INTELLIGENCE BRIEFING
Security Command Center
TLP:CLEAR
2026-07-13 06:42 UTC

RedHook Android Malware Weaponizes Wireless ADB for Agentless Shell Access

THREAT CAMPAIGN | **HIGH** | CVSS 7.5

SCC Item ID	SCC-CAM-2026-0657
Type	Threat Campaign
Severity	HIGH
CVSS Base Score	7.5
Affected Products	Android devices with Wireless Debugging (Wireless ADB) enabled
Published	2026-07-12T10:27:32
Discovery Source	Rss

Executive Summary

According to a BleepingComputer report, a malware family tracked as RedHook is abusing Android's Wireless Debugging feature to gain shell-level access on compromised Android devices over TCP/IP, without requiring physical USB access. Any Android device running Android 11 or later with Wireless Debugging enabled is potentially exposed to command execution, data exfiltration, and payload staging. This report is currently single-sourced; no vendor advisory or independent threat intelligence corroboration has been confirmed, which tempers confidence in the attribution and scope claims.

Technical Analysis

According to BleepingComputer (T2 source; no corroborating primary vendor advisory confirmed), a malware family designated RedHook has introduced a capability to abuse Android's Wireless Debugging feature, introduced in Android 11, to establish ADB sessions over TCP/IP on default port 5555 without requiring USB tethering or physical device proximity. Prior ADB-based attack chains depended on either USB connection or local network access with developer options manually enabled; this variant is reported to lower that barrier. Once an ADB session is established, the attacker obtains shell-level privileges enabling command execution (T1059.004), container or debug interface abuse (T1609), data exfiltration via alternative protocols (T1437), location tracking (T1430), and network configuration enumeration (T1422). CWE-284 (Improper Access Control) and CWE-269 (Improper Privilege Management) describe the underlying trust-model weakness in Wireless ADB. No CVE identifier is associated with this campaign in the source data. The reported CVSS base of 7.5 cannot be independently verified without a formal advisory record. Attribution to 'RedHook' as a distinct named threat actor is low-confidence; no threat intelligence vendor profile corroborates the designation. Confidence in the described technical mechanism is medium based on the single aggregated news source.

Action Checklist

1. Step 1: Containment, Audit all managed Android devices (Android 11+) enrolled in MDM and disable Wireless Debugging via policy enforcement immediately. Block inbound and outbound TCP port 5555 at network egress and segment boundaries for any segments hosting mobile devices. Reference: NIST AC-3 (Access Control), AC-6 (Least Privilege), and AC-17 (Remote Access), establish and enforce restrictions on remote debugging interfaces and network access to mobile device management ports.
2. Step 2: Detection, Query MDM and endpoint telemetry logs for active ADB-over-network sessions (TCP port 5555 connections originating from or destined to Android devices). Review network flow data for unexpected outbound connections on port 5555. Check for developer option enablement events in MDM audit logs. Reference: NIST AU-6 (Audit Record Review, Analysis, and Reporting); CIS 8.2 (Collect Audit Logs). D3FEND countermeasure: D3-LAM (Local Account Monitoring), monitor for unauthorized account or session activity on managed devices.
3. Step 3: Eradication, Enforce MDM policy to disable 'Wireless Debugging' and 'Developer Options' across the Android device fleet. Where MDM enforcement is unavailable, require manual disablement and confirm via device compliance check. No vendor patch is available or required, this abuses a legitimate OS feature. Reference: NIST CM controls (configuration management); CIS 4.6 (Securely Manage Enterprise Assets and Software).
4. Step 4: Recovery, Validate that Wireless Debugging is disabled across all managed devices via MDM compliance reporting. Confirm TCP port 5555 is blocked at all relevant network boundaries and that no active ADB sessions appear in network flow logs. Monitor for any re-enablement of developer options on enrolled devices. Reference: NIST SI-4 (System Monitoring); CIS 7.1 (Establish and Maintain a Vulnerability Management Process).
5. Step 5: Post-Incident, Review mobile device management policies to enforce baseline developer-option restrictions for all corporate-managed Android devices. Assess whether mobile endpoints are segregated from sensitive internal network segments. Add ADB port 5555 monitoring as a persistent detection rule in your SIEM or NDR platform. Reference: NIST AC-6 (Least Privilege); CIS 4.4 (Implement and Manage a Firewall on Servers) applied to network segments hosting mobile devices. D3FEND countermeasure: D3-UAP (User Account Permissions), restrict privileged interface access on managed endpoints.

IR / Forensic Enrichment

Triage Priority	URGENT
Escalation Criteria	Escalate immediately to CISO and legal/privacy counsel if network flow analysis confirms data exfiltration (large outbound byte volumes on TCP/5555 from devices holding PII, PHI, or credentials), if more than 10% of the managed Android fleet shows active or historical TCP/5555 sessions, or if MDM audit logs cannot account for developer-option enablement events — any of which triggers potential breach notification obligations under HIPAA, GDPR, or applicable state data protection law.

Recovery Notes	Before returning any Android device to production use, verify via MDM compliance dashboard that 'Developer Options' and 'Wireless Debugging' report as disabled and that the device's '/data/misc/adb/adb_keys' file contains no unrecognized RSA public keys that would allow a retained RedHook ADB authorization to re-establish sessions after network controls are relaxed. Maintain TCP/5555 blocking at all segment boundaries as a permanent control, not a temporary incident measure, given that Wireless ADB abuse requires no exploit and can be re-triggered any time a user or attacker re-enables the feature. Monitor MDM compliance reports daily for developer-option re-enablement alerts for a minimum of 30 days post-incident, as RedHook or copycat activity may attempt to reestablish access on devices where the ADB key was retained before eradication was confirmed.
Forensic Artifacts	ADB authorized keys file at '/data/misc/adb/adb_keys' on each Android device — contains RSA public keys of all clients that have been granted ADB shell access; presence of unrecognized keys is the primary persistence artifact of a RedHook ADB pairing event | Network flow records (NetFlow, IPFIX, or pcap) filtered on TCP port 5555 for the mobile device IP subnet — session metadata (five-tuple, byte count, duration) identifies which devices received ADB connections, from which external IPs, and whether payload staging occurred based on outbound byte volume | MDM audit log export filtered for 'developer options', 'wireless debugging', and 'USB debugging' state-change events — timestamps reveal when the attack surface was opened on each device and whether enablement was user-initiated or occurred outside business hours suggesting attacker action | Android logcat output ('adb logcat -d -b all') from suspect devices, specifically filtering on process tags 'adbd', 'shell', 'am', and 'pm' — these entries record ADB shell commands executed, Android package manager installs, and activity manager launches that would indicate RedHook payload staging or persistence mechanism installation | Android package manager installed application list ('adb shell pm list packages -f -3') captured pre-eradication — RedHook payload staging via ADB shell would produce third-party APK installations not present in MDM app inventory, and the '-f' flag reveals the APK file path which may indicate sideloaded malware staging directories

Per-Action IR Details

Step 1: Containment — Audit all managed Android devices (Android 11+) enrolled in MDM and disable Wireless Debugging via policy enforcement immediately. Block inbound and outbound TCP port 5555 at network egress and segment boundaries for any segments hosting mobile devices. Reference: NIST AC-17 (Remote Access) — establish and enforce restrictions on remote access connections.

NIST Phase: Containment

Reference: NIST 800-61r3 §3.3 — Containment Strategy: isolate affected systems and prevent further attacker access by cutting the Wireless ADB TCP channel before volatile evidence is destroyed

Controls: NIST AC-17 (Remote Access) — governs establishment and enforcement of restrictions on remote access connections, directly applicable to blocking unauthorized ADB-over-network sessions on TCP/5555, NIST AC-4 (Information Flow Enforcement) — governs network-layer blocking of TCP port 5555 at egress and segment boundaries to prevent ADB command flow to and from Android devices, CIS 4.4 (Implement and Manage a Firewall on Servers) — apply firewall rules at network segment boundaries hosting mobile devices to deny TCP/5555 inbound and outbound

Compensating: On network perimeter without enterprise firewall management: run 'iptables -I FORWARD -p tcp --dport 5555 -j DROP && iptables -I FORWARD -p tcp --sport 5555 -j DROP' on the segment gateway or SOHO router ACL. Audit MDM-enrolled device compliance via your MDM console's device list filtered on Android 11+ and flag any showing 'Developer Options: Enabled'. For teams using Android Debug Bridge locally, run 'adb devices' from a trusted workstation on the suspect network segment to enumerate any devices currently advertising ADB-over-network.

Evidence: BEFORE disabling Wireless Debugging via MDM policy or blocking TCP/5555, capture: (1) active ADB-over-network session state — run 'adb devices' from a monitor host on the affected segment to document all responding device IPs and ports; (2) full network flow records (NetFlow/IPFIX or pcap via Wireshark/tcpdump on 'tcp port 5555') showing source/destination IPs, byte counts, and session start times for all TCP/5555 connections from Android device IP ranges; (3) MDM compliance snapshot showing current 'Developer Options' and 'Wireless Debugging' state per device before policy push alters reported state. These artifacts establish which devices were accessible via ADB and whether any active shell sessions were in progress at time of containment.

Step 2: Detection — Query MDM and endpoint telemetry logs for active ADB-over-network sessions (TCP port 5555 connections originating from or destined to Android devices). Review network flow data for unexpected outbound connections on port 5555. Check for developer option enablement events in MDM audit logs. Reference: NIST AU-6 (Audit Record Review, Analysis, and Reporting); CIS 8.2 (Collect Audit Logs). D3FEND countermeasure: D3-LAM (Local Account Monitoring) — monitor for unauthorized account or session activity on managed devices.

NIST Phase: Detection Analysis

Reference: NIST 800-61r3 §3.2 — Detection and Analysis: correlate MDM audit events, network flow data, and ADB session telemetry to identify devices that received unauthorized shell access via Wireless ADB and determine scope of RedHook activity

Controls: NIST AU-6 (Audit Record Review, Analysis, and Reporting) — governs review of MDM audit logs and network flow records for Wireless ADB session indicators and developer option enablement events, NIST AU-2 (Event Logging) — governs identification and logging of ADB-over-network connection events and MDM policy state changes on Android 11+ devices, CIS 8.2 (Collect Audit Logs) — governs ensuring MDM and network infrastructure logging is enabled and capturing TCP/5555 connection telemetry across segments hosting mobile devices

Compensating: Without a SIEM: run 'tcpdump -i tcp port 5555 -w adb_capture.pcap' on the network segment gateway for 30 minutes and open in Wireshark, filtering 'tcp.port == 5555' to enumerate unique device IPs. Query MDM audit logs (export CSV from console) and grep or Excel-filter for events containing 'developer options', 'wireless debugging', or 'USB debugging' state changes within the last 30 days. For devices reachable on-network, run 'adb connect :5555' from a controlled host — a successful connection without a pairing prompt confirms Wireless ADB is open and unauthenticated, which is the exact condition RedHook exploits.

Evidence: This is a detection/analysis step that does not alter live state, so evidence capture is concurrent rather than prerequisite. Preserve: (1) MDM audit log export filtered for 'developer options enabled', 'wireless debugging enabled', and compliance policy override events per device, with timestamps; (2) NetFlow or pcap showing all TCP/5555 sessions — record five-tuple (src IP, src port, dst IP, dst port, protocol), session duration, and byte volume, as RedHook payload staging would show anomalously large outbound byte counts; (3) ADB pairing log if accessible on device at '/data/misc/adb/adb_keys' — presence of unrecognized RSA public keys indicates a prior ADB authorization that may correspond to RedHook infrastructure; (4) Android logcat output ('adb logcat -d > logcat.txt') if device is accessible, filtering for 'adbd', 'shell', and 'am start' entries that would indicate RedHook executing shell commands or staging payloads.

Step 3: Eradication — Enforce MDM policy to disable 'Wireless Debugging' and 'Developer Options' across the Android device fleet. Where MDM enforcement is unavailable, require manual disablement and confirm via device compliance check. No vendor patch is available or required — this abuses a legitimate OS feature. Reference: NIST CM controls (configuration management); CIS 4.6 (Securely Manage Enterprise Assets and Software).

NIST Phase: Eradication

Reference: NIST 800-61r3 §3.4 — Eradication: eliminate the RedHook attack surface by removing the Wireless ADB exposure on all Android 11+ devices and purging any unauthorized ADB authorization keys that may persist post-containment

Controls: NIST AC-3 (Access Enforcement) — governs enforcement of approved authorizations; disabling Developer Options and Wireless Debugging removes the unauthenticated shell access path RedHook abuses, NIST AC-6 (Least Privilege) — governs restricting device capabilities to only those required for business function; Wireless Debugging is

a developer capability that should not be present on production-managed Android endpoints, CIS 4.6 (Securely Manage Enterprise Assets and Software) — governs configuration management of enterprise assets; enforcing MDM policy to disable developer interfaces is a direct implementation of this safeguard, CIS 4.7 (Manage Default Accounts on Enterprise Assets and Software) — governs disabling or restricting pre-configured interfaces; Wireless ADB is an OS-native interface that must be explicitly locked down on managed devices

Compensating: For devices outside MDM: provide end-users with a written procedure — Settings → Developer Options → toggle 'Wireless Debugging' OFF, then Settings → Developer Options → toggle the master 'Developer Options' switch OFF. For verification without MDM compliance reporting, run 'adb connect :5555' from a controlled host after manual disablement — a 'failed to connect' or 'connection refused' result confirms TCP/5555 is no longer listening. Additionally, instruct users to navigate to Settings → Developer Options → 'Wireless debugging' → 'Paired devices' and revoke all listed pairings to purge any ADB RSA keys RedHook may have registered.

Evidence: BEFORE pushing MDM eradication policy (which overwrites current device configuration state): capture (1) a full MDM compliance report snapshot listing current Developer Options and Wireless Debugging state per device — MDM policy enforcement will overwrite these reported values; (2) on any device suspected of active compromise, pull '/data/misc/adb/adb_keys' to preserve all authorized ADB public keys, as these are the durable artifact of a prior RedHook pairing and will be removed or overwritten during eradication; (3) logcat dump ('adb logcat -d -b all > logcat_pre_eradication.txt') capturing any queued shell command history from the 'adbd' process before the daemon is terminated by disabling the feature. Phase sequencing note: eradication proceeds only after containment (TCP/5555 blocked, sessions terminated) per NIST 800-61r3 §3.4.

Step 4: Recovery — Validate that Wireless Debugging is disabled across all managed devices via MDM compliance reporting. Confirm TCP port 5555 is blocked at all relevant network boundaries and that no active ADB sessions appear in network flow logs. Monitor for any re-enablement of developer options on enrolled devices. Reference: NIST SI-4 (System Monitoring); CIS 7.1 (Establish and Maintain a Vulnerability Management Process).

NIST Phase: Recovery

Reference: NIST 800-61r3 §3.5 — Recovery: verify that the Wireless ADB attack surface is fully eliminated across the Android fleet and confirm no residual RedHook sessions or re-enabled developer interfaces exist before returning devices to normal operation

Controls: NIST AU-6 (Audit Record Review, Analysis, and Reporting) — governs ongoing review of MDM compliance reports and network flow logs to confirm TCP/5555 is absent from current session data and developer options remain disabled, CIS 7.1 (Establish and Maintain a Vulnerability Management Process) — governs the documented process for validating remediation completeness; confirms Wireless Debugging policy enforcement is verified, not assumed, across the Android 11+ fleet

Compensating: Without MDM compliance reporting: build a manual verification checklist — for each enrolled Android 11+ device, run 'adb connect :5555' from a controlled host and confirm 'connection refused'; simultaneously capture 'netstat -an | grep 5555' output from the network gateway to confirm no established sessions. Schedule a weekly 15-minute network scan using nmap ('nmap -p 5555 --open') to detect any device where Wireless Debugging has been re-enabled, and log results to a shared spreadsheet as a lightweight compliance record.

Evidence: Recovery validation is a verification step and does not alter live state, so the primary evidence burden is documentation of confirmed-clean state: (1) MDM compliance report export showing 100% of Android 11+ enrolled devices reporting 'Developer Options: Disabled' or 'Wireless Debugging: Disabled', timestamped post-eradication; (2) network flow records or nmap scan output confirming zero devices responding on TCP/5555 across all mobile device segments; (3) confirmation that '/data/misc/adb/adb_keys' on any previously compromised device has been cleared or contains only known, authorized keys — absence of unrecognized RSA public keys confirms RedHook's persistence mechanism (retained ADB authorization) has been eradicated.

Step 5: Post-Incident — Review mobile device management policies to enforce baseline developer-option restrictions for all corporate-managed Android devices. Assess whether mobile endpoints are segregated from sensitive internal network segments. Add ADB port 5555 monitoring as a persistent detection rule in your SIEM or NDR platform. Reference: NIST AC-6 (Least Privilege); CIS 4.4 (Implement and Manage a Firewall

on Servers) applied to network segments hosting mobile devices. D3FEND countermeasure: D3-UAP (User Account Permissions) — restrict privileged interface access on managed endpoints.

NIST Phase: Post Incident

Reference: NIST 800-61r3 §4 — Post-Incident Activity: conduct lessons-learned review to harden Android MDM baseline, improve network segmentation for mobile devices, and establish persistent detection coverage for Wireless ADB abuse as a recurring threat vector

Controls: NIST AC-6 (Least Privilege) — governs restricting Android device capabilities to only business-required functions; Developer Options and Wireless Debugging must be prohibited by MDM baseline policy on all non-developer-role devices, NIST AC-4 (Information Flow Enforcement) — governs persistent network-layer controls restricting TCP/5555 flows between mobile device segments and internal network zones as a durable architectural control, NIST AU-2 (Event Logging) — governs addition of TCP/5555 connection events and MDM developer-option enablement alerts as permanent logged event types in SIEM or NDR platform, CIS 4.4 (Implement and Manage a Firewall on Servers) — governs firewall rule management at segment boundaries; institutionalizing the TCP/5555 block as a permanent ACL entry on all segments hosting Android devices, CIS 4.6 (Securely Manage Enterprise Assets and Software) — governs MDM policy baseline updates to explicitly prohibit Wireless Debugging and Developer Options on all corporate Android 11+ endpoints as a standing configuration requirement

Compensating: Without a commercial SIEM: create a persistent Sigma rule targeting TCP/5555 connections from mobile device IP ranges and deploy it via the free Sigma rule converter to Elasticsearch or Graylog if available, or as a scheduled nmap cron job ('0 6 * * 1 nmap -p 5555 --open >> /var/log/adb_scan.log 2>&1') reviewed weekly. Document MDM policy changes in a version-controlled configuration file (Git repository) with the baseline prohibition on Developer Options recorded as a named policy artifact, satisfying CIS 4.6 without enterprise tooling.

Evidence: Post-incident documentation to retain for lessons-learned and future detection: (1) full timeline of MDM audit events showing when Wireless Debugging was enabled per device and by whom, to identify whether this was user-initiated or attacker-facilitated; (2) all network flow records capturing TCP/5555 session history for the incident window, retained per AU-11 (Audit Record Retention) requirements, as these establish attacker dwell time and data transfer volume; (3) final MDM compliance report confirming policy enforcement baseline, to serve as the documented pre-incident baseline comparison point for any future RedHook or ADB-abuse recurrence; (4) the '/data/misc/adb/adb_keys' contents from all confirmed-compromised devices, preserved as IOC material — the RSA public keys identify the attacker's ADB client and may appear in future intrusions or be shareable as threat intelligence.

Detection Guidance

Monitor network flow data for TCP connections on port 5555 originating from or directed to Android devices on managed network segments, this is the default Wireless ADB port. In your MDM platform, alert on any device where Developer Options or Wireless Debugging is enabled outside an approved exception list. In your SIEM, create a detection rule correlating port 5555 activity with known Android device MACs or IP ranges assigned to the mobile device subnet. Behavioral indicators include unexpected outbound data transfers from Android devices following ADB session establishment, and new process execution events consistent with shell command invocation on managed endpoints. No confirmed IOCs (hashes, IPs, domains) were present in the source data for this campaign. Reference D3FEND technique D3-LAM (Local Account Monitoring) for session-level detection on managed devices.

Framework Mappings

MITRE-ATTACK

- **T1059.004** — Unix Shell
- **T1609** — Container Administration Command

- **T1203** — Exploitation for Client Execution
- **T1437** — Application Layer Protocol
- **T1078** — Valid Accounts
- **T1430** — Location Tracking
- **T1422** — System Network Configuration Discovery
- **T1475**

NIST-800-53R5

- **CM-7** — Least Functionality
- **SI-3** — Malicious Code Protection
- **SI-4** — System Monitoring
- **SC-7** — Boundary Protection
- **SI-2** — Flaw Remediation
- **AC-2** — Account Management
- **AC-6** — Least Privilege
- **IA-2** — Identification and Authentication (Organizational Users)
- **IA-5** — Authenticator Management
- **AC-3** — Access Enforcement

OWASP-TOP10-2021

- **A01:2021** — Broken Access Control

CIS-V8

- **5.4** — Restrict Administrator Privileges to Dedicated Administrator Accounts
- **6.8** — Define and Maintain Role-Based Access Control
- **6.1** — Establish an Access Granting Process
- **6.2** — Establish an Access Revoking Process

SOC2-TSC

- **CC6.1** — The entity implements logical access security software, infrastructure, and architectures over protected information assets
- **CC7.4** — Responds to identified security incidents
- **CC9.2** — Manages risks associated with vendors and business partners

HIPAA-SECURITY

- **164.312(a)(1)** — Access Control
- **164.308(a)(6)(ii)** — Response and Reporting

ISO-27001-2022

- **A.5.34** — Privacy and protection of personal information
- **A.5.21** — Managing information security in the ICT supply chain

MITRE ATT&CK Mapping

Technique ID	Technique Name	Tactic
T1059.004	Unix Shell	Execution
T1609	Container Administration Command	Execution
T1203	Exploitation for Client Execution	Execution
T1437	Application Layer Protocol	Command-And-Control
T1078	Valid Accounts	Defense-Evasion
T1430	Location Tracking	Collection
T1422	System Network Configuration Discovery	Discovery
T1475		

Sources

Source	URL	Tier
Security News	https://www.bleepingcomputer.com/news/security/redhook-android-malw...	T2
Android Wireless Debugging (ADB) Appears to Bypass ...	https://security.stackexchange.com/questions/286781/android-wireles...	T3
Android ADB vulnerability CVE-2026-0073: Find impacted ...	https://www.runzero.com/blog/android-debug-bridge/	T3
USB or Wireless Debugging Detected – What Should I Do?	https://ask.gov.sg/singpass/questions/cm8idm20i0057wd6ud5h2kq4a	T1

Generated 2026-07-13 06:42 UTC by TJS Security Command Center