INTELLIGENCE BRIEFING
Security Command Center
TLP:CLEAR
2026-05-08 14:01 UTC

Dirty Frag: Chained Linux Kernel Zero-Day Grants Deterministic Root Access, No Patch Available

CVE VULNERABILITY | **CRITICAL** | CVSS 9.5

SCC Item ID	SCC-CVE-2026-0145
Type	CVE Vulnerability
CVE ID	CVE-2026-43284, CVE-2026-43500
Severity	CRITICAL
CVSS Base Score	9.5
EPSS Score	0.0001 (3th percentile)
Affected Products	Linux kernel (algif_aead, xfrm-ESP, RxRPC modules); Ubuntu, Red Hat Enterprise Linux, CentOS Stream, AlmaLinux, openSUSE Tumbleweed, Fedora, all major distributions
Published	2026-05-08T03:45:24
Discovery Source	Rss

Executive Summary

Two chained Linux kernel vulnerabilities, collectively named Dirty Frag (CVE-2026-43284, CVE-2026-43500), allow any local unprivileged user to gain full root access on affected Linux systems with no patch currently available for most distributions. A public proof-of-concept exploit exists and was released before vendors could prepare fixes, meaning the barrier to exploitation is low. Organizations running Linux servers, containers, or virtual machines face immediate risk of complete system compromise, credential theft, and lateral movement across their infrastructure.

Technical Analysis

Dirty Frag chains two kernel memory corruption flaws affecting the algif_aead, xfrm-ESP, and RxRPC kernel modules. CVE-2026-43284 (CWE-416: Use-After-Free) and CVE-2026-43500 (CWE-787: Out-of-Bounds Write) combine as page-cache write primitives to achieve deterministic local privilege escalation, not race-condition-dependent. An unprivileged local user can reliably escalate to root. CVSS base score: 9.5. All major distributions are affected: Ubuntu, RHEL, CentOS Stream, AlmaLinux, openSUSE Tumbleweed, and Fedora. Disclosure was accelerated by an embargo break before upstream patches were finalized. AlmaLinux published a fix candidate for testing on 2026-05-07; no confirmed upstream patch is generally available at time of analysis. EPSS score is 0.0014% (0.02597th percentile), low model-based exploitation probability, though this

score does not account for the public PoC or the deterministic exploitation characteristic. MITRE ATT&CK mappings: T1068 (Exploitation for Privilege Escalation), T1055 (Process Injection), T1059.004 (Unix Shell), T1203 (Exploitation for Client Execution), T1543 (Create or Modify System Process). CWE-269 (Improper Privilege Management) applies to the escalation outcome.

Action Checklist

1. Containment: Identify all Linux systems running affected kernel modules (algif_aead, xfrm-ESP, RxRPC). Where operationally feasible, unload these modules immediately using 'modprobe -r algif_aead' and equivalent commands for xfrm-ESP and RxRPC. Restrict local shell access and limit interactive login to authorized administrative accounts only. Prioritize internet-facing Linux systems and shared multi-tenant environments (container hosts, CI/CD runners, VDI).
2. Detection: Audit systems for unexpected privilege escalation events: review /var/log/auth.log (Debian/Ubuntu) or /var/log/secure (RHEL/CentOS) for su/sudo events from non-administrative users. Monitor for anomalous kernel module load/unload events via auditd rules targeting init_module and finit_module syscalls. Watch for new SUID binaries or unexpected cron entries added under root context. Check EDR telemetry for T1068 and T1055 technique signatures if applicable.
3. Eradication: Apply the AlmaLinux fix candidate (published 2026-05-07, available at https://almalinux.org/blog/2026-05-07-dirty-frag/) on AlmaLinux systems after internal testing. For all other distributions (Ubuntu, RHEL, CentOS Stream, openSUSE Tumbleweed, Fedora), monitor vendor security advisories for patch releases. Subscribe to RHEL errata at access.redhat.com and Ubuntu security notices at ubuntu.com/security/notices. Until patches are available, maintain module disablement as the primary mitigation.
4. Recovery: After patching, verify module integrity and confirm kernel version reflects the patched release using 'uname -r' and cross-referencing vendor errata. Re-enable disabled modules only after patch confirmation. Audit root-owned files and scheduled tasks modified during the exposure window. Review privileged account activity logs for the period between public PoC release and patch application.
5. Post-Incident: Conduct a privileged access review: this vulnerability exposed the risk of unrestricted local user access to kernel-level functionality. Evaluate whether least-privilege controls (restricting who can obtain interactive shell sessions on Linux systems) are enforced. Review kernel hardening configurations (e.g., kernel.unprivileged_userns_clone, seccomp profiles, AppArmor/SELinux policy coverage for affected modules). Update detection rules to alert on module unload/load anomalies as a permanent operational improvement.

IR / Forensic Enrichment

Triage Priority	IMMEDIATE
Escalation Criteria	Escalate to full incident response and initiate breach notification assessment if auth log review reveals any uid>1000 to uid=0 transition not attributable to authorized administrative activity during the window between public PoC release and module disablement, particularly on systems handling PII, PHI, PCI-DSS cardholder data, or systems with network access to production infrastructure — as Dirty Frag's deterministic root access means any such transition on a vulnerable, unmitigated system must be treated as confirmed compromise.

Recovery Notes	After patching, maintain heightened monitoring of kernel module load events (init_module/finit_module via auditd) and privileged account activity for a minimum of 30 days, as attackers with prior Dirty Frag-derived root access may have installed persistence mechanisms (cron jobs, SUID binaries, SSH authorized_keys, or kernel modules) that survive the patch. Any system where exploitation cannot be ruled out should be reimaged rather than patched in place, since the deterministic nature of the exploit means a skilled attacker had sufficient access to install kernel-resident implants not detectable by user-space tooling. Verify that /etc/modprobe.d/dirty-frag-block.conf entries are removed post-patch to restore full kernel functionality, and re-run the SUID binary audit to confirm no unauthorized binaries persist.
Forensic Artifacts	/var/log/audit/audit.log entries for syscalls 175 (init_module) and 176 (finit_module) — Dirty Frag's exploit chain interacts with algif_aead and xfrm kernel modules, and any exploit attempt or post-exploitation module manipulation will appear here if auditd was active | /var/log/auth.log (Debian/Ubuntu) or /var/log/secure (RHEL/CentOS) — look for PAM session records showing uid transitions from unprivileged users (UID >1000) to UID 0 without a corresponding authorized sudo grant, which is the direct observable outcome of successful Dirty Frag exploitation | dmesg / /var/log/kern.log — kernel ring buffer messages referencing BUG, Oops, use-after-free, or heap corruption in the algif_aead, xfrm, or rxrpc subsystems indicate either failed exploit attempts or the fragmentation primitive being exercised against the kernel heap | LiME memory image (/proc/iomem baseline + full RAM capture) taken before patch/reboot — Dirty Frag's heap fragmentation primitive leaves characteristic memory layout artifacts in kernel slab allocator state; volatile evidence of the exploit chain and any kernel-resident implants exists only in live memory and is destroyed on reboot | Filesystem timeline of SUID binaries and /etc/cron.d/, /var/spool/cron/, /root/.ssh/authorized_keys, /etc/passwd, /etc/sudoers modifications during the exposure window — deterministic root access via Dirty Frag gives an attacker immediate capability to install persistent privileged access, and these are the canonical persistence locations a post-exploitation framework would target on Linux

Per-Action IR Details

Containment — Identify all Linux systems running affected kernel modules (algif_aead, xfrm-ESP, RxRPC). Where operationally feasible, unload these modules immediately using 'modprobe -r algif_aead' and equivalent commands for xfrm-ESP and RxRPC. Restrict local shell access and limit interactive login to authorized administrative accounts only. Prioritize internet-facing Linux systems and shared multi-tenant environments (container hosts, CI/CD runners, VDI).

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: Run 'lsmod | grep -E "algif_aead|xfrm|rxrpc"' across all hosts via a parallel SSH loop (e.g., 'for h in $(cat hosts.txt); do ssh $h "lsmod | grep -E algif_aead\|xfrm\|rxrpc"; done') to inventory exposed systems before unloading. Use 'modprobe -r algif_aead xfrm_algo rxrpc' and add 'install algif_aead /bin/false', 'install rxrpc /bin/false' to /etc/modprobe.d/dirty-frag-block.conf to persist the block across reboots. Restrict interactive logins immediately by editing /etc/security/access.conf or using 'usermod -s /sbin/nologin' on non-administrative accounts.

Evidence: Before unloading modules, capture current module state with 'lsmod > /tmp/lsmod_pre_containment.txt' and 'cat /proc/modules > /tmp/proc_modules_snapshot.txt'. Collect 'dmesg' output for kernel ring buffer messages referencing algif_aead, xfrm, or rxrpc fault conditions that may indicate prior exploit attempts. Snapshot /proc/kallsyms if memory forensics is planned — Dirty Frag's heap fragmentation primitive may leave kernel memory corruption

indicators visible before reboot wipes volatile state.

Detection — Audit systems for unexpected privilege escalation events: review /var/log/auth.log (Debian/Ubuntu) or /var/log/secure (RHEL/CentOS) for su/sudo events from non-administrative users. Monitor for anomalous kernel module load/unload events via auditd rules targeting init_module and finit_module syscalls. Watch for new SUID binaries or unexpected cron entries added under root context. Check EDR telemetry for T1068 and T1055 technique signatures if applicable.

NIST Phase: Detection Analysis

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

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: Deploy auditd rules immediately: 'auditctl -a always,exit -F arch=b64 -S init_module -S finit_module -S delete_module -k dirty_frag_modules' and 'auditctl -a always,exit -F arch=b64 -S setuid -S setgid -k privesc_attempt'. Search auth logs with: 'grep -E "su|sudo" /var/log/auth.log | grep -v "pam_unix.*session opened for user root by root"' to isolate non-admin escalations. Find new SUID binaries created after initial exposure with: 'find / -perm -4000 -newer /tmp/lsmod_pre_containment.txt -type f 2>/dev/null'. Use osquery with 'SELECT * FROM suid_bin;' and 'SELECT * FROM crontab WHERE command LIKE "%root%";' for structured queries across a fleet.

Evidence: Collect /var/log/auth.log or /var/log/secure for the window starting at public PoC release (approximately 2026-05-01 based on advisory context) through present — look specifically for uid=0 transitions from unprivileged UIDs (>1000) without a corresponding sudo session. Pull auditd logs from /var/log/audit/audit.log filtering on syscalls 175 (init_module) and 176 (finit_module) to detect any module manipulation coinciding with the CVE-2026-43284/CVE-2026-43500 disclosure window. Capture output of 'stat -c "%n %U %G %a %y" $(find /usr/bin /usr/sbin /bin /sbin -perm -4000)' to establish a SUID binary baseline for comparison against post-incident state. MITRE ATT&CK T1068 (Exploitation for Privilege Escalation) and T1055 (Process Injection) are the relevant technique signatures for this kernel heap corruption chain.

Eradication — Apply the AlmaLinux fix candidate (published 2026-05-07, available at almalinux.org/blog/2026-05-07-dirty-frag/) on AlmaLinux systems after internal testing. For all other distributions (Ubuntu, RHEL, CentOS Stream, openSUSE Tumbleweed, Fedora), monitor vendor security advisories for patch releases. Subscribe to RHEL errata at access.redhat.com and Ubuntu security notices at ubuntu.com/security/notices. Until patches are available, maintain module disablement as the primary mitigation.

NIST Phase: Eradication

Reference: NIST 800-61r3 §3.4 — Eradication

Controls: NIST SI-2 (Flaw Remediation), NIST SI-5 (Security Alerts, Advisories, and Directives), NIST CM-3 (Configuration Change Control), CIS 7.1 (Establish and Maintain a Vulnerability Management Process), CIS 7.2 (Establish and Maintain a Remediation Process), CIS 7.3 (Perform Automated Operating System Patch Management)

Compensating: For AlmaLinux: download the fix candidate kernel RPM from almalinux.org/blog/2026-05-07-dirty-frag/, validate the package signature with 'rpm --checksig .rpm', test in a staging VM, then deploy with 'dnf update kernel'. For unpatched distributions, maintain /etc/modprobe.d/dirty-frag-block.conf entries and set a calendar-based polling cadence (every 48 hours minimum) against RHEL errata RSS feed (access.redhat.com/errata) and Ubuntu USN feed (ubuntu.com/security/notices.rss). If a system was confirmed compromised before module unload, treat it as fully compromised — reimage rather than patch, as Dirty Frag's deterministic root access means persistent kernel-level implants cannot be ruled out without memory forensics.

Evidence: Before applying the AlmaLinux patch, capture 'rpm -qa kernel*' and 'uname -r' output to document the pre-patch kernel version as the forensic baseline. If any system is suspected of prior exploitation (privilege escalation indicators found in step 2), collect a full memory image using LiME (Linux Memory Extractor) — 'insmod lime.ko path=/external/mem.lime format=lime' — before patching, as the patch process and reboot will destroy volatile evidence of the Dirty Frag heap fragmentation exploit chain in kernel memory. Preserve /var/log/audit/audit.log and /var/log/auth.log archives to an immutable external location before rebooting into the patched kernel.

Recovery — After patching, verify module integrity and confirm kernel version reflects the patched release using 'uname -r' and cross-referencing vendor errata. Re-enable disabled modules only after patch confirmation. Audit root-owned files and scheduled tasks modified during the exposure window. Review privileged account activity logs for the period between public PoC release and patch application.

NIST Phase: Recovery

Reference: NIST 800-61r3 §3.5 — Recovery

Controls: NIST SI-7 (Software, Firmware, and Information Integrity), NIST AU-11 (Audit Record Retention), NIST IR-4 (Incident Handling), NIST CP-10 (System Recovery and Reconstitution), CIS 7.4 (Perform Automated Application Patch Management)

Compensating: After applying the AlmaLinux patch (or equivalent when available), verify the running kernel matches the vendor-stated fixed version: 'uname -r' output must match the errata-specified kernel release string. Re-enable modules individually and validate with 'modinfo algif_aead | grep -E "version|srcversion"' to confirm module provenance. Audit root-context changes during the exposure window with: 'find /etc /usr /bin /sbin /root /var/spool/cron -newer /tmp/lsmod_pre_containment.txt -user root -type f 2>/dev/null' substituting the pre-containment snapshot timestamp as the reference point. Check for unauthorized SSH keys added to /root/.ssh/authorized_keys and new entries in /etc/passwd or /etc/sudoers during the window.

Evidence: Collect 'rpm -V kernel' (RHEL/AlmaLinux) or 'debsums -c' (Debian/Ubuntu) to verify no kernel package files were tampered with post-exploitation. Review /var/log/auth.log and /var/log/secure for the full exposure window (PoC release date through patch application) to identify any root sessions that cannot be attributed to authorized administrative activity — these represent confirmed exploitation events requiring escalation to full incident response. Pull crontab -l for all users and /var/spool/cron/* entries, and diff against any available backup to identify persistence mechanisms installed by an attacker leveraging Dirty Frag's deterministic root access.

Post-Incident — Conduct a privileged access review: this vulnerability exposed the risk of unrestricted local user access to kernel-level functionality. Evaluate whether least-privilege controls (restricting who can obtain interactive shell sessions on Linux systems) are enforced. Review kernel hardening configurations (e.g., kernel.unprivileged_userns_clone, seccomp profiles, AppArmor/SELinux policy coverage for affected modules). Update detection rules to alert on module unload/load anomalies as a permanent operational improvement.

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 SI-4 (System Monitoring), NIST CM-6 (Configuration Settings), CIS 4.6 (Securely Manage Enterprise Assets and Software), CIS 5.4 (Restrict Administrator Privileges to Dedicated Administrator Accounts), CIS 7.1 (Establish and Maintain a Vulnerability Management Process)

Compensating: Implement persistent kernel hardening via sysctl: add 'kernel.unprivileged_userns_clone=0', 'kernel.kptr_restrict=2', and 'kernel.dmesg_restrict=1' to /etc/sysctl.d/99-dirty-frag-hardening.conf and apply with 'sysctl --system'. Write a permanent auditd rule file (/etc/audit/rules.d/dirty-frag-permanent.rules) targeting init_module, finit_module, and delete_module syscalls. Draft a Sigma rule targeting auth log patterns of uid>1000 transitions to uid=0 without a preceding sudo grant for distribution to the team's log aggregator. Review and tighten AppArmor or SELinux profiles for algif_aead and rxrpc to deny loading by non-root processes even after patches are applied, as defense-in-depth against future variants of this vulnerability class.

Evidence: Produce a lessons-learned artifact documenting the time delta between public PoC release and module disablement across each affected system — this gap represents the deterministic root access exposure window for Dirty Frag and quantifies organizational risk for post-incident reporting. Retain all collected audit logs, dmesg captures, and memory images (if taken) per NIST AU-11 (Audit Record Retention) policy minimums to support potential regulatory notification obligations if privileged access was confirmed on systems handling PII or regulated data.

Detection Guidance

Primary detection targets: privilege escalation from unprivileged UID to UID 0 without a corresponding authorized sudo/su event, and anomalous kernel memory activity in algif_aead, xfrm-ESP, or RxRPC modules. Auditd rules: add watches on the execve syscall for processes spawning with euid=0 from non-root parents. Query auth logs for su/sudo events: grep -E 'sudo|su' /var/log/auth.log | grep -v 'session opened for user root by root'. For SIEM environments, alert on Linux process events where parent UID is non-zero and child UID is 0 outside approved escalation paths. Monitor for unexpected writes to /etc/passwd, /etc/shadow, or /etc/sudoers. SUID binary creation: find / -perm -4000 -newer /var/log/lastlog 2>/dev/null. No network-based IOCs are associated with this vulnerability class, exploitation is local. Behavioral indicators to prioritize: new root-owned cron jobs, unexpected SSH authorized_keys additions under root, and kernel oops/BUG messages in dmesg output referencing algif_aead or xfrm.

Framework Mappings

MITRE-ATTACK

- **T1055** — Process Injection
- **T1059.004** — Unix Shell
- **T1203** — Exploitation for Client Execution
- **T1543** — Create or Modify System Process
- **T1068** — Exploitation for Privilege Escalation

NIST-800-53R5

- **AC-6** — Least Privilege
- **SC-7** — Boundary Protection
- **SI-3** — Malicious Code Protection
- **SI-4** — System Monitoring
- **CM-7** — Least Functionality
- **SI-2** — Flaw Remediation
- **SI-16** — Memory Protection
- **IR-5** — Incident Monitoring

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
- **16.10** — Apply Secure Design Principles in Application Architectures
- **7.3** — Perform Automated Operating System Patch Management
- **7.4** — Perform Automated Application Patch Management

ISO-27001-2022

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

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

NIST-CSF-2

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

SOC2-TSC

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

MITRE ATT&CK Mapping

Technique ID	Technique Name	Tactic
T1055	Process Injection	Defense-Evasion
T1059.004	Unix Shell	Execution
T1203	Exploitation for Client Execution	Execution
T1543	Create or Modify System Process	Persistence
T1068	Exploitation for Privilege Escalation	Privilege-Escalation

Sources

Source	URL	Tier
Security News	https://www.bleepingcomputer.com/news/security/new-linux-dirty-frag...	T3
CVE-2026-43284 - Red Hat Customer Portal	https://access.redhat.com/security/cve/CVE-2026-43284	T3
Dirty Frag (CVE-2026-43284) vulnerability fix is ready for testing	https://almalinux.org/blog/2026-05-07-dirty-frag/	T3
New “Dirty Frag” Linux Kernel Vulnerability Could Lead to Root ...	https://www.reddit.com/r/cybersecurity/comments/1t6xfcu/new_dirty_f...	T3
Dirty Frag: Unpatched Linux vulnerability delivers root access	https://www.helpnetsecurity.com/2026/05/08/dirty-frag-linux-vulnera...	T3
NVD	https://nvd.nist.gov/vuln/detail/CVE-2026-43284, CVE-2026-43500	T1

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 2026-05-08 14:01 UTC by TJS Security Command Center