INTELLIGENCE BRIEFING
Security Command Center
TLP:CLEAR
2026-05-09 18:46 UTC

Mirai-Based xlabs_v1 Botnet Actively Exploiting Exposed Android Debug Bridge (ADB) Interfaces

THREAT CAMPAIGN | **HIGH** | CVSS 8.8

SCC Item ID	SCC-CAM-2026-0296
Type	Threat Campaign
Severity	HIGH
CVSS Base Score	8.8
Affected Products	IoT devices with Android Debug Bridge (ADB) port (TCP/5555) exposed to the internet, including Android TVs, routers, and embedded Android-based devices
Published	2026-05-08
Discovery Source	Gemini

Executive Summary

A Mirai-based botnet called xlabs_v1 is actively scanning the internet for Android devices with their debug port exposed, then hijacking those devices for distributed denial-of-service attacks. Affected devices include Android TV boxes, consumer routers, and embedded Android hardware where a factory default or administrative oversight left TCP port 5555 open to the public internet. Organizations operating Android-based infrastructure or consumer-grade IoT at scale face the risk of their devices becoming unwilling participants in DDoS campaigns, with associated bandwidth consumption, potential service disruption, and reputational exposure.

Technical Analysis

xlabs_v1 is a Mirai-variant botnet exploiting Android Debug Bridge (ADB) interfaces exposed on TCP/5555. ADB is an Android-native debugging protocol that permits unauthenticated shell command execution when internet-accessible. This is a misconfiguration condition, not a software vulnerability; no CVE has been assigned. Relevant CWEs: CWE-306 (Missing Authentication for Critical Function), CWE-284 (Improper Access Control), CWE-16 (Configuration). Exploitation follows the standard Mirai kill chain: mass scanning for port 5555 (T1190), unauthenticated ADB shell access (T1059.004), ingress tool transfer of downloader payload (T1105), C2 establishment over non-standard port (T1571), and DDoS execution (T1498). Infrastructure acquisition is consistent with T1583.005. Reported targets include Android TV devices and consumer routers. Primary DDoS targeting has been reported against Minecraft game servers. No patch is applicable; remediation is configuration-based. Sources: The Hacker News, Security Affairs, GBHackers.

Action Checklist

1. Containment, Immediately audit firewall and perimeter rules to block inbound TCP/5555 from any external source. If network devices or Android-based systems must be remotely managed, restrict ADB access to a specific internal management VLAN or jump host. Identify all internet-facing Android TV, router, and embedded Android assets in your environment.
2. Detection, Query firewall and NetFlow logs for inbound or outbound TCP/5555 connections. Alert on unexpected outbound traffic volumes from IoT or Android-based devices, which may indicate DDoS participation. Check endpoint or EDR telemetry for ADB daemon (adbd) process activity on non-development systems. Look for shell commands spawned from ADB sessions in device logs if accessible.
3. Eradication, Disable ADB on all production and consumer-grade Android devices where debugging is not operationally required. On Android devices, ADB can be disabled via Developer Options (Settings > Developer Options > USB Debugging off) or by issuing 'adb disconnect' and removing persistent ADB enable flags. Re-image or factory-reset any device confirmed to have been accessed via ADB without authorization.
4. Recovery, After disabling ADB and closing port 5555 at the perimeter, verify closure with an external port scan or firewall rule audit. Monitor previously affected devices for anomalous outbound traffic for a minimum of 72 hours. Confirm no unauthorized scheduled tasks, cron jobs, or persistence mechanisms were installed during ADB access.
5. Post-Incident, Conduct an asset inventory review specifically for Android-based and IoT devices; this campaign highlights that non-traditional endpoints often lack standard hardening baselines. Implement a configuration management policy requiring ADB to be disabled by default on all non-development Android deployments. Map control gaps to CIS Benchmarks for Android devices and NIST SP 800-53 CM-6 (Configuration Settings) and IA-3 (Device Identification and Authentication).

IR / Forensic Enrichment

Triage Priority	IMMEDIATE
Escalation Criteria	Escalate to senior IR leadership and legal/compliance if NetFlow or pcap evidence confirms any compromised Android device successfully participated in xlabs_v1 DDoS operations against external targets (potential liability as an unwitting attack source), if more than 10 devices are confirmed compromised indicating systemic inventory or hardening failure, or if any compromised device was co-located on a network segment housing PII, PHI, or payment data triggering applicable breach notification assessment.
Recovery Notes	After factory-resetting confirmed xlabs_v1-compromised devices, do not return them to internet-facing operation until an external port scan from an out-of-band host confirms TCP/5555 is closed and the device firmware build fingerprint matches the vendor-published factory image. Monitor all previously affected and adjacent Android/IoT devices for anomalous sustained outbound UDP or TCP SYN flood patterns for a minimum of 72 hours post-recovery, as Mirai variants occasionally install secondary persistence loaders that survive factory reset on devices with writable firmware partitions. Validate that the perimeter firewall deny rule for TCP/5555 is persistent across device reboots and firewall config saves, as rule loss after maintenance windows is a documented re-exposure vector for this campaign.

Forensic Artifacts	/data/local/tmp/ directory contents on compromised Android devices — standard staging path for xlabs_v1 Mirai dropper binaries delivered via unauthenticated ADB shell; collect file listing with timestamps and MD5/SHA256 hashes before factory reset | Android logcat output ('adb logcat -d') capturing shell command execution traces from the unauthorized ADB session, including wget/curl/busybox download commands used to pull the Mirai ELF binary and any 'am start' or 'pm install' commands indicating APK-based persistence attempts | NetFlow or pcap records of sustained high-volume outbound UDP or TCP SYN traffic from Android/IoT device IPs to random external destinations — the primary behavioral indicator that a device was actively executing xlabs_v1 DDoS flood modules | Firewall and perimeter syslog records of inbound TCP/5555 SYN packets from external source IPs — correlate source IPs against known Mirai botnet scanner infrastructure and xlabs_v1 campaign IOCs to establish whether scanning preceded confirmed exploitation | 'adb shell getprop' full output from compromised devices, specifically ro.debuggable, persist.service.adb.enable, and ro.build.fingerprint — these properties reveal whether ADB was intentionally left enabled via a persistent system property set by the factory image or modified post-compromise by the Mirai payload

Per-Action IR Details

Containment — Immediately audit firewall and perimeter rules to block inbound TCP/5555 from any external source. If network devices or Android-based systems must be remotely managed, restrict ADB access to a specific internal management VLAN or jump host. Identify all internet-facing Android TV, router, and embedded Android assets in your environment.

NIST Phase: Containment

Reference: NIST 800-61r3 §3.3 — Containment Strategy: isolate affected systems and block attack vector to prevent further exploitation or botnet enrollment

Controls: NIST IR-4 (Incident Handling), NIST SC-7 (Boundary Protection), NIST CM-6 (Configuration Settings), CIS 4.4 (Implement and Manage a Firewall on Servers), CIS 4.5 (Implement and Manage a Firewall on End-User Devices), CIS 1.1 (Establish and Maintain Detailed Enterprise Asset Inventory)

Compensating: Run 'nmap -p 5555 --open ' from an external vantage point (use a cloud VM or Shodan API free tier with query 'port:5555 product:Android') to enumerate exposed ADB surfaces before firewall rules are confirmed closed. On a Linux firewall/router, apply 'iptables -I INPUT -p tcp --dport 5555 -j DROP' immediately. For asset discovery on internal segments, run 'nmap -p 5555 --open 192.168.0.0/16' to identify any Android devices with ADB listening internally.

Evidence: Before modifying firewall rules, export and preserve current perimeter ACL/ruleset snapshots and NetFlow records showing historical inbound TCP/5555 connection attempts — these establish the exposure window and whether xlabs_v1 scanner IPs reached any internal device. Capture Shodan or Censys historical exposure data for your IP ranges to document how long TCP/5555 was publicly reachable. Preserve router/firewall syslog entries showing source IPs of TCP/5555 inbound SYN packets, which may match known xlabs_v1 scanner infrastructure.

Detection — Query firewall and NetFlow logs for inbound or outbound TCP/5555 connections. Alert on unexpected outbound traffic volumes from IoT or Android-based devices, which may indicate DDoS participation. Check endpoint or EDR telemetry for ADB daemon (adbd) process activity on non-development systems. Look for shell commands spawned from ADB sessions in device logs if accessible.

NIST Phase: Detection Analysis

Reference: NIST 800-61r3 §3.2 — Detection and Analysis: correlate network and host indicators to determine whether xlabs_v1 successfully enrolled devices into its botnet and assess DDoS participation scope

Controls: NIST SI-4 (System Monitoring), NIST AU-6 (Audit Record Review, Analysis, and Reporting), NIST AU-12 (Audit Record Generation), NIST IR-5 (Incident Monitoring), CIS 8.2 (Collect Audit Logs)

Compensating: On Linux-based network equipment, run 'tcpdump -nn -i tcp port 5555' to capture live ADB traffic. Query NetFlow with 'nfdump -r /var/cache/nfdump -s ip/bytes -n 20 "proto tcp and port 5555"' to rank top talkers on

TCP/5555. On the Android device itself (if accessible via local USB ADB), run 'adb shell ps | grep adbd' to confirm daemon state and 'adb shell netstat -anp | grep 5555' to identify active connections. Use Wireshark with display filter 'tcp.port == 5555' on a network tap or SPAN port to capture and inspect ADB protocol handshakes and shell command streams indicative of Mirai dropper staging.

Evidence: Preserve NetFlow or firewall session logs showing outbound high-volume UDP flood traffic from Android-based devices (Mirai DDoS typically uses UDP, TCP SYN, or HTTP floods — xlabs_v1 variants exhibit sustained outbound UDP bursts to random destinations). Capture 'adb shell getprop' output if device access is available — Mirai-based payloads on Android ADB often modify ro.debuggable and persist via /data/local/tmp/ dropper binaries with randomized names. Export Android device logcat output ('adb logcat -d > device_logcat.txt') before any remediation, as it may contain shell execution traces from the unauthorized ADB session including wget/curl commands used to pull the xlabs_v1 Mirai binary.

Eradication — Disable ADB on all production and consumer-grade Android devices where debugging is not operationally required. On Android devices, ADB can be disabled via Developer Options (Settings > Developer Options > USB Debugging off) or by issuing 'adb disconnect' and removing persistent ADB enable flags. Re-image or factory-reset any device confirmed to have been accessed via ADB without authorization.

NIST Phase: Eradication

Reference: NIST 800-61r3 §3.4 — Eradication: remove the Mirai-based xlabs_v1 implant and the ADB attack surface that enabled unauthorized access; factory reset is preferred over manual cleanup for consumer-grade Android devices where forensic completeness cannot be guaranteed

Controls: NIST IR-4 (Incident Handling), NIST SI-2 (Flaw Remediation), NIST SI-3 (Malicious Code Protection), NIST CM-6 (Configuration Settings), CIS 2.3 (Address Unauthorized Software), CIS 4.7 (Manage Default Accounts on Enterprise Assets and Software)

Compensating: Before factory reset, use 'adb shell find /data/local/tmp -type f -newer /data/local/tmp -ls' to enumerate recently dropped Mirai binaries staged in the world-writable /data/local/tmp directory (standard xlabs_v1 and Mirai-Android dropper staging path). Run 'adb shell ls -la /data/local/tmp/' and hash any unknown binaries with 'adb shell md5sum /data/local/tmp/' then check hashes against VirusTotal offline via ClamAV signatures updated locally. After factory reset, confirm ADB is off by running 'adb devices' from a connected workstation — the device should return 'unauthorized' or not appear. Disable Developer Options by revoking the setting in Android Settings or, on rooted/manageable devices, via 'adb shell settings put global development_settings_enabled 0' prior to reset for devices remaining in service.

Evidence: Before wiping, collect: (1) full list of running processes via 'adb shell ps -A > processes.txt'; (2) contents of /data/local/tmp/ including binary hashes; (3) 'adb shell netstat -anp' output to document active C2 or DDoS outbound connections; (4) 'adb shell cat /proc/net/tcp' and '/proc/net/udp' for raw socket state; (5) logcat dump as noted in Detection. These artifacts establish whether the device was actively participating in xlabs_v1 DDoS operations and whether the Mirai dropper established persistence beyond /data/local/tmp/ (e.g., via init.d scripts on rooted devices or crontab entries on Android-x86 embedded systems).

Recovery — After disabling ADB and closing port 5555 at the perimeter, verify closure with an external port scan or firewall rule audit. Monitor previously affected devices for anomalous outbound traffic for a minimum of 72 hours. Confirm no unauthorized scheduled tasks, cron jobs, or persistence mechanisms were installed during ADB access.

NIST Phase: Recovery

Reference: NIST 800-61r3 §3.5 — Recovery: restore devices to a known-good state and validate that xlabs_v1 persistence mechanisms have been fully removed before returning devices to production

Controls: NIST IR-4 (Incident Handling), NIST SI-7 (Software, Firmware, and Information Integrity), NIST AU-6 (Audit Record Review, Analysis, and Reporting), NIST CM-6 (Configuration Settings), CIS 7.1 (Establish and Maintain a Vulnerability Management Process), CIS 4.6 (Securely Manage Enterprise Assets and Software)

Compensating: Verify TCP/5555 closure externally using 'nmap -p 5555 -sV ' from an out-of-band host or cloud instance — a filtered or closed result confirms the block. For 72-hour monitoring without SIEM, configure 'tcpdump -nn -i src and not dst -w /var/log/device_monitor.pcap' on a network choke point, rotating every 24 hours. On Android

devices restored to service, use 'adb shell crontab -l' (if cron is present on the Android-x86/embedded variant) and 'adb shell ls /etc/init.d/' to check for Mirai-installed persistence scripts. For Android TV devices, verify no unauthorized APKs were sideloaded via ADB with 'adb shell pm list packages -f | grep -v /system/' to isolate non-system-image packages installed post-compromise.

Evidence: Capture a post-recovery baseline network traffic sample (24-hour pcap) from the restored device for comparison against the anomalous outbound DDoS traffic pattern captured during the incident. Document 'adb shell getprop ro.build.fingerprint' to confirm firmware version matches the factory image and has not been modified by the Mirai payload. Retain firewall rule export and external scan results as closure evidence for incident documentation per NIST IR-5 (Incident Monitoring) requirements.

Post-Incident — Conduct an asset inventory review specifically for Android-based and IoT devices; this campaign highlights that non-traditional endpoints often lack standard hardening baselines. Implement a configuration management policy requiring ADB to be disabled by default on all non-development Android deployments. Map control gaps to CIS Benchmarks for Android devices and NIST SP 800-53 CM-6 (Configuration Settings) and IA-3 (Device Identification and Authentication).

NIST Phase: Post Incident

Reference: NIST 800-61r3 §4 — Post-Incident Activity: use lessons learned from xlabs_v1 exposure to formalize Android/IoT hardening baselines and prevent recurrence across the broader device fleet

Controls: NIST CM-6 (Configuration Settings), NIST IA-3 (Device Identification and Authentication), NIST IR-4 (Incident Handling), NIST SI-2 (Flaw Remediation), NIST SI-5 (Security Alerts, Advisories, and Directives), CIS 1.1 (Establish and Maintain Detailed Enterprise Asset Inventory), CIS 2.1 (Establish and Maintain a Software Inventory), CIS 2.2 (Ensure Authorized Software is Currently Supported), CIS 7.1 (Establish and Maintain a Vulnerability Management Process), CIS 7.2 (Establish and Maintain a Remediation Process)

Compensating: Use Shodan Monitor (free tier) or FOFA to set up a persistent alert for your organization's IP ranges on 'port:5555 product:Android' to catch future ADB re-exposure from firmware updates, new device deployments, or misconfiguration. Build a Sigma rule (free, YAML-based) targeting firewall logs for any TCP/5555 outbound or inbound connection and deploy it to any log aggregator (Graylog CE, ELK Stack). Script a recurring internal ADB sweep using 'nmap -p 5555 --open -oG adb_sweep_$(date +%F).txt' scheduled weekly via cron to detect Developer Options re-enabled after OS updates. Reference the CIS Benchmark for Android (available free at cisecurity.org) specifically Section 2 (System Updates) and Section 6 (Developer Options) for hardening checklist items.

Evidence: Compile the final incident record including: confirmed exposure window (first Shodan/Censys indexed date for TCP/5555 vs. firewall block date), list of devices confirmed enrolled in xlabs_v1 botnet (based on NetFlow DDoS outbound evidence), binary hashes of any Mirai samples recovered from /data/local/tmp/, and documented control gaps (e.g., absence of ADB-disable policy, missing IoT asset inventory). This record supports NIST IR-6 (Incident Reporting) obligations and provides the evidence baseline for the CM-6 and IA-3 gap remediation roadmap.

Detection Guidance

Primary detection vector: firewall and perimeter logs showing inbound connection attempts or established sessions on TCP/5555. Secondary: NetFlow or IPFIX data showing high-volume outbound UDP or TCP floods originating from Android-based or IoT devices, consistent with DDoS participation. On devices with accessible shell logs, look for 'adbd' process spawning shell commands not initiated by a local user. Behavioral indicator: sudden spike in outbound bandwidth from a device segment housing Android TVs, set-top boxes, or embedded Android hardware. No confirmed IOC hashes or C2 IPs have been published in the available sources at the time this item was generated; monitor threat intelligence feeds for xlabs_v1-specific infrastructure indicators as reporting matures.

Indicators of Compromise

Type	Value	Context	Confidence
URL	`TCP/5555 (ADB port)`	Active scanning target; inbound or outbound connections on this port from non-development hosts indicate potential exposure or compromise	HIGH

Framework Mappings

MITRE-ATTACK

- **T1105** — Ingress Tool Transfer
- **T1059.004** — Unix Shell
- **T1583.005** — Botnet
- **T1190** — Exploit Public-Facing Application
- **T1498** — Network Denial of Service
- **T1571** — Non-Standard Port

NIST-800-53R5

- **CA-7** — Continuous Monitoring
- **SC-7** — Boundary Protection
- **SI-3** — Malicious Code Protection
- **SI-4** — System Monitoring
- **CM-7** — Least Functionality
- **CA-8** — Penetration Testing
- **RA-5** — Vulnerability Monitoring and Scanning
- **SI-2** — Flaw Remediation
- **SI-7** — Software, Firmware, and Information Integrity
- **AC-3** — Access Enforcement
- **IA-2** — Identification and Authentication (Organizational Users)
- **IR-5** — Incident Monitoring

OWASP-TOP10-2021

- **A01:2021** — Broken Access Control
- **A07:2021** — Identification and Authentication Failures

CIS-V8

- **6.1** — Establish an Access Granting Process
- **6.2** — Establish an Access Revoking Process
- **6.3** — Require MFA for Externally-Exposed Applications

SOC2-TSC

- **CC6.1** — The entity implements logical access security software, infrastructure, and architectures over protected information assets

HIPAA-SECURITY

- **164.312(a)(1)** — Access Control

ISO-27001-2022

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

NIST-CSF-2

- **DE.AE-08** — Incidents are declared when adverse events meet the defined incident criteria

MITRE ATT&CK Mapping

Technique ID	Technique Name	Tactic
T1105	Ingress Tool Transfer	Command-And-Control
T1059.004	Unix Shell	Execution
T1583.005	Botnet	Resource-Development
T1190	Exploit Public-Facing Application	Initial-Access
T1498	Network Denial of Service	Impact
T1571	Non-Standard Port	Command-And-Control

Sources

Source	URL	Tier
gemini	https://securityboulevard.com/2026/05/palo-alto-networks-warns-of-f...	T3
Mirai-Based xlabs_v1 Botnet Exploits ADB to Hijack IoT Devices for ...	https://thehackernews.com/2026/05/mirai-based-xlabsv1-botnet-exploi...	T3
From Android TVs to routers: the xlabs_v1 Mirai-based botnet built ...	https://securityaffairs.com/191796/malware/from-android-tvs-to-rout...	T3
A Mirai-based botnet dubbed xlabs_v1 is exploiting exposed ...	https://www.facebook.com/thehackernews/posts/-a-mirai-based-botnet-...	T3

Source	URL	Tier
Botnet Hijacks ADB-Exposed Android Devices to Target Minecraft ...	https://gbhackers.com/adb-exposed-android-devices/	T3

DISCLAIMER

This intelligence report is produced by Tech Jacks Solutions Security Command Center (SCC) for informational purposes only. It does not constitute professional security advice, legal counsel, or an incident response engagement. The information herein is derived from publicly available sources and AI-assisted analysis; while every effort is made to ensure accuracy, Tech Jacks Solutions makes no warranties regarding completeness or timeliness. Organizations should conduct their own validation and consult qualified security professionals before taking action based on this report. Tech Jacks Solutions is not liable for any damages resulting from the use of this information.

Generated 2026-05-09 18:46 UTC by TJS Security Command Center