INTELLIGENCE BRIEFING
Security Command Center
TLP:CLEAR
2026-03-29 18:39 UTC

CrackArmor: Multiple Confused Deputy Vulnerabilities in Linux AppArmor Enable Local Privilege Escalation to Root

CVE VULNERABILITY | **HIGH** | CVSS 7.8

SCC Item ID	SCC-CVE-2026-0025
Type	CVE Vulnerability
Severity	HIGH
CVSS Base Score	7.8
Affected Products	Linux AppArmor (multiple versions; specific version ranges not confirmed from available sources, see Qualys TRU advisory for version scope)
Published	2026-03-26
Discovery Source	Serper

Executive Summary

Qualys Threat Research Unit disclosed nine privilege escalation vulnerabilities in Linux AppArmor, collectively named CrackArmor. An unprivileged local user can exploit confused deputy flaws in AppArmor to gain full root access without authentication. Organizations running Linux workloads where AppArmor is active should treat this as high-priority and verify patch availability immediately via the Qualys advisory and distribution vendor channels.

Technical Analysis

CrackArmor is a set of nine confused deputy vulnerabilities (CWE-269: Improper Privilege Management; CWE-610: Externally Controlled Reference to a Resource in Another Sphere) in the Linux AppArmor mandatory access control framework. A confused deputy attack occurs when a privileged component is manipulated by a lower-privileged actor into performing privileged operations on its behalf. Exploitation is local, requires no special privileges beyond a standard user account, and yields root-level access. MITRE ATT&CK mappings: T1548 (Abuse Elevation Control Mechanism) and T1068 (Exploitation for Privilege Escalation). The disclosure source is the Qualys TRU blog and a coordinated oss-security mailing list thread (seclists.org/oss-sec/2026/q1/303). CVE identifiers were not confirmed in available sources at time of this analysis; NVD and the Qualys advisory are the authoritative references for CVE assignment and CVSS scoring. A provisional CVSS base score of 7.8 (High) is reported by secondary sources. EPSS score and KEV status are not yet available. Affected version ranges require confirmation from the Qualys TRU advisory directly; specific distributions (Ubuntu, SUSE,

Debian, and others shipping AppArmor by default) are likely affected. No vendor patch status was confirmed from available sources at time of this writing.

Confidence & Limitations: This analysis is based on secondary sources (Qualys TRU and oss-sec). CVE identifiers, CVSS vectors, and specific version ranges are not confirmed from NVD. Vendor patch status is unknown. All operational decisions should be validated against the Qualys advisory and your distribution's security channel before implementation.

Action Checklist

1. Step 1: Containment. Identify all Linux systems in your environment running AppArmor (Ubuntu, SUSE, Debian, and derivatives are common AppArmor distributions). Restrict interactive and remote local shell access for non-administrative accounts on those systems until patching is confirmed. Check whether AppArmor is active: run 'aa-status' (Ubuntu/Debian) or 'apparmor_status' (SUSE), consult your distribution documentation if neither command exists. Consult the Qualys TRU advisory for confirmed affected version ranges before scoping exposure.
2. Step 2: Detection. Query asset inventory and configuration management tools for Linux hosts with AppArmor loaded in enforcement or complain mode. Review /var/log/audit/audit.log and /var/log/syslog for unexpected apparmor policy violations, DENIED entries involving privileged operations from unprivileged UIDs, or anomalous setuid/setgid activity. Note: Systems with active AppArmor policies may generate high volumes of audit.log entries. Filter on DENIED entries involving setuid/setgid, capability operations, or policy loads from non-root UIDs to reduce noise. Look for privilege escalation behavioral indicators: unexpected root-owned processes spawned from non-root parent PIDs, UID 0 processes with unusual parent-child relationships, or new entries in /etc/sudoers, /etc/passwd, or SSH authorized_keys modified by non-root accounts. No CVE-specific IOC patterns (hashes, IPs, domains) are available at this time; this is a local exploitation path with no network IOC footprint.
3. Step 3: Eradication. IMPORTANT: As of this analysis, vendor patches are not yet confirmed available. Execute this step only once patches are announced by your distribution's security advisory. Once patches are available, apply vendor-issued kernel or AppArmor package updates from your Linux distribution's security channel (e.g., apt-get update && apt-get upgrade apparmor on Debian/Ubuntu; zypper patch on SUSE). Cross-reference patch version against the Qualys advisory for version scope. Do not assume a kernel update alone resolves AppArmor-specific package flaws; verify the AppArmor package version post-patch. If patches remain unavailable, evaluate temporarily setting AppArmor to complain mode as a monitoring measure while accepting increased risk, or restrict local login access to trusted accounts only.
4. Step 4: Recovery. After patching, re-run 'aa-status' to confirm AppArmor is loaded and enforcement mode is active. Audit privileged accounts created or modified during the exposure window; check /etc/passwd, /etc/shadow, /etc/sudoers, and ~/.ssh/authorized_keys for unexpected changes. Review audit logs for root-level activity originating from non-administrative sessions in the period since systems were last known-clean. Revalidate AppArmor policy profiles are enforcing correctly on critical workloads.
5. Step 5: Post-Incident. Evaluate local access controls across Linux infrastructure: principle of least privilege for interactive accounts, SSH key hygiene, and removal of unnecessary local accounts. Assess whether your detection stack captures AppArmor audit events and routes them to SIEM for alerting on policy violations and privilege anomalies. Map this event to CIS Benchmark controls for Linux (CIS Linux Benchmark, Section 1: Initial Setup; Section 4: Logging and Auditing) and NIST SP 800-53 AC-6 (Least Privilege) and AU-2 (Event Logging) to identify control gaps.

IR / Forensic Enrichment

Triage Priority	URGENT
Escalation Criteria	Escalate immediately to CISO and legal/compliance if auditd records confirm a successful UID 0 process spawned from a non-root session (indicating active exploitation of CrackArmor), if unauthorized accounts or SSH keys are discovered in /etc/passwd or authorized_keys during recovery review, or if affected systems are confirmed to process PII, PHI, or PCI-scoped data requiring breach notification assessment under applicable regulatory frameworks.
Recovery Notes	After patching, maintain elevated auditd monitoring — specifically the priv_esc_apparmor ruleset targeting setuid, setgid, capset, and execve from non-root audit UIDs — for a minimum of 30 days to detect delayed exploitation attempts or persistence mechanisms installed prior to remediation. Revalidate AppArmor enforcement mode on all patched hosts weekly for the first month using scheduled 'aa-status' checks, and confirm no profiles have transitioned from enforce to complain mode without a change ticket. If any host showed indicators of successful exploitation during the detection phase, treat it as a confirmed compromise requiring full reimaging rather than in-place recovery, as AppArmor policy integrity and kernel state cannot be trusted post-exploitation.
Forensic Artifacts	/var/log/audit/audit.log — AVC records showing AppArmor DENIED or unexpected ALLOWED decisions for privileged syscalls (setuid, setgid, capset) initiated by non-root audit UIDs (auid >= 1000); absence of expected denial records for a known-restricted profile is itself an artifact of successful confused deputy bypass. | /proc//status for anomalous processes — Uid and Gid fields showing a mismatch between real UID (non-zero) and effective UID (0) would be a direct artifact of a successful CrackArmor privilege escalation prior to process re-exec; capture with 'cat /proc/*/status | grep -A5 "^Name:" | grep -E "Name:|Uid:"' across all running PIDs. | /etc/passwd, /etc/shadow, /etc/sudoers, /etc/sudoers.d/, /root/.ssh/authorized_keys, and /home/*/.ssh/authorized_keys — inode change times (ctime via stat) and content integrity for post-exploitation persistence; an attacker achieving root via CrackArmor would likely write a new UID 0 account or add an SSH key as a first persistence action. | auditd USER_MGMT, ADD_USER, and SUDO event records — 'ausearch -m ADD_USER,USER_MGMT,SUDO -ts ' to surface any account manipulation or sudo invocation that originated from a non-administrative session during the window AppArmor-vulnerable systems were exposed. | LiME (Linux Memory Extractor) kernel memory image captured before reboot/patch application — CrackArmor's confused deputy flaws operate within the AppArmor LSM (Linux Security Module) kernel subsystem; memory forensics may reveal manipulated AppArmor label structures, capability sets, or exploit shellcode/ROP chains in kernel address space that disk artifacts alone would not capture.

Per-Action IR Details

Step 1: Containment — Identify all Linux systems in your environment running AppArmor (Ubuntu, SUSE, Debian, and derivatives are common AppArmor distributions). Restrict interactive and remote local shell access for non-administrative accounts on those systems until patching is confirmed. Check whether AppArmor is active: run 'aa-status' or 'apparmor_status' to confirm enforcement mode. Consult the Qualys TRU advisory (blog.qualys.com) for confirmed affected version ranges before scoping exposure.

NIST Phase: Containment

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

Controls: NIST IR-4 (Incident Handling), NIST AC-6 (Least Privilege), NIST CM-7 (Least Functionality), CIS 4.6 (Securely Manage Enterprise Assets and Software), CIS 5.4 (Restrict Administrator Privileges to Dedicated Administrator Accounts)

Compensating: For a 2-person team without enterprise asset inventory: run 'for host in $(cat hosts.txt); do ssh $host "aa-status 2>/dev/null && echo APPARMOR_ACTIVE || echo APPARMOR_ABSENT"; done' to enumerate AppArmor status across known Linux hosts. Restrict non-admin shell access by temporarily modifying /etc/security/access.conf to deny login for non-privileged accounts, or use 'usermod -s /sbin/nologin ' for non-essential interactive accounts on exposed systems. Use osquery ('SELECT name, status FROM apparmor_profiles;') if deployed to query enforcement state at scale.

Evidence: Before restricting access, capture 'aa-status --verbose' output to document current AppArmor enforcement state and loaded profile list — this establishes the pre-containment baseline. Snapshot running process tree with 'ps auxf > /tmp/proctree_$(hostname)_$(date +%Y%m%d%H%M).txt' to preserve parent-child PID relationships that may reveal already-escalated processes spawned via the confused deputy flaw. Record 'id' and 'who' output for all active sessions.

Step 2: Detection — Query asset inventory and configuration management tools for Linux hosts with AppArmor loaded in enforcement or complain mode. Review /var/log/audit/audit.log and /var/log/syslog for unexpected apparmor policy violations, DENIED entries involving privileged operations from unprivileged UIDs, or anomalous setuid/setgid activity. Look for privilege escalation behavioral indicators: unexpected root-owned processes spawned from non-root parent PIDs, UID 0 processes with unusual parent-child relationships, or new entries in /etc/sudoers, /etc/passwd, or SSH authorized_keys modified by non-root accounts. No CVE-specific IOC patterns (hashes, IPs, domains) are available at this time — this is a local exploitation path with no network IOC footprint.

NIST Phase: Detection Analysis

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

Controls: NIST AU-2 (Event Logging), NIST AU-6 (Audit Record Review, Analysis, and Reporting), NIST SI-4 (System Monitoring), NIST IR-5 (Incident Monitoring), CIS 8.2 (Collect Audit Logs)

Compensating: Without a SIEM, parse /var/log/audit/audit.log directly using ausearch: 'ausearch -m AVC,USER_AVC -ts today | grep -E "uid=[^0].*exe=" | grep -i "apparmor"' to surface AppArmor-denied operations from non-root UIDs. For process anomaly detection, install Sysmon for Linux and write a rule targeting process creation where ParentUser != root and EffectiveUID == 0. Deploy this Sigma rule concept manually via auditd: 'auditctl -a always,exit -F arch=b64 -S execve -F euid=0 -F auid!=0 -k priv_esc_apparmor'. Cross-check /etc/passwd and /etc/sudoers modification timestamps with 'find /etc -name passwd -o -name sudoers -o -name shadow -newer /var/log/dpkg.log -ls' to detect post-exploitation persistence writes.

Evidence: Collect full /var/log/audit/audit.log and rotate-preserved audit logs (audit.log.1, audit.log.2) covering the exposure window — CrackArmor's confused deputy exploitation path will generate AVC denial records or, if bypassed successfully, an absence of expected denial records for privileged syscalls from non-root UIDs. Extract auditd records for syscalls setuid, setgid, capset, and execve filtered on auid (audit UID) values corresponding to non-privileged accounts. Capture /proc//status for any suspicious UID 0 processes to confirm real vs. effective UID mismatch artifacts. Check inode change times (ctime) on /etc/sudoers, /etc/passwd, and /root/.ssh/authorized_keys using 'stat' to identify unauthorized writes from non-root sessions.

Step 3: Eradication — Apply vendor-issued kernel or AppArmor package updates from your Linux distribution's security channel (e.g., apt-get update && apt-get upgrade apparmor on Debian/Ubuntu; zypper patch on SUSE) once patches are confirmed available. Cross-reference patch version against the Qualys advisory for version scope. Do not assume a kernel update alone resolves AppArmor-specific package flaws — verify the AppArmor package version post-patch. If patches are unavailable, evaluate temporarily setting AppArmor to complain mode as a monitoring measure while accepting increased risk, or restrict local login access to trusted accounts only.

NIST Phase: Eradication

Reference: NIST 800-61r3 §3.4 — Eradication

Controls: NIST SI-2 (Flaw Remediation), NIST CM-3 (Configuration Change Control), NIST SI-7 (Software, Firmware, and Information Integrity), CIS 7.3 (Perform Automated Operating System Patch Management), CIS 7.4 (Perform Automated Application Patch Management), CIS 7.2 (Establish and Maintain a Remediation Process)

Compensating: For a 2-person team managing patching manually: on Debian/Ubuntu systems, run 'apt-get changelog apparmor | head -40' after patching to confirm the changelog references the Qualys CrackArmor fixes, then verify installed version with 'dpkg -l apparmor'. On SUSE, use 'rpm -q --changelog apparmor | head -40'. If vendor patches are not yet available, harden the interim posture by combining 'usermod -s /sbin/nologin' for all non-essential local accounts with PAM restrictions in /etc/security/access.conf limiting login to admin-group members only — document this as a time-bounded compensating control with a review trigger tied to patch availability.

Evidence: Before applying patches, snapshot installed AppArmor package versions with 'dpkg -l apparmor apparmor-utils apparmor-profiles' (Debian/Ubuntu) or 'rpm -qa | grep apparmor' (SUSE/RPM-based) to document the vulnerable state for the incident record. If exploitation is suspected, preserve a memory image using LiME (Linux Memory Extractor) prior to patching — CrackArmor's confused deputy flaws could leave artifacts in kernel memory structures related to AppArmor label or capability handling that would be lost after patching and reboot. Preserve the full auditd log and a copy of /etc/apparmor.d/ profile directory to support post-incident analysis of whether any profiles were tampered with during exploitation.

Step 4: Recovery — After patching, re-run 'aa-status' to confirm AppArmor is loaded and enforcement mode is active. Audit privileged accounts created or modified during the exposure window — check /etc/passwd, /etc/shadow, /etc/sudoers, and ~/.ssh/authorized_keys for unexpected changes. Review audit logs for root-level activity originating from non-administrative sessions in the period since systems were last known-clean. Revalidate AppArmor policy profiles are enforcing correctly on critical workloads.

NIST Phase: Recovery

Reference: NIST 800-61r3 §3.5 — Recovery

Controls: NIST IR-4 (Incident Handling), NIST AU-6 (Audit Record Review, Analysis, and Reporting), NIST AC-2 (Account Management), NIST SI-6 (Security and Privacy Function Verification), CIS 5.1 (Establish and Maintain an Inventory of Accounts), CIS 5.3 (Disable Dormant Accounts)

Compensating: Verify AppArmor enforcement post-patch with 'aa-status | grep -E "profiles in enforce mode|processes in enforce mode"' and confirm the count matches pre-incident baseline. For account integrity verification without a PAM audit tool, run: 'awk -F: "($3 == 0) {print}" /etc/passwd' to list all UID 0 accounts (only root should appear); 'find /home /root -name authorized_keys -newer /var/log/dpkg.log' to identify SSH keys added during the exposure window; and 'grep -v "^#" /etc/sudoers /etc/sudoers.d/*' to review all current sudo grants. Use AIDE (Advanced Intrusion Detection Environment) if installed — run 'aide --check' to detect file integrity changes against the pre-incident database.

Evidence: During recovery validation, re-collect 'aa-status --verbose' and diff against the pre-containment baseline captured in Step 1 to confirm no AppArmor profiles were added, removed, or mode-changed by an attacker. Query auditd for all 'type=ADD_USER', 'type=USER_MGMT', and 'type=USER_CHAUTHTOK' events in the exposure window using 'ausearch -m ADD_USER,USER_MGMT -ts ' to surface unauthorized account creation or password changes that would indicate successful privilege escalation and post-exploitation persistence. Verify /etc/sudoers and /etc/sudoers.d/ inode ctimes are consistent with authorized administrative changes only.

Step 5: Post-Incident — Evaluate local access controls across Linux infrastructure: principle of least privilege for interactive accounts, SSH key hygiene, and removal of unnecessary local accounts. Assess whether your detection stack captures AppArmor audit events and routes them to SIEM for alerting on policy violations and privilege anomalies. Map this event to CIS Benchmark controls for Linux (CIS Linux Benchmark, Section 1: Initial Setup; Section 4: Logging and Auditing) and NIST SP 800-53 AC-6 (Least Privilege) and AU-2 (Event Logging) to identify control gaps.

NIST Phase: Post Incident

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

Controls: NIST IR-4 (Incident Handling), NIST IR-8 (Incident Response Plan), NIST AC-6 (Least Privilege), NIST AU-2 (Event Logging), NIST AU-11 (Audit Record Retention), NIST SI-5 (Security Alerts, Advisories, and Directives), CIS 7.1 (Establish and Maintain a Vulnerability Management Process), CIS 6.3 (Require MFA for Externally-Exposed Applications), CIS 8.2 (Collect Audit Logs)

Compensating: To improve AppArmor-specific detection posture without a commercial SIEM: configure rsyslog or syslog-ng to forward /var/log/audit/audit.log AVC records to a centralized log host, then run auditd with a persistent rule set ('auditctl -a always,exit -F arch=b64 -S setuid,setgid,capset -F auid>=1000 -F auid!=4294967295 -k priv_esc') to capture future confused-deputy-style escalation attempts targeting capability and setuid syscalls. Write a Sigma rule for the detection pattern: process creation (execve) where auid (login UID) is non-privileged and effective UID resolves to 0, mapping to MITRE ATT&CK T1068 (Exploitation for Privilege Escalation). Schedule a quarterly 'aa-status' audit via cron to alert if any profile drops from enforce to complain mode unexpectedly.

Evidence: For the lessons-learned record, collect and retain: the full auditd log corpus from the exposure window (per NIST AU-11 (Audit Record Retention) retention requirements); a diff of /etc/apparmor.d/ profiles before and after the incident window; the output of 'dpkg --get-selections | grep apparmor' or RPM equivalent showing package state at incident open and close; and any osquery or auditd query results that confirmed or ruled out active exploitation. These artifacts collectively document whether CrackArmor's confused deputy path was triggered on any host and support both internal review and any required regulatory notification assessment.

Detection Guidance

CrackArmor exploits a local privilege escalation path; there is no network-based IOC footprint to detect. Detection relies on host-level telemetry. Query SIEM or log aggregation for the following on AppArmor-enabled Linux hosts: (1) audit.log entries with 'apparmor=DENIED' or 'apparmor=ALLOWED' paired with operations that should not occur from non-root UIDs (e.g., capability additions, profile loads); (2) process creation events where a child process runs as UID 0 but the parent process is non-root (detectable via auditd syscall rules on execve with EUID tracking, or via EDR process tree telemetry); (3) modifications to /etc/passwd, /etc/sudoers, /etc/sudoers.d/*, or ~/.ssh/authorized_keys by non-root users (auditd file watch rules: -w /etc/passwd -p wa -k identity); (4) setuid/setgid binary executions from unexpected paths or by unexpected users (auditd rule: -a always,exit -F arch=b64 -S execve -F euid=0 -F auid>=1000). No CVE-specific hashes, IPs, or domains are available; exploitation is entirely local. Confirm CVE identifiers and any vendor-specific detection signatures via NVD and the Qualys TRU advisory before finalizing detection rules.

Framework Mappings

MITRE-ATTACK

- **T1548** — Abuse Elevation Control Mechanism
- **T1068** — Exploitation for Privilege Escalation

NIST-800-53R5

- **AC-6** — Least Privilege
- **CM-6** — Configuration Settings
- **SC-7** — Boundary Protection
- **SI-2** — Flaw Remediation

OWASP-TOP10-2021

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

CIS-V8

- **5.4**
- **6.8**

ISO-27001-2022

- **A.5.23** — Information security for use of cloud services

SOC2-TSC

- **CC6.3** — Authorizes, modifies, or removes access

MITRE ATT&CK Mapping

Technique ID	Technique Name	Tactic
T1548	Abuse Elevation Control Mechanism	Privilege-Escalation
T1068	Exploitation for Privilege Escalation	Privilege-Escalation

Sources

Source	URL	Tier
	https://blog.qualys.com/vulnerabilities-threat-research/2026/03/12/...	T3
Nine CrackArmor Flaws in Linux AppArmor Enable Root Escalation ...	https://thehackernews.com/2026/03/nine-crackarmor-flaws-in-linux-ap...	T3
CrackArmor Vulnerability: 12.6M Linux Servers at Risk | MaxAPEX	https://www.maxapex.com/blogs/crackarmor-vulnerability-linux-server...	T3
Critical CrackArmor Vulnerabilities Expose 12.6 Million Linux ...	https://www.cryptika.com/critical-crackarmor-vulnerabilities-expose...	T3
oss-sec: Re: Multiple vulnerabilities in AppArmor - Seclists.org	https://seclists.org/oss-sec/2026/q1/303	T3

Generated 2026-03-29 18:39 UTC by TJS Security Command Center