if the IR team lacks macOS forensic capability to verify wallet binary replacement on production financial systems.
Recovery Notes	Before restoring any financial operations, verify SHA-256 hashes of all reinstalled wallet binaries (Exodus, Atomic Wallet, Ledger Live, Electrum, Trezor Suite) against vendor-published checksums, as SHub Reaper's wallet binary replacement technique means a visually intact application may still be trojanized. Monitor all previously affected endpoints for C2-indicative outbound DNS queries and HTTPS connections to non-corporate destinations for a minimum of 30 days, paying particular attention to periodic beaconing patterns (fixed-interval connections suggesting a persistent LaunchAgent that survived remediation). Any crypto wallet seed phrases or private keys that were accessible on a compromised endpoint must be treated as fully compromised and migrated to a new wallet address, regardless of whether active exfiltration is confirmed.
Forensic Artifacts	Malicious LaunchAgent plist files in ~/Library/LaunchAgents/ — created by Reaper to establish persistence; capture with 'cp -p' preserving timestamps and hash with SHA-256 before removal; plist RunAtLoad and ProgramArguments keys will reference the dropped payload script. | macOS Unified Log archive filtered for 'osascript', 'applescript', 'xattr', and 'codesign' subsystem events — captures the full AppleScript delivery chain from the browser lure click through xattr quarantine stripping and forced code signing of the replaced wallet binaries. | Trojanized wallet application bundles (specifically /Applications/Exodus.app, /Applications/Atomic\ Wallet.app, /Applications/Ledger\ Live.app) — binary replacement is Reaper's persistence mechanism for financial theft; capture the full .app bundle before reinstalling for malware analysis and hash comparison against vendor-published checksums. | Browser extension Local Storage and IndexedDB for MetaMask and Phantom at ~/Library/Application\ Support/Google/Chrome/Default/Local\ Extension\ Settings/[extension_id]/ — Reaper specifically targets these paths to harvest seed phrases and wallet session tokens; capture as SQLite/LevelDB dumps before credential rotation. | macOS TCC database at ~/Library/Application\ Support/com.apple.TCC/TCC.db — records whether osascript or the Reaper payload requested and received permissions to access sensitive resources (keychain, full disk, screen recording); query 'SELECT service, client, auth_value, last_modified FROM access' to identify permission grants correlated with the incident timeline.

Per-Action IR Details

Containment — Block the applescript:// URL scheme at the enterprise web proxy and email gateway. Prevent applescript:// links from rendering in browsers via managed configuration profiles (MDM). Isolate any macOS endpoint where Reaper execution is suspected. Source: SentinelOne SHub Reaper research.

NIST Phase: Containment

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

Controls: NIST IR-4 (Incident Handling), NIST SC-7 (Boundary Protection), NIST CM-7 (Least Functionality), CIS 4.4 (Implement and Manage a Firewall on Servers), CIS 4.5 (Implement and Manage a Firewall on End-User Devices)

Compensating: On endpoints without MDM: use macOS Launch Services database manipulation via the CLI command 'sudo /System/Library/Frameworks/CoreServices.framework/Frameworks/LaunchServices.framework/Support/lsregister -u /path/to/handler' to de-register the applescript:// scheme handler. At the network perimeter, add a URL category block for 'applescript://*' in Squid proxy (deny_info directive) or pfSense with squidGuard. For email gateways running Postfix, add a body_checks regexp rule matching 'applescript://' with REJECT action. A two-person team can deploy

these manually across priority endpoints within a 2-hour window.

Evidence: Before isolating the endpoint, capture: (1) macOS Unified Log entries via 'log collect --last 24h --output /tmp/reaper_unified.logarchive' filtering for 'osascript' and 'applescript' subsystems; (2) current LaunchAgent inventory at ~/Library/LaunchAgents/ and /Library/LaunchAgents/ with 'ls -la @timestamp' to establish baseline plist timestamps; (3) browser history and download records from ~/Library/Application Support/Google/Chrome/Default/History (SQLite) and equivalent Firefox, Brave, Arc, and Orion profile paths to identify the WeChat or Miro fake installer lure URL clicked; (4) running process snapshot via 'ps aux' and 'launchctl list' to capture any active osascript or persistence agents before network isolation kills C2 callbacks.

Detection — Search endpoint telemetry for: LaunchAgent plist creation events in ~/Library/LaunchAgents from non-standard parent processes; osascript or applescript:// invocations spawned from browser processes; xattr -cr and codesign --force executions targeting wallet application directories; curl or wget fetching scripts from non-corporate domains post-browser interaction. Review EDR process trees for osascript → curl → plist write sequences.

NIST Phase: Detection Analysis

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

Controls: NIST SI-4 (System Monitoring), NIST AU-2 (Event Logging), NIST AU-6 (Audit Record Review, Analysis, and Reporting), NIST IR-5 (Incident Monitoring), CIS 8.2 (Collect Audit Logs), CIS 7.1 (Establish and Maintain a Vulnerability Management Process)

Compensating: Without EDR, deploy Sysmon for macOS equivalent via osquery with the following query targeting the osascript → curl chain: SELECT p.pid, p.name, p.cmdline, p.parent, pp.name AS parent_name FROM processes p JOIN processes pp ON p.parent = pp.pid WHERE p.name IN ('osascript', 'curl', 'wget') AND pp.name IN ('Google Chrome', 'firefox-bin', 'Brave Browser', 'Arc', 'Orion'); Additionally, use a find command to detect recently modified wallet binaries: 'find /Applications -name '*.app' -newer /tmp/baseline_timestamp -path '*Exodus*' -o -path '*Atomic*' -o -path '*Electrum*' -o -path '*Ledger*' 2>/dev/null'. For the xattr stripping behavior, hunt with: 'log show --predicate 'eventMessage contains "xattr"' --last 48h | grep -E "(codesign|xattr)"'. Write a YARA rule targeting the LaunchAgent plist label pattern and RunAtLoad key combination written by Reaper for scanning with 'yara -r reaper_launchagent.yar ~/Library/LaunchAgents/'.

Evidence: Capture before analysis: (1) macOS Unified Log filtered for 'xpc' and 'launchd' subsystems showing plist registration events — 'log show --predicate 'subsystem == "com.apple.launchd"' --last 48h'; (2) File system metadata for all plists in ~/Library/LaunchAgents/ using 'mdls -name kMDItemFSCreationDate -name kMDItemFSContentChangeDate ~/Library/LaunchAgents/*.plist' to correlate creation time with the lure click timestamp; (3) macOS TCC (Transparency, Consent, and Control) database at ~/Library/Application Support/com.apple.TCC/TCC.db — query via 'sqlite3 ~/Library/Application\ Support/com.apple.TCC/TCC.db "SELECT service, client, auth_value FROM access WHERE client LIKE '%osascript%';"' to identify whether Reaper requested elevated permissions; (4) Network connection table at time of execution via 'netstat -an' or osquery 'SELECT * FROM process_open_sockets WHERE pid IN (SELECT pid FROM processes WHERE name='osascript');' to identify C2 IP/domain.

Eradication — Remove malicious LaunchAgent plists from ~/Library/LaunchAgents on affected hosts. Reinstall all crypto wallet applications (Exodus, Atomic Wallet, Ledger Live, Electrum, Trezor Suite) from official vendor sources after verifying binary integrity. Rotate all credentials stored in targeted browsers and password managers on affected machines. Revoke iCloud and Telegram sessions from the account security dashboards.

NIST Phase: Eradication

Reference: NIST 800-61r3 §3.4 — Eradication

Controls: NIST IR-4 (Incident Handling), NIST SI-2 (Flaw Remediation), NIST SI-7 (Software, Firmware, and Information Integrity), NIST IA-5 (Authenticator Management), CIS 2.3 (Address Unauthorized Software), CIS 5.3 (Disable Dormant Accounts)

Compensating: Without automated software management tooling, execute eradication manually in sequence: (1) Run 'launchctl unload ~/Library/LaunchAgents/[reaper_plist].plist && rm ~/Library/LaunchAgents/[reaper_plist].plist' for each

identified malicious plist. (2) Verify wallet binary replacement by comparing SHA-256 hashes against vendor-published checksums — for Exodus: 'shasum -a 256 /Applications/Exodus.app/Contents/MacOS/Exodus' versus the hash published at exodus.com/releases. For Electrum, verify against the GPG-signed hashes at electrum.org. (3) Remove and reinstall browser extensions for MetaMask and Phantom from official extension stores only, as Reaper targets extension storage directories at '~/Library/Application Support/Google/Chrome/Default/Local Extension Settings/'. (4) Use ClamAV with the freshclam-updated database to scan /Applications and ~/Library before reinstalling: 'clamscan -r --bell /Applications ~/Library/Application\ Support'.

Evidence: Before removing any artifact: (1) Forensically copy all malicious LaunchAgent plists with 'cp -p ~/Library/LaunchAgents/[reaper].plist /evidence/launchagents/' preserving metadata timestamps for chain of custody; (2) Export the full contents of targeted browser credential stores — Chrome Login Data at '~/Library/Application Support/Google/Chrome/Default/Login Data' (SQLite) and equivalent paths for Firefox, Brave, Arc, and Orion — as evidence of what data was accessible to the stealer; (3) Capture the modified wallet application bundle via 'tar -czpf /evidence/exodus_bundle.tar.gz /Applications/Exodus.app' before reinstalling, preserving the trojanized binary for malware analysis; (4) Screenshot or export iCloud active sessions from appleid.apple.com and Telegram active sessions list before revoking, documenting unauthorized session timestamps and device identifiers.

Recovery — Verify wallet binary hashes against official vendor-published checksums before restoring financial operations. Confirm LaunchAgent persistence paths are clean. Monitor C2-indicative outbound traffic patterns from previously affected endpoints for at least 30 days post-remediation. Require re-authentication from affected user accounts across all enterprise SSO-connected services.

NIST Phase: Recovery

Reference: NIST 800-61r3 §3.5 — Recovery

Controls: NIST SI-7 (Software, Firmware, and Information Integrity), NIST IR-4 (Incident Handling), NIST AU-6 (Audit Record Review, Analysis, and Reporting), NIST IA-5 (Authenticator Management), CIS 7.2 (Establish and Maintain a Remediation Process), CIS 6.2 (Establish an Access Revoking Process)

Compensating: Without SIEM for 30-day C2 monitoring, configure a cron job on each recovered endpoint to run every 6 hours: 'netstat -an | grep ESTABLISHED >> /var/log/reaper_netmon.log' and parse results weekly with 'sort -u /var/log/reaper_netmon.log | grep -v [known_corporate_CIDRs]'. For binary integrity verification without commercial tooling, create a local SHA-256 manifest of all reinstalled wallet binaries using 'shasum -a 256 /Applications/*.app/Contents/MacOS/* > /var/db/wallet_baseline_hashes.txt' and run a weekly cron comparison: 'shasum -a 256 -c /var/db/wallet_baseline_hashes.txt 2>&1 | grep FAILED'. For SSO re-authentication enforcement without an IdP admin console, coordinate with Okta, Azure AD, or Google Workspace admins to invalidate all active tokens for affected user accounts and force re-enrollment of MFA factors.

Evidence: Before restoring financial operations: (1) Re-run the osquery process socket query against the recovered endpoint to confirm no osascript or suspicious curl processes have re-established C2 connections; (2) Verify LaunchAgent cleanliness with 'launchctl list | grep -v apple | grep -v com.google | grep -v [known_legitimate_vendors]' and cross-reference against the pre-incident baseline; (3) For each reinstalled wallet, document the vendor-published checksum source URL, the hash value, and the local computed hash in the incident record to satisfy NIST SI-7 (Software, Firmware, and Information Integrity) integrity verification requirements; (4) Capture the SSO session revocation confirmation (Okta system log event, Azure AD sign-in log, or Google Workspace audit log) showing forced logout timestamp for affected accounts as evidence of IA-5 (Authenticator Management) compliance.

Post-Incident — Evaluate MDM policy coverage for applescript:// URL scheme blocking on all managed macOS endpoints. Assess whether macOS Tahoe 26.4 is deployed across the fleet and enforce the update. Review employee security awareness content to include AppleScript lure recognition alongside existing ClickFix guidance. Audit which users have locally installed crypto wallet or financial applications, and determine whether policy controls are warranted.

NIST Phase: Post Incident

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

Controls: NIST IR-4 (Incident Handling), NIST IR-2 (Incident Response Training), NIST IR-8 (Incident Response Plan), NIST SI-2 (Flaw Remediation), NIST CM-7 (Least Functionality), CIS 7.3 (Perform Automated Operating System

Patch Management), CIS 2.1 (Establish and Maintain a Software Inventory), CIS 1.1 (Establish and Maintain Detailed Enterprise Asset Inventory)

Compensating: Without commercial MDM, assess macOS Tahoe 26.4 deployment fleet-wide using osquery: 'SELECT version FROM os_version;' run as a scheduled pack query across all enrolled endpoints. For the software audit of locally installed wallet and financial applications, run 'find /Applications ~/Applications -name '*.app' -maxdepth 2 | xargs -I {} mdls -name kMDItemCFBundleIdentifier {} 2>/dev/null | grep -iE "(exodus|atomic|ledger|electrum|trezor|metamask|phantom|1password|bitwarden|lastpass)"' on each endpoint and aggregate results. Update the security awareness training module to include a screenshot-based scenario showing the WeChat and Miro fake installer lure patterns used in this campaign, explicitly distinguishing the single-click AppleScript delivery mechanism from the multi-step ClickFix social engineering pattern employees may already recognize.

Evidence: For the lessons-learned record: (1) Aggregate all Unified Log archives collected during the incident into the SIEM or a centralized log store, indexed by hostname and incident ticket ID, retained per NIST AU-11 (Audit Record Retention) requirements (minimum 1 year recommended for this severity); (2) Document the MDM configuration gap that permitted applescript:// rendering in managed browsers as a finding with a remediation deadline; (3) Export the software inventory audit results showing which users had unauthorized crypto wallet installs as input to a formal policy decision on BYOA (Bring Your Own Application) financial software; (4) Record the macOS Tahoe 26.4 patch compliance percentage at incident declaration time versus post-remediation as a metric for the IR-3 (Incident Response Testing) lessons-learned report.

Detection Guidance

Primary behavioral indicators: (1) osascript process spawned by a browser process (Chrome, Firefox, Brave, Edge, Arc, Orion), this is anomalous and warrants immediate investigation. (2) LaunchAgent plist creation in ~/Library/LaunchAgents by a process other than a known installer. (3) xattr -cr or codesign --force execution targeting paths under /Applications or ~/Applications, particularly wallet app directories. (4) curl or wget network calls initiated from osascript or shell processes immediately following browser interaction. (5) File write events modifying Exodus, Atomic Wallet, Ledger Live, Electrum, or Trezor Suite application bundles. IOC patterns: look for applescript:// scheme handling in browser logs and proxy logs; outbound C2 beacon traffic (periodic, low-volume HTTP/S to non-corporate domains) from osascript or child processes post-infection. Sandbox evasion checks (system profiler queries, screen resolution checks) may appear in process argument logs before payload execution. Verify wallet binary hashes against vendor-published values as a targeted integrity check. Note: specific file hashes, domains, and IP IOCs are not available in current secondary source coverage; consult the SentinelOne SHub Reaper blog post directly for published IOCs before building detection rules.

Indicators of Compromise

Type	Value	Context	Confidence
URL	`applescript://`	Delivery mechanism — malicious AppleScript payloads delivered via the applescript:// URL scheme without invoking a visible Terminal session	MEDIUM
DOMAIN	`[not published in current secondary source coverage]`	C2 domains referenced in SentinelOne primary research — consult SentinelOne SHub Reaper blog post for specific IOCs	LOW

Framework Mappings

MITRE-ATTACK

- **T1543.004** — Launch Daemon
- **T1539** — Steal Web Session Cookie
- **T1176** — Software Extensions
- **T1140** — Deobfuscate/Decode Files or Information
- **T1555.003** — Credentials from Web Browsers
- **T1195.002** — Compromise Software Supply Chain
- **T1555** — Credentials from Password Stores
- **T1566** — Phishing
- **T1059.002** — AppleScript
- **T1036.005** — Match Legitimate Resource Name or Location
- **T1105** — Ingress Tool Transfer
- **T1547.011**
- **T1553.001** — Gatekeeper Bypass
- **T1071.001** — Web Protocols
- **T1564.001** — Hidden Files and Directories
- **T1560** — Archive Collected Data
- **T1555.001** — Keychain
- **T1041** — Exfiltration Over C2 Channel
- **T1497.001** — System Checks
- **T1204.001** — Malicious Link
- **T1027** — Obfuscated Files or Information

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
- **AT-2** — Literacy Training and Awareness
- **CA-7** — Continuous Monitoring
- **SC-7** — Boundary Protection
- **SI-3** — Malicious Code Protection
- **SI-4** — System Monitoring
- **SI-8** — Spam Protection
- **CM-3** — Configuration Change Control
- **SI-2** — Flaw Remediation

OWASP-TOP10-2021

- **A08:2021** — Software and Data Integrity Failures

CIS-V8

- **2.5** — Allowlist Authorized Software
- **2.6** — Allowlist Authorized Libraries
- **6.3** — Require MFA for Externally-Exposed Applications
- **7.3** — Perform Automated Operating System Patch Management
- **7.4** — Perform Automated Application Patch Management
- **14.2** — Train Workforce Members to Recognize Social Engineering Attacks

HIPAA-SECURITY

- **164.312(d)** — Person or Entity Authentication

SOC2-TSC

- **CC6.1** — Logical access security software, infrastructure, and architectures

ISO-27001-2022

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

MITRE ATT&CK Mapping

Technique ID	Technique Name	Tactic
T1543.004	Launch Daemon	Persistence
T1539	Steal Web Session Cookie	Credential-Access
T1176	Software Extensions	Persistence
T1140	Deobfuscate/Decode Files or Information	Defense-Evasion
T1555.003	Credentials from Web Browsers	Credential-Access
T1195.002	Compromise Software Supply Chain	Initial-Access
T1555	Credentials from Password Stores	Credential-Access
T1566	Phishing	Initial-Access
T1059.002	AppleScript	Execution
T1036.005	Match Legitimate Resource Name or Location	Defense-Evasion
T1105	Ingress Tool Transfer	Command-And-Control
T1547.011		
T1553.001	Gatekeeper Bypass	Defense-Evasion
T1071.001	Web Protocols	Command-And-Control

Technique ID	Technique Name	Tactic
T1564.001	Hidden Files and Directories	Defense-Evasion
T1560	Archive Collected Data	Collection
T1555.001	Keychain	Credential-Access
T1041	Exfiltration Over C2 Channel	Exfiltration
T1497.001	System Checks	Defense-Evasion
T1204.001	Malicious Link	Execution
T1027	Obfuscated Files or Information	Defense-Evasion

Sources

Source	URL	Tier
Security News	https://www.bleepingcomputer.com/news/security/shub-macos-infosteal...	T3
SHub Reaper | macOS Stealer Spoofs Apple, Google, and Microsoft ...	https://www.sentinelone.com/blog/shub-reaper-macos-stealer-spoofs-a...	T3
Impersonating popular cryptocurrency wallets like MetaMask ...	https://www.facebook.com/BitPinas/posts/-impersonating-popular-cryp...	T3
Should you use your browser's built-in password manager? - YouTube	https://www.youtube.com/watch?v=4CDV_AI9SG8	T3
MetaMask Crypto Wallet. Buy and Sell Bitcoin, Ethereum, Solana	https://metamask.io/	T3

Generated 2026-05-19 06:43 UTC by TJS Security Command Center