INTELLIGENCE BRIEFING
Security Command Center
TLP:CLEAR
2026-04-24 13:42 UTC

Lazarus Group Brings ClickFix Social Engineering to macOS, Targeting Executives at Mac-Heavy Organizations

THREAT CAMPAIGN | **HIGH** | CVSS 7.5

SCC Item ID	SCC-CAM-2026-0214
Type	Threat Campaign
Severity	HIGH
CVSS Base Score	7.5
Affected Products	macOS (all versions; executive and privileged users at Mac-heavy organizations)
Published	2026-04-24T09:00:00
Discovery Source	Rss

Executive Summary

North Korea's Lazarus Group has extended its ClickFix social engineering campaign to macOS, specifically targeting executives and privileged users at organizations with significant Apple hardware deployments. The attack tricks users into manually running malicious commands by presenting convincing fake error or CAPTCHA dialogs, bypassing traditional technical defenses and exploiting human behavior. Organizations with Mac-heavy environments and high-value personnel face elevated risk of credential theft, persistent access, and potential financial or intellectual property loss consistent with Lazarus Group's established objectives.

Technical Analysis

Lazarus Group (HIDDEN COBRA; overlapping with APT38 subgroup) has adapted the ClickFix technique for macOS environments. ClickFix presents victims with a fabricated browser error or CAPTCHA prompt instructing them to copy a command and paste it into Terminal or a run dialog, achieving user-initiated command execution without exploiting a software vulnerability. No CVE is assigned; the attack relies entirely on social engineering to achieve code execution. Applicable weaknesses are CWE-77 (Improper Neutralization of Special Elements used in a Command) and CWE-693 (Protection Mechanism Failure), reflecting the command injection pathway and the bypass of user awareness controls. Observed MITRE ATT&CK techniques include T1566 (Phishing), T1204.002 (Malicious File execution), T1059.004 (Unix Shell), T1105 (Ingress Tool Transfer), T1078 (Valid Accounts), T1547.011 (Plist Modification for persistence), and T1036 (Masquerading). The campaign targets macOS across all versions. No software patch resolves this threat; the vector is entirely social. Attribution is

based on TTP and infrastructure overlap with prior Lazarus operations. Primary reporting via Dark Reading; independent confirmation from CISA or MITRE was not available at time of publication. Organizations should monitor CISA and vendor threat feeds for additional corroboration.

Action Checklist

1. Containment, Identify and alert all executive and privileged macOS users within Mac-heavy environments. Issue an emergency communication warning against copying or pasting commands from browser dialogs, error messages, or CAPTCHA prompts into Terminal or any run dialog. After coordination with IT leadership, consider using MDM (e.g., Jamf, Kandji) to restrict Terminal access for executive roles that do not require command-line tools, with documented rollback procedures and 72-hour review checkpoint.
2. Detection, Review macOS endpoint logs and EDR telemetry for anomalous Terminal or shell process launches spawned from browser processes (e.g., Chrome, Safari, Brave). Query for execution of curl, bash, or sh commands that follow unusual parent-process chains. Hunt for T1059.004 indicators: unexpected shell invocations from non-administrative user accounts. Check for new or modified plist files in ~/Library/LaunchAgents and /Library/LaunchAgents (T1547.011). If available, cross-reference network logs for outbound connections to unknown infrastructure following shell execution events.
3. Eradication, Remove any identified malicious LaunchAgent plist files and kill associated processes. Revoke and rotate credentials for any account on a potentially compromised host. If a host is confirmed compromised, isolate it from the network immediately and engage incident response procedures before reconnection.
4. Recovery, Reimage confirmed compromised endpoints rather than attempting in-place cleanup, given Lazarus Group's known persistence mechanisms. Validate that no unauthorized LaunchAgents, cron jobs, or login items remain on recovered systems. Monitor reinstated accounts for anomalous authentication or access patterns for a minimum of 30 days post-remediation.
5. Post-Incident, Conduct targeted security awareness training for executive and privileged macOS users specifically addressing ClickFix-style lures. Formalize MDM policy to restrict shell access for non-technical privileged users. Review and update macOS endpoint detection rules to cover T1059.004 and T1547.011. Evaluate whether existing phishing simulation programs include terminal-command lure scenarios.

IR / Forensic Enrichment

Triage Priority	IMMEDIATE
Escalation Criteria	Escalate to senior IR leadership, legal counsel, and executive stakeholders immediately if forensic evidence confirms credential exfiltration from a privileged account, if C2 communication is detected indicating active Lazarus Group command-and-control, or if any compromised account had access to regulated data (PII, PHI, financial records) triggering breach notification obligations under GDPR, HIPAA, or applicable state law.

Recovery Notes	Reimage all confirmed compromised macOS endpoints from a known-good MDM-enrolled baseline rather than attempting manual cleanup, as Lazarus Group has demonstrated multi-stage persistence across LaunchAgents, login items, and cron jobs that resist incomplete remediation. After reinstating accounts, enforce step-up MFA verification for all privileged account logins and monitor IdP authentication logs, macOS Unified Logs, and network egress for anomalous patterns — particularly off-hours access, new device registrations, or outbound connections to unfamiliar infrastructure — for a minimum of 30 days. Validate that SIP (System Integrity Protection) and Gatekeeper are re-enabled on all recovered endpoints via `csrutil status` and `spctl --status` before returning devices to service.
Forensic Artifacts	Malicious LaunchAgent plist files in ~/Library/LaunchAgents or /Library/LaunchAgents — Lazarus ClickFix payloads use these for persistence (T1547.011); capture full XML content, creation timestamp via `mdls -name kMDItemFSCreationDate`, and SHA-256 hash before removal | macOS shell history files (~/.zsh_history, ~/.bash_history) — will contain the exact curl/bash one-liner command the victim was socially engineered to paste into Terminal, revealing the remote payload URL and any arguments passed to the Lazarus stager | macOS Unified System Log collected via `log collect --last 72h` — captures process execution events showing the parent-child relationship between the victim's browser (Chrome, Safari, Brave) and the spawned bash/zsh/curl process, confirming the ClickFix execution chain (T1059.004) | Network connection logs and DNS query history (from router, firewall, or endpoint `tcpdump` capture) — outbound connections made by curl or bash immediately following the malicious paste event will reveal Lazarus C2 infrastructure domains or IP addresses used in this campaign | IdP and SSO authentication logs (Okta, Azure AD, or Google Workspace admin console) for the 72-hour window surrounding the compromise — Lazarus Group targets executives specifically to harvest high-privilege credentials for downstream access; these logs will show whether harvested credentials were used from a foreign IP or unfamiliar device

Per-Action IR Details

Containment — Identify and alert all executive and privileged macOS users within Mac-heavy environments. Issue an emergency communication warning against copying or pasting commands from browser dialogs, error messages, or CAPTCHA prompts into Terminal or any run dialog. Temporarily restrict Terminal access via MDM (e.g., Jamf, Kandji) for Terminal access via MDM (e.g., Jamf, Kandji) for roles that do not require it.

NIST Phase: Containment

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

Controls: NIST IR-4 (Incident Handling), NIST IR-6 (Incident Reporting), NIST AC-6 (Least Privilege), CIS 4.6 (Securely Manage Enterprise Assets and Software), CIS 5.4 (Restrict Administrator Privileges to Dedicated Administrator Accounts)

Compensating: For teams without MDM: deploy a macOS configuration profile via Apple Configurator 2 (free) to restrict Terminal.app and osascript execution for standard user accounts. Alternatively, use the following command pushed via SSH to non-admin accounts: `chmod 000 /System/Applications/Utilities/Terminal.app`. Document all accounts notified with timestamps as evidence of notification scope. Draft and send alert via email explicitly describing the ClickFix lure pattern: a browser dialog (fake error or CAPTCHA) instructing the user to open Terminal and paste a command — emphasize that no legitimate service requires this.

Evidence: Before restricting Terminal access, capture the current MDM enrollment state and device compliance posture for all targeted macOS endpoints. Pull the full list of user accounts with admin rights from each Mac via `dscl . -list /Users UniqueID` and `dscl . -read /Groups/admin`. Collect current shell history files at `~/.zsh_history` and `~/.bash_history` for all executive accounts to establish a pre-containment command baseline. Export any existing Jamf or Kandji policy logs showing prior Terminal usage by privileged accounts.

Detection — Review macOS endpoint logs and EDR telemetry for anomalous Terminal or shell process launches spawned from browser processes (e.g., Chrome, Safari, Brave). Query for execution of curl, bash, or

sh commands that follow unusual parent-process chains. Hunt for T1059.004 indicators: unexpected shell invocations from non-administrative user accounts. Check for new or modified plist files in ~/Library/LaunchAgents and /Library/LaunchAgents (T1547.011). If available, cross-reference network logs for outbound connections to unknown infrastructure following shell execution events.

NIST Phase: Detection Analysis

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

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

Compensating: Without EDR, deploy osquery on macOS endpoints and run the following query to detect browser-spawned shell processes: `SELECT p.pid, p.name, p.cmdline, pp.name AS parent_name, pp.cmdline AS parent_cmdline FROM processes p JOIN processes pp ON p.parent = pp.pid WHERE p.name IN ('bash','sh','zsh','curl','python3') AND pp.name IN ('Google Chrome','Safari','Brave Browser','Firefox');`. For LaunchAgent hunting, run: `find /Library/LaunchAgents ~/Library/LaunchAgents -name '*.plist' -newer /var/db/.AppleSetupDone -ls` to surface recently created or modified plists. Enable macOS Unified Logging and query for shell spawns: `log show --predicate 'process == "bash" OR process == "zsh"' --last 7d`. Use Wireshark or `tcpdump -i en0 -w capture.pcap` to capture outbound traffic from endpoints immediately post-detection.

Evidence: Collect macOS Unified System Log (via `log collect --last 72h --output /tmp/syslog_collection`) to capture process execution telemetry around the suspected compromise window. Export all plist files from `~/Library/LaunchAgents` and `/Library/LaunchAgents` with creation and modification timestamps (`ls -la@` and `mdls` for extended attributes). Pull shell history from `~/.zsh_history` and `~/.bash_history` and cross-reference commands against known Lazarus ClickFix payload patterns (e.g., one-liner curl-to-bash fetching a remote script). Capture DNS query logs or `/etc/hosts` modifications that may reveal C2 infrastructure contacted after shell execution. Collect `/private/var/log/system.log` and Apple System Log (ASL) entries filtered for `Terminal`, `bash`, `zsh`, and `curl` process events.

Eradication — Remove any identified malicious LaunchAgent plist files and kill associated processes. Revoke and rotate credentials for any account on a potentially compromised host. If a host is confirmed compromised, isolate it from the network immediately and engage incident response procedures before reconnection.

NIST Phase: Eradication

Reference: NIST 800-61r3 §3.4 — Eradication

Controls: NIST IR-4 (Incident Handling), NIST SI-2 (Flaw Remediation), NIST AC-6 (Least Privilege), NIST IA-5 (Authenticator Management), CIS 5.3 (Disable Dormant Accounts), CIS 6.2 (Establish an Access Revoking Process)

Compensating: For each identified malicious LaunchAgent: `launchctl unload ~/Library/LaunchAgents/[malicious].plist && rm -f ~/Library/LaunchAgents/[malicious].plist`. Kill associated processes by name: `pkill -f [malicious_process_name]`. Verify removal with: `launchctl list | grep -v com.apple`. For credential rotation without an enterprise PAM tool, immediately reset macOS local account passwords via `dscl . -passwd /Users/[username]` and force SSO/IdP credential rotation (Okta, Azure AD, Google Workspace) for any account that authenticated from the compromised host. Revoke all active SSO sessions for affected accounts in the IdP admin console. Document each action with timestamps for the chain of custody record.

Evidence: Before removing any LaunchAgent plist, create a forensic copy: `cp -p ~/Library/LaunchAgents/[malicious].plist /tmp/forensic_evidence/` and compute SHA-256 hash: `shasum -a 256 [file]`. Capture the running process list at time of eradication: `ps auxww > /tmp/process_snapshot_$(date +%Y%m%d%H%M%S).txt`. Extract the full content of the malicious plist (XML keys for ProgramArguments, RunAtLoad, KeepAlive values will reveal the persistence command and C2 callback mechanism specific to the Lazarus payload). Pull authentication logs from the IdP for the compromised account covering the 72-hour window prior to detection to identify any unauthorized access or lateral movement using stolen credentials.

Recovery — Reimage confirmed compromised endpoints rather than attempting in-place cleanup, given Lazarus Group's known persistence mechanisms. Validate that no unauthorized LaunchAgents, cron jobs, or login items remain on recovered systems. Monitor reinstated accounts for anomalous authentication or

access patterns for a minimum of 30 days post-remediation.

NIST Phase: Recovery

Reference: NIST 800-61r3 §3.5 — Recovery

Controls: NIST IR-4 (Incident Handling), NIST SI-7 (Software, Firmware, and Information Integrity), NIST CP-9 (System Backup), NIST AU-6 (Audit Record Review, Analysis, and Reporting), CIS 7.3 (Perform Automated Operating System Patch Management), CIS 4.6 (Securely Manage Enterprise Assets and Software)

Compensating: Reimage using Apple Configurator 2 with a known-good macOS image, or initiate Erase All Content and Settings on macOS Ventura and later (System Settings > General > Transfer or Reset). Before reinstating the endpoint, run a post-image validation script to check for residual persistence: `for dir in ~/Library/LaunchAgents /Library/LaunchAgents /Library/LaunchDaemons; do echo "=== $dir ==="; ls -la $dir; done` and `crontab -l -u [username]`. For login item validation: `osascript -e 'tell application "System Events" to get the name of every login item'`. Enable macOS audit logging (`auditd`) before returning the device to the user: `sudo launchctl load -w /System/Library/LaunchDaemons/com.apple.auditd.plist`.

Evidence: Prior to reimaging, create a full disk image of the compromised endpoint using `dd` or Target Disk Mode if forensic investigation is warranted, storing the image with SHA-256 verification. Document the macOS version, patch level, and MDM enrollment status of the compromised host at time of incident. After reimaging, verify system integrity using `csrutil status` (SIP enabled) and `sudo /usr/sbin/spctl --status` (Gatekeeper enabled). Collect baseline LaunchAgent and LaunchDaemon inventories from the clean image for comparison against any future anomaly detection. Retain all pre-reimage evidence for minimum 90 days consistent with NIST AU-11 (Audit Record Retention) requirements.

Post-Incident — Conduct targeted security awareness training for executive and privileged macOS users specifically addressing ClickFix-style lures. Formalize MDM policy to restrict shell access for non-technical privileged users. Review and update macOS endpoint detection rules to cover T1059.004 and T1547.011. Evaluate whether existing phishing simulation programs include terminal-command lure scenarios.

NIST Phase: Post Incident

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

Controls: NIST IR-2 (Incident Response Training), NIST IR-3 (Incident Response Testing), NIST IR-8 (Incident Response Plan), NIST SI-5 (Security Alerts, Advisories, and Directives), NIST AT-2 (Literacy Training and Awareness), CIS 7.1 (Establish and Maintain a Vulnerability Management Process), CIS 7.2 (Establish and Maintain a Remediation Process)

Compensating: For detection rule updates without a commercial SIEM: deploy the community Sigma rule for T1059.004 macOS shell execution (search the SigmaHQ GitHub repository for `macos_shell_execution` rules) and convert to osquery scheduled queries or Elastic SIEM if available. For ClickFix-specific lure simulation: create a tabletop exercise scenario where a facilitator shows executives a mock browser CAPTCHA dialog instructing them to open Terminal and run a command — measure and document the response. Publish an internal one-page advisory with a screenshot of the ClickFix lure pattern (sourced from the Lazarus Group campaign reporting by ESET or Recorded Future) so executives can visually recognize the social engineering format. Set a recurring monthly osquery scheduled query to alert on new LaunchAgent plist creation: `SELECT * FROM file WHERE path LIKE '/Users/%/Library/LaunchAgents/%' AND ctime > (strftime('%s','now') - 86400);`.

Evidence: Compile a post-incident report documenting: the initial entry point (which user, which browser, which lure dialog), the specific shell command executed by the victim (recovered from `~/.zsh_history`), the LaunchAgent plist payload and its persistence mechanism, C2 infrastructure contacted (domains/IPs from network logs), and credential scope potentially exposed. This report serves as the basis for the lessons-learned session (NIST 800-61r3 §4.1) and must be retained per NIST AU-11 (Audit Record Retention). Use the incident timeline to identify detection gaps — specifically how long the malicious LaunchAgent persisted before discovery — and feed that dwell time metric into updated detection rule SLA requirements.

Detection Guidance

Focus detection on the execution chain from browser to shell. Key indicators: (1) Behavioral, shell processes (bash, zsh, sh) with a parent process of a browser (e.g., Chrome, Safari) or Finder, particularly when followed by network activity (curl, wget, python -c). (2) Persistence artifacts, new or recently modified .plist files in ~/Library/LaunchAgents or /Library/LaunchAgents created by non-root, non-system processes. (3) Network, outbound connections from Terminal or shell child processes to unknown external IPs or domains, especially shortly after a user session. (4) Log sources, macOS Unified Log (via log stream or Console.app), EDR process telemetry (CrowdStrike, SentinelOne, Jamf Protect), and macOS audit framework (auditd if enabled). (5) SIEM query pattern: process_name IN ('bash','zsh','sh') AND parent_process_name IN ('Google Chrome','Safari','firefox') AND event_type = 'process_create'. Note: As of publication date, CISA had not issued dedicated advisory; check CISA alerts and threat feeds for confirmed IOCs (hashes, domains, IPs) as they become available.

Framework Mappings

MITRE-ATTACK

- **T1036** — Masquerading
- **T1566** — Phishing
- **T1078** — Valid Accounts
- **T1547.011**
- **T1059.004** — Unix Shell
- **T1105** — Ingress Tool Transfer
- **T1204.002** — Malicious File

NIST-800-53R5

- **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
- **AC-2** — Account Management
- **AC-6** — Least Privilege
- **IA-2** — Identification and Authentication (Organizational Users)
- **IA-5** — Authenticator Management
- **CM-7** — Least Functionality
- **SI-10** — Information Input Validation

OWASP-TOP10-2021

- **A03:2021** — Injection

CIS-V8

- **16.10** — Apply Secure Design Principles in Application Architectures

- **14.2** — Train Workforce Members to Recognize Social Engineering Attacks

ISO-27001-2022

- **A.8.8** — Management of technical vulnerabilities

MITRE ATT&CK Mapping

Technique ID	Technique Name	Tactic
T1036	Masquerading	Defense-Evasion
T1566	Phishing	Initial-Access
T1078	Valid Accounts	Defense-Evasion
T1547.011		
T1059.004	Unix Shell	Execution
T1105	Ingress Tool Transfer	Command-And-Control
T1204.002	Malicious File	Execution

Sources

Source	URL	Tier
Security News	https://www.darkreading.com/threat-intelligence/north-koreas-lazaru...	T3
Apple security releases	https://support.apple.com/en-us/100100	T3
About the security content of macOS Sequoia 15.5 - Apple Support	https://support.apple.com/en-us/122716	T3
Apple Fixes Exploited Zero-Day Affecting iOS, macOS, and Other ...	https://thehackernews.com/2026/02/apple-fixes-exploited-zero-day.html	T3
macOS Spotlight Vulnerability Discovered by Microsoft - Reddit	https://www.reddit.com/r/apple/comments/1mbl9g8/macos_spotlight_vul...	T3

Generated 2026-04-24 13:42 UTC by TJS Security Command Center