INTELLIGENCE BRIEFING
Security Command Center
TLP:CLEAR
2026-06-25 18:45 UTC

macOS.Gaslight: North Korean Threat Actor Embeds Prompt Injection Strings to Blind AI-Assisted Malware Triage

THREAT CAMPAIGN | HIGH | CVSS 7.5

SCC Item ID	SCC-CAM-2026-0570
Type	Threat Campaign
Severity	HIGH
CVSS Base Score	7.5
Affected Products	macOS systems; AI-powered malware analysis platforms and LLM-assisted security triage tooling (specific vendors not identified in available reporting)
Published	2026-06-25T12:23:19
Discovery Source	Rss

Executive Summary

SentinelOne researchers identified a novel macOS malware family, attributed with high confidence to North Korea's Lazarus Group (specific sub-group unconfirmed), that embeds fabricated system-failure strings specifically designed to mislead AI-assisted malware triage tools rather than evade traditional sandboxes. Organizations that have integrated large language model (LLM) agents into their security operations pipelines face a new category of blind spot: the malware's observable behavior remains visible to conventional tools, but the AI layer's interpretation is corrupted before analysts act on it. The business risk is delayed or incorrect incident response, particularly for security teams that have increased analyst reliance on AI-generated triage summaries. Attribution and technical details are sourced from SentinelOne research; the primary publication should be reviewed for full verification.

Technical Analysis

macOS.Gaslight is a Rust-compiled macOS binary attributed to a Lazarus Group-linked North Korean threat actor (SentinelOne, high-confidence attribution; specific sub-group unconfirmed). The malware embeds 38 fabricated system-failure diagnostic strings within the binary. These strings are not sandbox evasion artifacts in the traditional sense, they are crafted to be ingested by LLM-assisted triage pipelines that process binary strings, decompiled output, or sandbox telemetry as context. When an AI agent processes the binary, the injected strings corrupt the model's interpretation, producing misleading triage conclusions that mask the malware's actual capabilities: information stealing (T1119), backdoor C2 communication over application-layer

protocols (T1071), command execution via Rust binary (T1059/T1059.004), and obfuscation via embedded fake strings (T1027). Sandbox evasion mapping falls closest to T1497, though the technique specifically targets the LLM analysis layer rather than the virtualization layer itself. Capability acquisition aligns with T1588.001. No CVE has been assigned. Applicable CWEs: CWE-74 (Improper Neutralization of Special Elements, prompt injection into LLM analysis pipeline) and CWE-116 (Improper Encoding or Escaping of Output, unsanitized binary string ingestion into LLM context windows). No vendor patch is applicable; the attack surface is the AI triage pipeline configuration, not a patchable software vulnerability. Source: SentinelOne research, surfaced via BleepingComputer RSS feed (T3); primary SentinelOne publication should be confirmed before treating attribution and technical details as fully verified.

Action Checklist

1. Step 1: Containment, Audit your SOC's LLM-assisted triage integrations immediately. Identify any pipeline that ingests raw binary strings, decompiled output, or sandbox telemetry directly into an LLM context window without sanitization. Flag those pipelines for human-in-the-loop review on all macOS binary submissions until validation controls are in place. Align with NIST IR-4 (Incident Handling), activate your incident handling capability for this emergent blind spot.
2. Step 2: Detection, Query endpoint detection telemetry for unsigned or anomalous Rust binaries on macOS endpoints (file magic byte verification per NIST SI-7 (Software, Firmware, and Information Integrity)). Review LLM triage output logs for analysis summaries that include language matching system-failure diagnostics ('disk I/O error', 'kernel panic', 'hardware fault' patterns) on files that sandbox telemetry shows active network callbacks, a mismatch between AI summary and raw sandbox output is a high-fidelity indicator. Enable local account monitoring for privilege changes following macOS binary execution. Reference NIST AU-6 (Audit Record Review, Analysis, and Reporting) for structured log review cadence. Reference NIST SI-4 (Information System Monitoring) for real-time detection of anomalous process behavior on macOS endpoints.
3. Step 3: Eradication, No patch exists. The remediation is pipeline hardening: implement input sanitization on all string data ingested from binary analysis into LLM context windows before model processing. Remove or quarantine any identified macOS.Gaslight binary from affected endpoints. Rotate credentials on any macOS host where the binary executed, per NIST AC-2 (Account Management). Align with NIST CM-6 (Configuration Settings), enforce secure configuration of AI triage pipeline components as a configuration-controlled system component.
4. Step 4: Recovery, Validate that AI triage pipeline outputs for previously analyzed macOS binaries are consistent with raw sandbox telemetry. Re-analyze any macOS binary submissions processed through LLM pipelines in the past 90 days where AI summaries and sandbox behavior diverged. Monitor for C2 callback patterns from macOS hosts that executed unknown Rust binaries (T1071, application-layer protocol). Reference CIS 8.2 (Collect Audit Logs) and CIS 8.3 (Centralize Audit Logs), confirm audit logging is active and centralized across macOS endpoints and AI pipeline components.
5. Step 5: Post-Incident, Conduct a formal review of your AI-assisted triage architecture against prompt injection risk. Implement output validation: require LLM triage summaries to cross-reference at least one structured, non-string-derived sandbox signal (network behavior, system call trace) before escalation or de-escalation decisions are made. Document the control gap, LLM pipeline input sanitization, as a formal risk item. Reference NIST IR-8 (Incident Response Plan) to update your IR plan with AI-specific triage validation steps. Reference CIS 7.1 (Establish and Maintain a Vulnerability Management Process) to formally track AI pipeline hardening as a remediation action.

IR / Forensic Enrichment

Triage Priority	URGENT
Escalation Criteria	Escalate immediately to CISO and legal/compliance if forensic review of the 90-day LLM triage backlog identifies macOS binaries that were de-escalated or closed by LLM summary despite sandbox telemetry showing active C2 callbacks, as this indicates a confirmed blind-spot exploitation window that may trigger breach notification obligations depending on data accessed on affected hosts.
Recovery Notes	After eradicating macOS.Gaslight binaries and hardening LLM pipeline input sanitization, conduct a structured re-analysis of all macOS binary submissions from the prior 90 days using human analyst review against raw sandbox telemetry — do not rely on LLM re-analysis until input sanitization controls are validated end-to-end. Monitor all macOS endpoints that executed unidentified Rust binaries for at least 30 days post-recovery, specifically watching for outbound connections to new or previously unseen IP ranges indicative of Lazarus Group infrastructure rotation. Validate that LLM triage pipeline output validation logic (requiring corroborating structured sandbox signal) is functioning correctly by running a controlled red-team test using a benign binary embedded with Gaslight-style fake-failure strings before declaring the pipeline restored to trusted operation.
Forensic Artifacts	macOS Unified Log entries (`log collect --last 720h`) for the process name and PID of the suspect Rust binary — Gaslight's prompt injection strings will appear in log output consumed by LLM triage pipelines, making these logs evidence of both execution and the injection attempt | LLM triage output logs for all macOS binary submissions in the past 90 days — specifically any summary containing 'disk I/O error', 'kernel panic', or 'hardware fault' language paired with sandbox telemetry showing active outbound network connections, representing the core Gaslight divergence indicator | macOS memory image (acquired via osxpmem before process termination) — Lazarus Group Rust implants typically maintain C2 configuration, beacon intervals, and encryption keys only in memory; these are unrecoverable after process kill or host isolation | Network PCAP or macOS `pfctl` firewall logs capturing outbound connections from the Rust binary process — Gaslight's prompt injection is specifically designed to hide active C2 callbacks from LLM summary, so raw network telemetry is the ground-truth counter-evidence | Filesystem timeline (`find / -newer -type f`) and macOS QuarantineEvents database (`~/Library/Preferences/com.apple.LaunchServices.QuarantineEventsV2`) recording initial binary drop path, download source, and execution timestamp to establish the full infection chain

Per-Action IR Details

Step 1: Containment — Audit your SOC's LLM-assisted triage integrations immediately. Identify any pipeline that ingests raw binary strings, decompiled output, or sandbox telemetry directly into an LLM context window without sanitization. Flag those pipelines for human-in-the-loop review on all macOS binary submissions until validation controls are in place. Align with NIST IR-4 (Incident Handling) — activate your incident handling capability for this emergent blind spot.

NIST Phase: Containment

Reference: NIST 800-61r3 §3.3 — Containment Strategy: Stop the spread of impact by isolating affected pipeline components from automated decision authority until input sanitization controls are validated.

Controls: NIST IR-4 (Incident Handling), NIST AC-3 (Access Enforcement), CIS 4.6 (Securely Manage Enterprise Assets and Software)

Compensating: Without an enterprise pipeline audit tool, enumerate LLM integrations manually: grep your SOAR/automation configs and cron jobs for API calls to OpenAI, Anthropic, or local LLM endpoints that accept binary-derived string input. Temporarily gate all macOS binary submissions to a human analyst queue using a Slack or ticketing webhook rather than automated LLM forwarding. Document every pipeline path in a shared spreadsheet before proceeding.

Evidence: Before modifying any pipeline configuration, export current LLM integration workflow definitions (SOAR playbook exports, API gateway logs, pipeline config files) to a read-only archive. Capture a snapshot of recent LLM triage output logs — specifically any macOS binary analyses from the past 90 days — before pipeline changes could alter or overwrite them. These logs are your ground truth for identifying which submissions may have been misdirected by Gaslight prompt injection strings.

Step 2: Detection — Query endpoint detection telemetry for unsigned or anomalous Rust binaries on macOS endpoints (file magic byte analysis supports D3-FMBV). Review LLM triage output logs for analysis summaries that include language matching system-failure diagnostics ('disk I/O error', 'kernel panic', 'hardware fault' patterns) on files that sandbox telemetry shows active network callbacks — a mismatch between AI summary and raw sandbox output is a high-fidelity indicator. Enable local account monitoring (D3-LAM) for privilege changes following macOS binary execution. Reference NIST AU-6 (Audit Record Review, Analysis, and Reporting) for structured log review cadence. Reference NIST SI-4 is not mapped in the verified control set; no mapped control for real-time system monitoring will be cited.

NIST Phase: Detection Analysis

Reference: NIST 800-61r3 §3.2 — Detection and Analysis: Correlate multiple evidence sources to identify macOS.Gaslight infections, specifically hunting for the divergence signal between LLM triage output and raw sandbox behavioral telemetry.

Controls: NIST AU-6 (Audit Record Review, Analysis, And Reporting), NIST AU-2 (Event Logging), CIS 8.2 (Collect Audit Logs), CIS 1.1 (Establish and Maintain Detailed Enterprise Asset Inventory)

Compensating: On macOS endpoints without EDR, run: `find / -type f -perm +111 -exec file {} \; 2>/dev/null | grep -i 'mach-o'` to enumerate executables, then cross-reference against `codesign -vvv ` to flag unsigned binaries. Use osquery with query `SELECT * FROM signature WHERE path LIKE '/Users/%' AND signed=0` to surface unsigned executables in user-writable paths. For the LLM log divergence check, write a simple Python script that diffs sandbox JSON telemetry (network_connections, syscalls fields) against LLM summary text, flagging any summary containing 'disk I/O error', 'kernel panic', or 'hardware fault' where sandbox output shows active outbound connections.

Evidence: Capture volatile state before any process termination: run `netstat -anp tcp` and `lsof -i` on suspect macOS hosts to record active network connections from Rust binary processes. Collect `ps aux` output to document process tree and parent-child relationships. Preserve LLM triage output logs in their unmodified state — these are primary evidence of the prompt injection attack surface. Capture macOS Unified Log entries via `log collect --last 24h` targeting the process name of the suspect binary before killing any process.

Step 3: Eradication — No patch exists. The remediation is pipeline hardening: implement input sanitization on all string data ingested from binary analysis into LLM context windows before model processing. Remove or quarantine any identified macOS.Gaslight binary from affected endpoints. Rotate credentials on any macOS host where the binary executed, per D3-CRO (Credential Rotation). Align with NIST CM-6 (Configuration Settings) — enforce secure configuration of AI triage pipeline components as a configuration-controlled system component.

NIST Phase: Eradication

Reference: NIST 800-61r3 §3.4 — Eradication: Remove macOS.Gaslight binaries from affected hosts and harden LLM pipeline input handling to eliminate the prompt injection attack vector from the environment.

Controls: NIST CM-6 (Configuration Settings), NIST AC-6 (Least Privilege), CIS 4.6 (Securely Manage Enterprise Assets and Software), CIS 5.2 (Use Unique Passwords)

Compensating: For credential rotation without a PAM tool: use macOS `dscl . -passwd /Users/` to rotate local account passwords on affected hosts; for SSH keys, revoke and regenerate with `ssh-keygen` and update

`~/.ssh/authorized_keys`. For binary removal, compute SHA-256 of the identified Gaslight binary (`shasum -a 256 `), then run `find / -name -delete` across affected hosts and verify removal. For LLM pipeline sanitization without enterprise tooling, implement a Python pre-processing wrapper that strips or escapes any string segment matching Gaslight's fake-failure patterns before passing content to the LLM API endpoint.

Evidence: BEFORE removing the binary or rotating credentials, acquire a full memory image using macOS-compatible tools (osxpmem or similar) to preserve in-memory Lazarus Group C2 configuration, encryption keys, and any injected code that will not survive process termination. Capture a forensic copy of the binary itself with `dd` and record its SHA-256 hash for threat intelligence sharing. Document all macOS Keychain entries and recently accessed credentials via `security dump-keychain` (with appropriate authorization) before rotation, to establish what the threat actor may have accessed.

Step 4: Recovery — Validate that AI triage pipeline outputs for previously analyzed macOS binaries are consistent with raw sandbox telemetry. Re-analyze any macOS binary submissions processed through LLM pipelines in the past 90 days where AI summaries and sandbox behavior diverged. Monitor for C2 callback patterns from macOS hosts that executed unknown Rust binaries (T1071 — application-layer protocol). Reference CIS 8.2 (Collect Audit Logs) — confirm audit logging is active across macOS endpoints and AI pipeline components.

NIST Phase: Recovery

Reference: NIST 800-61r3 §3.5 — Recovery: Restore trusted operation of LLM-assisted triage pipelines by validating prior outputs against ground-truth sandbox telemetry, and confirm macOS endpoints are clean before returning them to production.

Controls: CIS 8.2 (Collect Audit Logs), NIST AU-6 (Audit Record Review, Analysis, And Reporting), NIST AU-11 (Audit Record Retention)

Compensating: Without a SIEM to correlate 90 days of pipeline logs, build a manual audit spreadsheet: export LLM triage output records and sandbox JSON reports, then use a Python diffing script to flag rows where LLM summary contains Gaslight's fake-failure vocabulary but sandbox telemetry shows outbound connections or file writes. For C2 callback monitoring on macOS without EDR, configure `pfctl` (macOS packet filter) to log outbound connections from user-space processes and pipe output to a local log file reviewed daily via cron.

Evidence: Before returning any macOS host to production, collect a final `netstat -anp tcp` and `lsof -i` snapshot to confirm no residual Lazarus Group C2 connections remain active. Preserve the 90-day LLM triage output log archive as an evidentiary record of the pipeline blind spot duration — this establishes the potential exposure window for any regulatory notification assessment. Verify macOS Unified Log continuity to confirm no log gaps exist that would obscure post-eradication activity.

Step 5: Post-Incident — Conduct a formal review of your AI-assisted triage architecture against prompt injection risk. Implement output validation: require LLM triage summaries to cross-reference at least one structured, non-string-derived sandbox signal (network behavior, system call trace) before escalation or de-escalation decisions are made. Document the control gap — LLM pipeline input sanitization — as a formal risk item. Reference NIST IR-8 (Incident Response Plan) to update your IR plan with AI-specific triage validation steps. Reference CIS 7.1 (Establish and Maintain a Vulnerability Management Process) to formally track AI pipeline hardening as a remediation action.

NIST Phase: Post Incident

Reference: NIST 800-61r3 §4 — Post-Incident Activity: Conduct lessons-learned review and update the IR plan and triage architecture to close the LLM prompt injection blind spot exposed by macOS.Gaslight.

Controls: NIST IR-4 (Incident Handling), CIS 7.1 (Establish and Maintain a Vulnerability Management Process), CIS 7.2 (Establish and Maintain a Remediation Process), NIST AU-6 (Audit Record Review, Analysis, And Reporting)

Compensating: For teams without a GRC platform to track the control gap formally, open a tracked GitHub Issue or Jira ticket titled 'Risk Item: LLM Pipeline Input Sanitization — macOS.Gaslight Exposure' with severity, owner, and a 30/60/90-day remediation milestone. Author a one-page IR plan addendum specifying that any LLM triage summary recommending de-escalation of a macOS binary must be countered by at least one structured sandbox signal (pcap

evidence of no outbound connection, or strace/dtrace syscall log showing no persistence mechanism) before the case is closed.

Evidence: Compile the full incident timeline from initial Gaslight binary submission through pipeline blind spot detection as the primary lessons-learned input document. Retain all LLM triage output logs, sandbox telemetry exports, and pipeline configuration snapshots from the incident period per your audit record retention policy (NIST AU-11) — minimum retention should align with any applicable regulatory reporting window. Include the SHA-256 hashes of identified Gaslight binaries in your threat intelligence repository for future YARA rule development.

Detection Guidance

Primary detection signal: mismatch between LLM-assisted triage conclusions and raw sandbox or endpoint telemetry for macOS binary submissions. Specifically, flag any AI triage summary that characterizes a binary as benign or as generating system errors while the underlying sandbox log shows active outbound network connections, file system writes outside expected paths, or privilege escalation attempts. Secondary signals: presence of unsigned Rust binaries on macOS endpoints (Rust binaries are identifiable by characteristic ELF/Mach-O metadata and standard library signatures); file magic byte verification on macOS executables submitted for analysis can surface binaries whose string tables are abnormally dense with diagnostic-language tokens relative to actual functional code. Hunt hypothesis: query your SIEM or EDR for macOS processes spawning network connections within 60 seconds of first execution, where the binary has no prior execution history on that host and no associated software inventory entry (reference CIS 2.1, Establish and Maintain a Software Inventory). Behavioral indicator for LLM pipeline compromise: AI triage outputs referencing hardware failure, kernel panic, or I/O error conditions on a binary that exhibits active C2 callback behavior in sandbox network logs, this specific combination indicates probable prompt injection success. IOCs: none confirmed in available reporting at time of writing; monitor SentinelOne's primary publication for indicator release.

Framework Mappings

MITRE-ATTACK

- **T1027** — Obfuscated Files or Information
- **T1564** — Hide Artifacts
- **T1059** — Command and Scripting Interpreter
- **T1588.001** — Malware
- **T1119** — Automated Collection
- **T1059.004** — Unix Shell
- **T1071** — Application Layer Protocol
- **T1497** — Virtualization/Sandbox Evasion

NIST-800-53R5

- **SI-3** — Malicious Code Protection
- **SI-4** — System Monitoring
- **CM-7** — Least Functionality
- **SI-7** — Software, Firmware, and Information Integrity
- **AC-6** — Least Privilege

- **SC-7** — Boundary Protection
- **CA-7** — Continuous Monitoring
- **SI-10** — Information Input Validation

OWASP-TOP10-2021

- **A03:2021** — Injection

CIS-V8

- **16.10** — Apply Secure Design Principles in Application Architectures
- **8.2** — Collect Audit Logs

NIST-CSF-2

- **DE.CM-01** — Networks and network services are monitored

MITRE ATT&CK Mapping

Technique ID	Technique Name	Tactic
T1027	Obfuscated Files or Information	Defense-Evasion
T1564	Hide Artifacts	Defense-Evasion
T1059	Command and Scripting Interpreter	Execution
T1588.001	Malware	Resource-Development
T1119	Automated Collection	Collection
T1059.004	Unix Shell	Execution
T1071	Application Layer Protocol	Command-And-Control
T1497	Virtualization/Sandbox Evasion	Defense-Evasion

Sources

Source	URL	Tier
Security News	https://www.bleepingcomputer.com/news/security/new-macos-malware-em...	T3
Best AI Cybersecurity Tools in 2026: Top Platforms by Use Case	https://checkmarx.com/learn/ai-security/best-ai-cybersecurity-tools...	T3
Apple Alerted to macOS Security Vulnerability Uncovered With AI Tool	https://www.facebook.com/MacRumors/posts/apple-alerted-to-macos-sec...	T3

Source	URL	Tier
Apple Alerted to macOS Security Vulnerability Uncovered With AI Tool	https://www.reddit.com/r/apple/comments/1td5oqu/apple_alerted_to_ma...	T3
CVE-2025-43289: Apple macOS Information Disclosure Issue	https://www.sentinelone.com/vulnerability-database/cve-2025-43289/	T3

Generated 2026-06-25 18:45 UTC by TJS Security Command Center