H-0003: DigitStealer macOS Infostealer Detection

Hunt Metadata

- **Date:** 2025-11-18
- **Hunter:** Sydney
- **Status:** Completed
- **MITRE ATT&CK:** T1539 - Steal Web Session Cookie, T1555 - Credentials from Password Stores, T1059.002 - AppleScript

LEARN: Prepare the Hunt

Hypothesis Statement

Detect DigitStealer macOS infostealer infection attempts targeting credential stores and financial data on macOS endpoints, based on recent threat intelligence reporting. The malware uses multi-stage script execution, TCC privacy permission manipulation, and hardware-level VM detection to evade analysis while exfiltrating credentials, browser data, and cryptocurrency wallets.

ABLE Scoping

Define your hunt scope using the ABLE framework:

Field	Your Input
Actor *(Optional)*	Unknown financially motivated threat actor - Targets cryptocurrency users and financial services
Behavior	Multi-stage macOS infostealer using bash → osascript → JXA execution chains to bypass Gatekeeper, manipulate TCC privacy permissions, exfiltrate credentials/wallets via goldenticketsshop[.]com C2 infrastructure
Location	macOS endpoints (specifically Apple Silicon M2+ systems), targeting ~/Library/Keychains/, browser credential stores, cryptocurrency wallet

Field	Your Input
	applications
Evidence	**Source:** macOS EDR telemetry (process execution, network connections) **Key Fields:** process_name, command_line, parent_process, network_destination, file_path **Example:** bash executing `curl -fsSL | bash` piped commands, osascript with remote content, `tccutil reset All` commands, connections to goldenticketsshop[.]com **Source:** DNS query logs **Key Fields:** query_domain, query_type, response **Example:** DNS TXT record queries to goldenticketsshop[.]com for dynamic payload retrieval

Threat Intel & Research

- **MITRE ATT&CK Techniques:**
  - `T1566.001 – Spearphishing Attachment` (malicious DMG distribution)
  - `T1204.002 – User Execution: Malicious File` (drag-to-terminal bypass)
  - `T1059.004 – Unix Shell` (bash execution chains)
  - `T1059.002 – AppleScript` (osascript/JXA payloads)
  - `T1027 – Obfuscated Files or Information` (Base64 encoding)
  - `T1497.001 – Virtualization/Sandbox Evasion` (system_profiler VM detection)
  - `T1480.001 – Execution Guardrails` (locale-based targeting)
  - `T1562.001 – Impair Defenses` (TCC privacy permission resets)
  - `T1555 – Credentials from Password Stores` (keychain theft)
  - `T1552.001 – Unsecured Credentials in Files` (wallet harvesting)
  - `T1071.001 – Application Layer Protocol: Web` (C2 API endpoints)
  - `T1071.004 – Application Layer Protocol: DNS` (TXT record payload delivery)
  - `T1565.001 – Stored Data Manipulation` (Ledger Live application tampering)
- **CTI Sources & References:**
  - Jamf Threat Labs: https://www.jamf.com/blog/jtl-digitstealer-macos-infostealer-analysis/
  - Initial discovery reporting on DigitStealer campaign targeting macOS users
  - Typosquatted domain dynamiclake.org (vs legitimate dynamiclake.com)
- **Historical Context:**
  - First hunt for this specific macOS infostealer variant
  - Part of broader macOS threat landscape monitoring

- Response to emerging macOS-targeted financial malware campaigns

OBSERVE: Expected Behaviors

What Normal Looks Like

- **Legitimate TCC resets:** Apple installers using `tccutil reset All com.apple.installer` for system-level installations
- **Enterprise device management:** MDM agents (e.g., Qisda.DDPM) performing TCC resets for policy enforcement
- **Administrative scripts:** IT automation using osascript for legitimate administrative tasks
- **System profiling:** Hardware inventory tools running system_profiler for asset management
- **Keychain operations:** CloudKeychainProxy events for iCloud keychain synchronization

What Suspicious Looks Like

- **Multi-stage execution chains:** bash → curl piping to bash → osascript → JXA in rapid succession from non-administrative users
- **Unauthorized TCC manipulation:** `tccutil reset All` without installer context, especially targeting All permissions
- **Remote script execution:** `curl -fsSL | bash` patterns fetching and executing remote content
- **Hardware enumeration for evasion:** system_profiler querying SPHardwareDataType followed by virtualization checks via sysctl
- **C2 communications:** Connections to goldenticketsshop[.]com, sweetseedsbeep[.]com, nevadabtcshill[.]com, ledgmanyman[.]com
- **DNS TXT record retrieval:** Queries for TXT records to known malicious domains for dynamic payload delivery
- **Credential access patterns:** Keychain database reads (login.keychain-db) outside normal application context
- **Application tampering:** Modifications to Ledger Live app.asar or app.json configuration files

Expected Observables

- **Processes:**
  - bash with curl pipe patterns: `curl -fsSL <URL> | bash`
  - osascript executing remote content or password prompts

    - nohup spawning backgrounded remote payloads
    - tccutil with suspicious reset targets
    - system_profiler with hardware enumeration
- **Network:**
    - Connections to goldenticketsshop[.]com (primary C2)
    - API endpoints: /api/credentials, /api/grabber, /api/log, /api/poll
    - ~10 second polling intervals to C2 infrastructure
    - DNS TXT queries to malicious domains
    - Cloudflare Pages (pages.dev) connections with ASPX endpoints
- **Files:**
    - /tmp/wid.txt (unique ID storage)
    - ~/...txt (password validation cache)
    - ~/Library/LaunchAgents/goldenticketsshop.com.plist (persistence)
    - Modified Ledger Live files: app.asar, app.json
    - DynamicLake.dmg or similar malicious disk images
- **Registry:** N/A (macOS-specific - LaunchAgents instead)
- **Authentication:**
    - Unsolicited password prompts via AppleScript dialogs
    - Privilege escalation attempts

CHECK: Execute & Analyze

Data Source Information

- **Index/Data Source:** macOS EDR database (SQL)
- **Time Range:** Full historical dataset - all available EDR telemetry
- **Events Analyzed:** 134,655 events across 570 specific indicator searches
- **Data Quality:** Good - Comprehensive coverage across entire macOS fleet with process execution, network connections, command-line arguments, and file system monitoring

Hunting Activities

Search 1 - C2 Domain Detection

What We Searched For:

Searched all network connection events from macOS endpoints looking for outbound connections to

the four known DigitStealer command-and-control domains: goldenticketsshop.com, sweetseedsbeep.com, nevadabtcshill.com, and ledgmanyman.com. Grouped results by destination domain, source host, and user to identify any infected systems.

Search Results:

- **Results:** 0 connections detected to any malicious C2 domains
- **Coverage:** 2,533 macOS endpoints searched
- **Assessment:** No active C2 communications observed - strong indicator that fleet is not compromised
- **False Positives:** None - these are confirmed malicious domains

Search 2 - TCC Permission Manipulation

What We Searched For:
Analyzed all process execution events where the `tccutil` utility was invoked with the `reset` command. Categorized TCC resets by scope (full privacy reset vs. specific application permissions) to distinguish between high-risk unauthorized resets and expected Apple installer activity. Examined parent process, user context, and command-line arguments for each reset.

Search Results:

- **Total Events:** 2 instances detected
- **Instance 1:**
  - Host: AHACK-MFV04M972P
  - Command: `tccutil reset All com.apple.installer`
  - Executed by: root via bash
  - Assessment: ✅ Benign - Standard Apple installer operation
- **Instance 2:**
  - User: eotto
  - Command: `tccutil reset ListenEvent Qisda.DDPM`
  - Context: Enterprise device management activity
  - Assessment: ✅ Benign - Authorized enterprise MDM agent

Search Notes:

- Both TCC reset instances have legitimate business context
- No unauthorized privacy permission manipulation detected
- DigitStealer typically uses `tccutil reset All` without installer context - not observed

Search 3 - Multi-Stage Script Execution Chains

What We Searched For:
Identified bash processes executing curl commands piped directly to bash (remote script execution pattern), then correlated these events with osascript executions occurring on the same host within a 5-minute window. This search detects DigitStealer's characteristic multi-stage execution chain: bash → curl pipe → osascript → JXA payload.

Search Results:

- **Bash executions:** 11,625 total events analyzed
- **Curl pipe patterns:** 0 suspicious chains detected
- **Multi-stage chains:** No bash → osascript execution sequences matching DigitStealer behavior
- **Assessment:** No remote script execution patterns consistent with DigitStealer infection

Search 4 - Hardware VM Detection Patterns

What We Searched For:
Searched for system_profiler processes querying hardware information (SPHardwareDataType) and correlated with sysctl commands checking hardware properties within 60 seconds on the same host. This pattern detects DigitStealer's anti-VM detection technique where it enumerates Apple Silicon M2+ processors and checks for virtualization indicators before executing the payload.

Search Results:

- **system_profiler executions:** 65,481 events (routine system information queries)
- **sysctl hardware queries:** Analyzed for Apple Silicon M2+ processor enumeration
- **VM detection patterns:** 0 instances matching DigitStealer evasion techniques
- **Assessment:** No hardware-level virtualization detection consistent with malware anti-analysis

Search 5 - AppleScript/JXA Execution Monitoring

What We Searched For:
Examined all osascript process executions and classified them based on suspicious indicators in command-line arguments: remote content execution (curl), password prompts, Base64-encoded payloads, and keychain access attempts. Filtered out standard executions to focus on potentially malicious AppleScript/JXA activity.

Search Results:

- **Total osascript executions:** 813 events

- **Suspicious patterns:** 0 events with DigitStealer characteristics
  - No password harvesting dialogs
  - No Base64-encoded JXA payloads
  - No keychain access patterns
- **Assessment:** All osascript executions within normal administrative bounds

Refined Search - Keychain Access Monitoring

What We Searched For:

Searched for file access events targeting the macOS keychain database file (login.keychain-db) while excluding legitimate system processes and browsers that normally access keychains (SecurityAgent, secd, securityd, CloudKeychainProxy, Safari, Chrome, Firefox, Brave, Edge). This refined approach focused on unauthorized credential theft attempts.

Refinement Rationale:

- Initial broad search returned 134,655 CloudKeychainProxy events (normal iCloud sync)
- Refined to exclude legitimate keychain access processes
- Focus on unauthorized keychain database access attempts
- **Results:** No unauthorized keychain access detected

Visualization & Analytics

- **Time-series analysis:** Event volume over time showed consistent patterns with no anomalous spikes
- **Host distribution:** Activity distributed evenly across 2,533 macOS endpoints
- **Process execution heatmap:** CloudKeychainProxy (134,655), Keychain Circle Notifications (82,921), sysctl (65,481), bash (11,625), osascript (813)
- **Network connection patterns:** No connections to known malicious infrastructure
- **TCC reset timeline:** 2 instances over historical dataset - both with legitimate context

Search Performance

- **What Worked Well:**
  - C2 domain searches - clear negative result provided high confidence
  - TCC monitoring - low false positive rate with business context attribution
  - Multi-stage execution chain detection - effective at identifying script relationships
  - Hardware enumeration patterns - successfully distinguished malware evasion from legitimate inventory

- **What Didn't Work:**
  - Initial keychain access queries returned excessive legitimate activity (134K+ events)
  - Required refinement to exclude normal system processes
- **Iterations Made:**
  - Iteration 1: Broad keychain access search → 134,655 events
  - Iteration 2: Refined to exclude legitimate processes (SecurityAgent, secd, CloudKeychainProxy, browsers)
  - Iteration 3: Added parent process context for better false positive reduction

KEEP: Findings & Response

Executive Summary

Completed comprehensive threat hunt for DigitStealer macOS infostealer across 2,533 macOS endpoints (100% of fleet), analyzing 134,655 events across 570 specific indicator and behavioral searches. **Hypothesis disproved - no DigitStealer infection detected.** All observed activity aligns with legitimate enterprise operations. The 2 TCC reset instances identified were attributed to authorized Apple installers and enterprise MDM agents. No connections to malicious C2 infrastructure, no multi-stage script execution chains, and no unauthorized credential access patterns were observed. Environment assessed as **low risk** for this threat.

Findings

Finding	Ticket	Description
False Positive	N/A	TCC reset by Apple installer (AHACK-MFV04M972P) - Standard system installation behavior
False Positive	N/A	TCC reset by Qisda.DDPM MDM agent (user: eotto) - Authorized enterprise device management
True Negative	N/A	Zero connections to goldenticketsshop[.]com C2 infrastructure across 2,533 endpoints
True Negative	N/A	No multi-stage bash → osascript → JXA execution chains detected in 11,625 bash events

Finding	Ticket	Description
True Negative	N/A	No unauthorized keychain database access outside legitimate system processes
True Negative	N/A	No hardware VM detection patterns consistent with DigitStealer evasion techniques

True Positives: 0 - No confirmed DigitStealer infections
False Positives: 2 - Both TCC reset events had legitimate business context
Suspicious Events: 0 - No activity requiring further investigation

Detection Logic

Automation Opportunity:

Yes - This hunt logic can become automated detection with the following behavioral patterns:

1. **Unauthorized TCC Privacy Manipulation:** Detect `tccutil reset All` executed outside Apple installer context (without com.apple.installer in command line) → Medium severity, investigate within 4 hours
2. **Multi-Stage Script Execution Chain:** Bash process executing curl piped to bash, followed by osascript execution on same host within 5-minute window → High severity, immediate investigation
3. **Remote Script Execution Pattern:** Bash executing curl with pipe to shell interpreter (`curl -fsSL | bash` patterns) from non-administrative users → Medium severity
4. **Hardware Enumeration for Anti-VM:** system_profiler querying SPHardwareDataType followed by sysctl hardware queries within 60 seconds → Low severity, requires correlation with other suspicious behaviors

Detection Rule Summary:

A production detection rule should focus on **behavioral patterns** rather than specific IOCs:

High-Value Behavioral Detections:

- **TCC Permission Reset Abuse:** Unauthorized privacy database manipulation, excluding known-good Apple installer and MDM agent contexts
- **Multi-Stage Execution Chains:** Time-based correlation of bash → osascript execution sequences indicating staged payload delivery

- **Remote Code Execution via Curl Pipe:** Command patterns fetching and executing remote scripts without user interaction
- **Anti-Analysis Evasion:** Hardware/VM detection followed by credential access or network activity

Tuning Considerations:

- Whitelist Apple system installers performing legitimate TCC resets
- Whitelist authorized MDM agents by process name/path
- Establish baseline for legitimate administrative curl usage patterns
- Correlate multiple low-severity behaviors to elevate priority

Lessons Learned

What Worked Well:

- **Indicator-based hunting:** C2 domain searches provided immediate high-confidence assessment
- **Behavioral pattern detection:** Multi-stage execution chain queries effectively identified script relationships
- **Contextual analysis:** Evaluating TCC resets with parent process and business context reduced false positives
- **Comprehensive coverage:** Searching all 2,533 macOS endpoints provided full fleet visibility
- **Threat intelligence integration:** Using detailed IOCs from Jamf reporting enabled precise hunting

What Could Be Improved:

- **Initial query tuning:** Keychain access queries required significant refinement to reduce legitimate activity noise
- **Behavioral baseline establishment:** Need documented baseline for normal TCC reset frequency, multi-stage script execution patterns, and hardware enumeration activity
- **Process whitelisting:** Should maintain whitelist of authorized MDM agents and administrative scripts for automated filtering
- **Execution chain correlation:** Time-window correlation between bash and osascript could be enhanced with process tree analysis

Telemetry Gaps Identified:

1. **Process tree relationships:** Limited parent-child process lineage visibility for multi-stage execution chain analysis
2. **Application integrity monitoring:** No behavioral detection for application bundle tampering or configuration file manipulation
3. **Keychain access context:** Insufficient telemetry to attribute keychain access to legitimate application workflows vs. unauthorized access
4. **LaunchAgent behavioral monitoring:** Need enhanced detection for persistence mechanisms beyond file creation (e.g., execution frequency, network activity)
5. **Script content visibility:** Limited ability to inspect script content executed via curl pipe or osascript for malicious patterns

Follow-up Actions

- [x] ~~Escalate true positives to incident response~~ - No infections detected
- [ ] Create behavioral detection rules - Develop detections for multi-stage execution chains, TCC manipulation, and remote script execution patterns
- [ ] Establish behavioral baselines - Document normal TCC reset patterns, administrative script execution, and hardware enumeration activity
- [ ] Address telemetry gaps - Enhance process tree visibility and script content inspection capabilities
- [ ] Schedule recurring hunt execution - Quarterly hunt for macOS infostealer behaviors, not campaign-specific
- [ ] Document findings in knowledge base - Share behavioral hunting methodology and detection patterns with SOC
- [ ] Share insights with SOC/IR/TI teams - Brief on macOS multi-stage execution TTPs and behavioral detection opportunities
- [ ] Whitelist legitimate activity - Work with IT to identify and whitelist authorized MDM agents and administrative scripts
- [ ] Enhance persistence monitoring - Implement behavioral detection for LaunchAgent abuse patterns
- [ ] User awareness training - Educate on social engineering tactics: drag-to-terminal execution, Gatekeeper bypasses, and privacy permission manipulation

Follow-up Hunts

- **H-0004:** macOS LaunchAgent persistence hunt - Expand detection to all persistence mechanisms, not just DigitStealer-specific

- **H-0005:** Application bundle tampering detection - Hunt for unauthorized modifications to app.asar, Info.plist, or other application components
- **H-0006:** macOS credential access patterns - Broader hunt for keychain theft, browser credential harvesting, wallet exfiltration across all malware families
- **H-0007:** DNS exfiltration and dynamic payload delivery - Hunt for DNS TXT record abuse, DNS tunneling, and DNS-based C2
- **H-0008:** macOS Gatekeeper bypass techniques - Hunt for drag-to-terminal execution, quarantine attribute removal, and other bypass methods

Hunt Completed: 2025-11-18
Next Review: Quarterly (2026-02-18) or triggered by new DigitStealer campaign intelligence