INTELLIGENCE BRIEFING
Security Command Center
TLP:CLEAR
2026-07-02 07:13 UTC

Phishing Campaigns Fingerprint Devices via User-Agent to Deliver OS-Tailored Payloads

THREAT CAMPAIGN | **HIGH** | CVSS 7.5

SCC Item ID	SCC-CAM-2026-0613
Type	Threat Campaign
Severity	HIGH
CVSS Base Score	7.5
Affected Products	End users across all operating systems; organizations relying on static email filtering or single-payload detection signatures
Published	2026-07-01T16:31:21
Discovery Source	Rss

Executive Summary

Threat actors are reportedly using browser user-agent data to automatically detect a victim's operating system at the moment a phishing link is clicked, then delivering a tailored malicious file for that specific platform. According to Dark Reading, this technique affects end users across Windows, macOS, Linux, and Android, and organizations relying on single-payload detection signatures or single-OS sandbox detonation may not detect the full scope of an attack. Confidence in this report is medium, the technique is technically consistent with observed phishing evolution, but the claim rests on a single aggregated news source without independent confirmation from CISA or a vendor PSIRT.

Technical Analysis

According to Dark Reading (T2 source), phishing campaigns are dynamically serving OS-specific payloads by parsing the HTTP User-Agent string at the point of redirect. Payload selection is reported to follow device fingerprint: .exe for Windows, .dmg or .sh for macOS/Linux, and .apk for Android. The delivery chain exploits browser redirect trust and the limited inspection depth of traditional secure email gateways, which typically detonate links against a single OS image. No CVE is associated with this campaign, it is a technique-level behavior, not a vulnerability in a specific product. Relevant CWEs: CWE-693 (Protection Mechanism Failure) and CWE-290 (Authentication Bypass by Spoofing). MITRE ATT&CK techniques: T1566 (Phishing), T1566.002 (Spearphishing Link), T1204.001 (Malicious Link), T1204.002 (Malicious File), T1071.001 (Web Protocols), T1027.001 and T1027.006 (Obfuscated Files or Information), T1036 (Masquerading), T1598 (Phishing for Information). No patch is applicable; mitigation is detection and control-layer focused. No threat actor attribution is available in the source material. Confidence: MEDIUM, single aggregated news source, no authoritative

advisory corroboration.

Action Checklist

1. Step 1: Containment, Audit email security gateway and proxy configurations to confirm link detonation and sandboxing operates across multiple OS environments (Windows, macOS, Linux, Android), not only a single image. Restrict or quarantine emails containing multi-redirect URL chains from unknown senders pending review. Reference: NIST AC-4 (Information Flow Enforcement) for controlling content traversal paths.
2. Step 2: Detection, Query proxy and email gateway logs for redirect chains where a single shortened or obfuscated URL resolves to different file types depending on the requesting User-Agent string. Flag delivery of .exe, .dmg, .sh, and .apk file types originating from the same parent URL. Monitor for T1566.002 / T1204.001 behavioral patterns in EDR telemetry. Reference: NIST AU-6 (Audit Record Review, Analysis, and Reporting); CIS 8.2 (Collect Audit Logs).
3. Step 3: Eradication, Update YARA and hash-based detection signatures to cover all expected OS-specific payload variants (.exe, .dmg, .sh, .apk) linked from a common campaign URL pattern, not only the first observed file type. Where sandbox detonation is used for URL analysis, configure multi-OS detonation or alert on redirect chains that serve different content types per User-Agent. No vendor patch applies, this is a detection coverage gap, not a product vulnerability.
4. Step 4: Recovery, After signature and sandbox updates, re-scan previously delivered emails containing redirect-chain URLs against updated multi-payload detection logic. Validate that endpoint protection on macOS, Linux, and Android devices is current and active; do not assume Windows-only endpoint coverage is sufficient. Reference: CIS 7.3 (Perform Automated Operating System Patch Management) for endpoint hygiene; NIST SI-4 (Information System Monitoring) for continuous monitoring of endpoint and email security controls post-remediation.
5. Step 5: Post-Incident, Conduct a detection coverage review specifically for non-Windows payloads and multi-redirect phishing delivery. Document gaps in sandbox OS coverage and single-signature detection logic as control deficiencies. Evaluate whether email security architecture satisfies NIST AU-2 (Event Logging) requirements for capturing full redirect chain behavior. Reference: CIS 7.1 (Establish and Maintain a Vulnerability Management Process) for formalizing the gap review; evaluate MFA enforcement (CIS 6.3, Require MFA for Externally-Exposed Applications) as a compensating control to reduce post-click credential theft impact.

IR / Forensic Enrichment

Triage Priority	URGENT
Escalation Criteria	Escalate to CISO and legal/privacy counsel immediately if proxy or endpoint logs confirm payload delivery and execution on any host, particularly if the affected host stores PII, PHI, or payment card data, as multi-OS payload delivery to data-bearing endpoints may trigger breach notification obligations under HIPAA, GDPR, or PCI DSS; additionally escalate if non-Windows hosts (macOS, Linux, Android) show signs of execution, since most organizations lack EDR coverage on these platforms and dwell time may be materially longer than on Windows.

Recovery Notes	After updating YARA signatures and sandbox configurations, re-scan the full inbound email archive for the 30-day window preceding discovery, targeting redirect-chain URLs using User-Agent-aware detonation across all four OS profiles (Windows, macOS, Linux, Android) to identify any delivered-but-undetected payloads. Monitor proxy and DNS logs for 14 days post-remediation for any resumed callbacks to campaign-associated C2 infrastructure, as OS-tailored implants on non-Windows hosts may have established persistence without triggering Windows-centric detection. Verify that endpoint protection agents on macOS, Linux, and Android endpoints have successfully ingested updated signatures and confirm real-time protection is active on each device class before returning any quarantined systems to production.
Forensic Artifacts	Proxy access logs with full User-Agent strings and HTTP response Content-Type headers for each hop in observed redirect chains — the primary artifact confirming OS-fingerprinting branching behavior occurred | Email gateway message headers and raw MIME bodies for delivered phishing emails containing shortened or obfuscated redirect-chain URLs, including X-Mailer, Return-Path, and Received headers for sender infrastructure attribution | Browser or mail client process memory dumps (via ProcDump on Windows, `sudo gcore` on Linux, or `osxpmem` on macOS) from endpoints where a user clicked a campaign link, to recover in-memory HTTP response data and any partially staged payload | File system artifacts in OS-specific download and temp directories — Windows `%USERPROFILE%\Downloads\` and `%TEMP%\`, macOS `~/Downloads/` and `/tmp/`, Linux `~/Downloads/` and `/tmp/`, Android `/sdcard/Download/` — for each OS-specific payload extension (.exe, .dmg, .sh, .apk) with creation timestamps correlated to click events | DNS resolver logs (Pi-hole query log, Windows DNS debug log at `C:\Windows\System32\dns\dns.log`, or `/var/log/named/`) showing resolution history for all shortened URLs in the campaign, including timestamps and requesting client IPs, to reconstruct the full victim click timeline across all OS platforms

Per-Action IR Details

Step 1: Containment — Audit email security gateway and proxy configurations to confirm link detonation and sandboxing operates across multiple OS environments (Windows, macOS, Linux, Android), not only a single image. Restrict or quarantine emails containing multi-redirect URL chains from unknown senders pending review. Reference: NIST AC-4 (Information Flow Enforcement) for controlling content traversal paths.

NIST Phase: Containment

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

Controls: NIST AC-4 (Information Flow Enforcement), CIS 4.4 (Implement and Manage a Firewall on Servers)

Compensating: Without enterprise email gateway sandbox capability, use a pfSense or iptables rule to block outbound HTTP/HTTPS to known URL shortener domains (bit.ly, t.co, rebrand.ly, etc.) at the perimeter. Implement a Pi-hole or local BIND RPZ blocklist for shortened-URL resolver domains. A 2-person team can deploy Pi-hole in under 2 hours and import community blocklists targeting phishing redirect infrastructure. For quarantine, configure MTA-level header filtering (Postfix `header_checks`) to hold any message where the URL body matches a multi-hop redirect pattern (regex: `https?://(bit\.ly|t\.co|rb\.gy|shorturl\.at)/`).

Evidence: Before modifying gateway or proxy rules, export and preserve: (1) current email gateway quarantine queue with full message headers and raw MIME bodies for any held redirect-chain emails; (2) proxy access logs (Squid `/var/log/squid/access.log` or equivalent) covering the past 30 days, capturing full request URI, User-Agent string, referrer, and response content-type for each hop in observed redirect chains; (3) DNS query logs showing resolution history for each shortened or obfuscated URL. These are the only records that will reconstruct which User-Agent strings were served which payload types — gateway rule changes may suppress future logging of the same patterns.

Step 2: Detection — Query proxy and email gateway logs for redirect chains where a single shortened or obfuscated URL resolves to different file types depending on the requesting User-Agent string. Flag delivery

of .exe, .dmg, .sh, and .apk file types originating from the same parent URL. Monitor for T1566.002 / T1204.001 behavioral patterns in EDR telemetry. Reference: NIST AU-6 (Audit Record Review, Analysis, and Reporting); CIS 8.2 (Collect Audit Logs).

NIST Phase: Detection Analysis

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

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

Compensating: Without SIEM, run this Squid proxy log query via bash to surface User-Agent-differentiated content delivery from a common parent URL: `awk '{print $7, $10, $11}' /var/log/squid/access.log | sort | awk 'seen[$1]++ && $3 != prev[$1] {print "MULTI-TYPE:", $1, prev[$1], $3} {prev[$1]=$3}'`. For email gateway logs, grep for `.exe`, `.dmg`, `.sh`, and `.apk` extensions appearing in URLs within message bodies: `grep -Ei '\.(exe|dmg|sh|apk)' /var/log/maillog | grep -i 'shorturl\|bit\.ly\|t\.co\|rb\.gy'`. Deploy the free Sigma rule `proc_creation_win_susp_file_download_via_browser.yml` against Windows Event Log ID 4688 to catch browser child processes writing OS-specific payload extensions to disk.

Evidence: Capture before any EDR-level process termination or network block: (1) proxy log entries showing the full redirect chain — specifically the HTTP 301/302 Location headers at each hop and the final Content-Type and Content-Disposition response headers, which will confirm User-Agent-based branching; (2) browser or mail client process memory (using ProcDump on Windows: `procdump -ma `) if a user has already clicked a link, to recover in-memory HTTP response data including the served payload and User-Agent header sent; (3) Windows Security Event Log Event ID 4688 (Process Creation) filtered on browser or mail client processes spawning child processes that write `.exe` files to `%USERPROFILE%\Downloads` or `%TEMP%`; (4) on macOS, Unified Log entries via `log show --predicate 'process == "Safari" OR process == "Mail"' --info` covering the click timeframe.

Step 3: Eradication — Update YARA and hash-based detection signatures to cover all expected OS-specific payload variants (.exe, .dmg, .sh, .apk) linked from a common campaign URL pattern, not only the first observed file type. Where sandbox detonation is used for URL analysis, configure multi-OS detonation or alert on redirect chains that serve different content types per User-Agent. No vendor patch applies — this is a detection coverage gap, not a product vulnerability.

NIST Phase: Eradication

Reference: NIST 800-61r3 §3.4 — Eradication

Controls: CIS 7.1 (Establish and Maintain a Vulnerability Management Process), CIS 7.2 (Establish and Maintain a Remediation Process)

Compensating: Without a commercial sandbox, use Cape Sandbox (free, open-source) or Any.run community tier to manually submit the campaign URL four times with forged User-Agent strings for each target OS: Windows (`Mozilla/5.0 (Windows NT 10.0; Win64; x64)`), macOS (`Mozilla/5.0 (Macintosh; Intel Mac OS X 13_0)`), Linux (`Mozilla/5.0 (X11; Linux x86_64)`), and Android (`Mozilla/5.0 (Linux; Android 13; Pixel 7)`). Use `curl -A '' -L -o payload_.bin ''` to retrieve each OS-specific payload for local YARA rule development. Write a campaign-specific YARA rule targeting shared string artifacts across all four variants (e.g., C2 domain, mutex name, embedded certificate, or common packer stub).

Evidence: Before deploying updated YARA signatures or altering sandbox configuration (which may purge queued detonation artifacts), preserve: (1) all previously detonated sandbox reports and PCAP files associated with the campaign URL, including HTTP response bodies for each User-Agent variant already observed; (2) file system artifacts on any confirmed-compromised endpoint — on Windows, collect `%USERPROFILE%\Downloads\*`, `%TEMP%\*`, and `%APPDATA%\*` created within the click timeframe using `robocopy` to a write-protected evidence share; on macOS, collect `~/Downloads/` and `/tmp/` with `sudo find / -newer /tmp/reference_time -type f 2>/dev/null`; (3) any existing IDS/IPS alert logs tied to the campaign URL pattern before signature updates overwrite alert context.

Step 4: Recovery — After signature and sandbox updates, re-scan previously delivered emails containing redirect-chain URLs against updated multi-payload detection logic. Validate that endpoint protection on macOS, Linux, and Android devices is current and active; do not assume Windows-only endpoint coverage is sufficient. Reference: CIS 7.3 (Perform Automated Operating System Patch Management) for endpoint

hygiene; NIST SI-4 is not in the provided knowledge base — no mapped control for post-remediation monitoring beyond AU-6.

NIST Phase: Recovery

Reference: NIST 800-61r3 §3.5 — Recovery

Controls: CIS 7.3 (Perform Automated Operating System Patch Management), CIS 7.4 (Perform Automated Application Patch Management), CIS 2.2 (Ensure Authorized Software is Currently Supported)

Compensating: Without centralized MDM or endpoint management, use `osquery` to audit AV/EDR presence and definition currency across Linux and macOS endpoints: `SELECT name, version, last_opened_time FROM apps WHERE name LIKE '%antivirus%' OR name LIKE '%endpoint%';`. For Android devices, use Android Debug Bridge (ADB): `adb shell pm list packages | grep -i security` and `adb shell settings get global package_verifier_enable` to confirm Google Play Protect is active. On Windows, use PowerShell: `Get-MpComputerStatus | Select-Object AMProductVersion, AntivirusSignatureLastUpdated, RealTimeProtectionEnabled` to confirm Defender signature currency. Re-scan quarantined email archive using updated ClamAV signatures: `clamscan -r --detect-pua --alert-encrypted /path/to/mail/quarantine/`.

Evidence: Before re-scanning production mail archives or altering endpoint protection state on potentially compromised non-Windows hosts, capture: (1) macOS — run `sudo log collect --last 7d` to preserve the Unified Log archive before any AV scan modifies file access timestamps; (2) Linux — collect `/var/log/auth.log`, `/var/log/syslog`, and bash history (`~/.bash_history`, `/root/.bash_history`) before running any scan that may alter inode metadata; (3) Android — if a device is suspected compromised, extract Android logcat output via ADB (`adb logcat -d > device_logcat.txt`) and the package install history (`adb shell pm list packages -i -3`) before initiating MDM wipe or AV remediation; (4) preserve the original email archive in read-only state before re-scanning, so re-scan results can be compared forensically against pre-update baseline.

Step 5: Post-Incident — Conduct a detection coverage review specifically for non-Windows payloads and multi-redirect phishing delivery. Document gaps in sandbox OS coverage and single-signature detection logic as control deficiencies. Evaluate whether email security architecture satisfies NIST AU-2 (Event Logging) requirements for capturing full redirect chain behavior. Reference: CIS 7.1 (Establish and Maintain a Vulnerability Management Process) for formalizing the gap review; CIS 6.3 (Require MFA for Externally-Exposed Applications) as a compensating control to reduce post-click credential theft impact.

NIST Phase: Post Incident

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

Controls: NIST AU-2 (Event Logging), CIS 7.1 (Establish and Maintain a Vulnerability Management Process), CIS 6.3 (Require MFA for Externally-Exposed Applications)

Compensating: Without a formal GRC platform, document detection gaps in a structured gap register using a shared spreadsheet with columns: Gap ID, Control Deficiency Description, Affected OS Scope, Detection Method Tested, Result, Remediation Owner, Target Date. For the MFA compensating control, if no commercial IdP is available, deploy Authelia (free, open-source) as a reverse proxy MFA layer in front of externally exposed web applications within scope. Test sandbox OS coverage gaps by resubmitting stored campaign IOCs through updated detection stack and documenting pass/fail per OS variant. Publish a threat-specific Sigma rule to your internal detection library covering the User-Agent-branched redirect chain pattern for future hunt reuse.

Evidence: No live-state alteration occurs in this phase, so volatile capture is not a prerequisite. However, assemble the following for the post-incident report and lessons-learned session: (1) complete proxy log extract showing all User-Agent strings and corresponding Content-Type responses served from campaign URLs, annotated to show which OS variants were and were not detected by pre-incident signatures; (2) email gateway quarantine statistics showing volume of redirect-chain emails received, date range, and which were caught versus delivered, segmented by recipient OS profile if available; (3) sandbox detonation records for each OS variant tested, with verdicts and any missed-detection entries clearly marked; (4) endpoint telemetry summary from non-Windows devices (macOS Unified Log, Linux syslog, Android logcat) confirming whether any payload execution occurred on non-Windows hosts during the campaign window; (5) a detection timeline mapping first IOC observed to first alert generated, to quantify dwell time and coverage gap duration for the lessons-learned record.

Detection Guidance

Focus detection on redirect-chain behavior and multi-payload delivery patterns rather than static file hashes, which will differ per OS. Specific indicators to query: (1) Proxy logs, identify URLs that return different Content-Type headers or file extensions (.exe vs .dmg vs .apk) based on the User-Agent field in the HTTP request from the same originating URL. (2) Email gateway logs, flag messages containing URLs with one or more intermediate redirects before final file delivery, particularly those using URL shorteners or low-reputation domains. (3) EDR telemetry, alert on browser or mail client child process spawning a downloaded file with extensions matching .exe, .dmg, .sh, or .apk within seconds of a link click. (4) Sandbox detonation, if your link detonation environment uses a single OS image, any OS-specific payload variant for other platforms will not be detected; this is a coverage gap, not a clean verdict. MITRE ATT&CK T1027.006 (HTML Smuggling) and T1036 (Masquerading) are also relevant if payloads are obfuscated or renamed to appear benign. No specific IOCs are available in the source material for this campaign. Reference: NIST AU-6, AU-12, CIS 8.2.

Framework Mappings

MITRE-ATTACK

- **T1566** — Phishing
- **T1204.001** — Malicious Link
- **T1071.001** — Web Protocols
- **T1204.002** — Malicious File
- **T1027.006** — HTML Smuggling
- **T1027.001** — Binary Padding
- **T1566.002** — Spearphishing Link
- **T1036** — Masquerading
- **T1598** — Phishing for Information

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

CIS-V8

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

HIPAA-SECURITY

- **164.308(a)(5)(i)** — Security Awareness and Training

ISO-27001-2022

- **A.5.34** — Privacy and protection of personal information
- **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	Phishing	Initial-Access
T1204.001	Malicious Link	Execution
T1071.001	Web Protocols	Command-And-Control
T1204.002	Malicious File	Execution
T1027.006	HTML Smuggling	Defense-Evasion
T1027.001	Binary Padding	Defense-Evasion
T1566.002	Spearphishing Link	Initial-Access
T1036	Masquerading	Defense-Evasion
T1598	Phishing for Information	Reconnaissance

Sources

Source	URL	Tier
Security News	https://www.darkreading.com/application-security/phishing-campaigns...	T2
Vulnerabilities - NVD	https://nvd.nist.gov/vuln	T1
What Is a Security Vulnerability? Definition, Types, and ...	https://www.picussecurity.com/resource/glossary/what-is-a-security-...	T3
ECS: Solution to Address CVE-2022-31231 Security ...	https://www.dell.com/support/kbdoc/en-sg/000200962/ecs-solution-to-...	T3
Software vendor refuses to fix security vulnerability	https://security.stackexchange.com/questions/264626/software-vendor...	T3

DISCLAIMER

Generated 2026-07-02 07:13 UTC by TJS Security Command Center