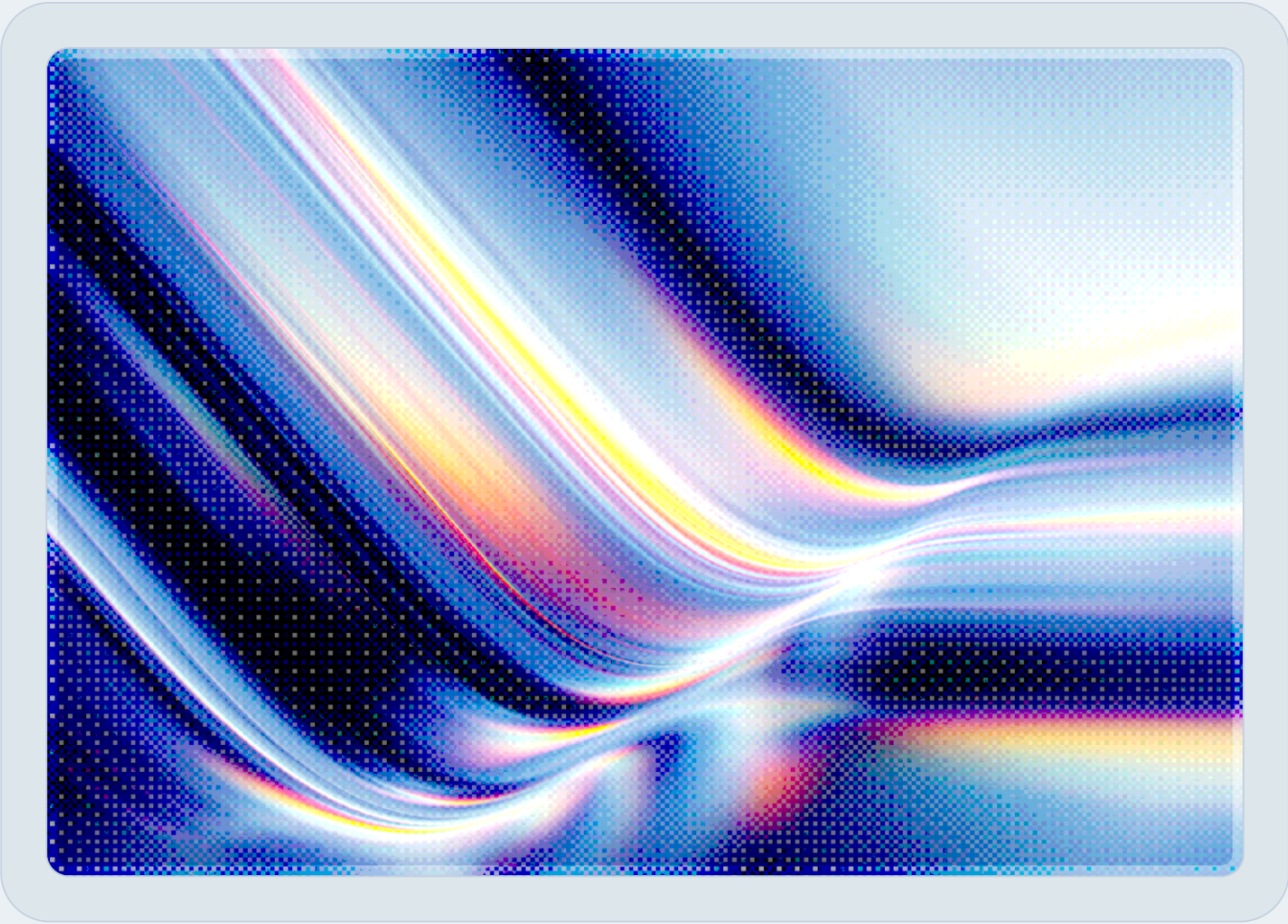
CSAI Foundation | Cloud Security Alliance

Dirty Frag: Linux Kernel LPE via ESP and RxRPC

2026-05-10

AI-assisted Rapid Research



CSAI

CSA cloud security alliance®

© 2026 Cloud Security Alliance. Some rights reserved.

You may download, store, display, view, print, redistribute, and link to this document in its original, unmodified form, provided that attribution to the Cloud Security Alliance is maintained and all trademark and copyright notices remain intact.

This document may not be modified or altered. You may quote portions of the document as permitted by the Fair Use provisions of the United States Copyright Act, provided that attribution is given to the Cloud Security Alliance.

This document may be shared on professional and social media platforms in its original form with attribution.

This document was generated with AI assistance and has not undergone official CSA review and approval processes.

Key Takeaways

- Dirty Frag (CVE-2026-43284 / CVE-2026-43500) is a chained local privilege escalation (LPE) vulnerability in the Linux kernel that allows any unprivileged local user to obtain full root access on virtually all major distributions. [1][2]
- The vulnerability was disclosed publicly on May 7–8, 2026, after a coordinated embargo was broken by an unrelated third party, and a working proof-of-concept exploit was released before most distributions had issued patches. [3][4]
- Both flaws originate in the kernel's in-place decryption path: CVE-2026-43284 affects the xfrm-ESP (IPsec) subsystem (present since January 2017), and CVE-2026-43500 affects the RxRPC subsystem; the specific vulnerable code path was introduced in June 2023. [5][6]
- Container workloads running on unpatched host kernels are at elevated risk: an attacker who achieves code execution inside a container — through application vulnerabilities, stolen credentials, or supply chain compromise — can leverage Dirty Frag to escalate from container-scoped execution to host root, achieving a full container escape. [7]
- As of this writing, patched kernels are available for RHEL-based distributions (RHEL, AlmaLinux, CentOS Stream). Ubuntu patches are still pending for all supported releases, and CVE-2026-43500 remains unpatched in mainline for many distributions. [8][9]
- Immediate mitigations include unloading the `esp4`, `esp6`, and `rxrpc` kernel modules and applying kernel live-patch tools where available, though disabling these modules will break IPsec tunnels. [10][11]

Background

On May 7, 2026, an unrelated third party broke a coordinated embargo on the Dirty Frag vulnerability chain, compelling researcher Hyunwoo Kim (known publicly as "V4bel") to release full technical documentation and a proof-of-concept exploit ahead of the originally planned disclosure window. Kim had privately reported both flaws to Linux kernel maintainers on April 29–30, 2026, and submitted candidate patches to the netdev mailing list. Distribution security teams were notified via the private linux-distros coordination list on May 7—the same day the embargo collapsed. [3][12]

The vulnerability name derives from the kernel's "dirty" page-cache write primitive: when the in-place decryption path in the ESP or RxRPC subsystems processes data from paged buffers that are not privately owned by the kernel (for example, pipe pages arriving via `splice(2)` or `sendfile(2)`), unprivileged processes can retain references to the resulting plaintext pages. An attacker who controls the input data can exploit this retained reference to overwrite arbitrary page-cache content, ultimately corrupting protected kernel or system-file mappings to achieve privilege escalation. [5][6]

Dirty Frag is technically two distinct vulnerabilities sharing a common exploitation pattern. CVE-2026-43284 resides in the `esp4` and `esp6` modules within the xfrm (IPsec) networking framework, code paths that have been present in the mainline kernel since January 2017. CVE-2026-43500 resides in the RxRPC subsystem, which has existed in the mainline kernel since approximately 2007; the specific vulnerable code path was introduced in June 2023. This means that nearly every Linux kernel shipped since 2017 is affected by at least the first flaw, with the second flaw compounding exposure on systems built from kernels that include the June 2023 change. [5][6][13]

Dirty Frag arrives in the immediate wake of Copy Fail (CVE-2026-31431), a separate Linux kernel LPE disclosed in late April 2026 that CISA added to its Known Exploited Vulnerabilities catalog on May 1, 2026 after active exploitation was confirmed. [14] In the authors' assessment, the rapid succession of kernel-level root exploits with public PoC code represents an elevated threat environment for organizations running Linux at scale. While the specific exploitation dynamics of the two vulnerabilities differ, the Copy Fail precedent illustrates how quickly kernel privilege escalation primitives can be operationalized by threat actors following public disclosure.

Security Analysis

Exploitation Mechanics

The core exploitation primitive is a writable reference into the kernel page cache that persists beyond the lifetime of the network operation that created it. By carefully staging a `splice(2)` or `sendfile(2)` call that routes pipe-backed pages through the ESP or RxRPC decryption path, an attacker with access to an unprivileged local shell can cause the kernel to decrypt ciphertext in-place over those pipe pages. Because the pages are shared rather than privately owned by the kernel buffer, the attacker retains a file descriptor that maps to the now-decrypted—and attacker-controlled—page. A subsequent write to that descriptor overwrites arbitrary read-only page-cache content, including cached copies of system binaries and configuration files. A crafted payload that overwrites a setuid executable in the page cache can then be triggered to yield a root shell. [5][6]

The chained exploitation model means neither flaw must be present alone for full impact: CVE-2026-43284 alone is sufficient for root on systems where the `esp4` or `esp6` modules are loaded, which is the default on most distributions that ship with IPsec tooling enabled. CVE-2026-43500 provides an alternative path via the RxRPC subsystem, relevant on hosts running AFS (Andrew File System) client services or similar components that depend on that protocol stack. The PoC exploit published by Kim supports automated detection and selection of the available attack path, making exploitation accessible beyond expert-level adversaries. [3][5]

Scope and Affected Systems

Dirty Frag affects all Linux distributions shipping kernel versions from 4.x onward, which encompasses every enterprise distribution currently in mainstream or extended support. Confirmed affected distributions include Red Hat Enterprise Linux (versions 8 and 9), AlmaLinux (8 and 9), CentOS Stream, Ubuntu (18.04 LTS through 24.04 LTS), Debian (10 through 13), Fedora, Arch Linux, openSUSE, Amazon Linux 2 and 2023, and CloudLinux. [8][9][10] The vulnerability is exploitable regardless of whether the host is a physical server, a virtual machine, or the underlying node of a Kubernetes or container runtime cluster.

Container environments present a distinct and elevated concern. As documented by Sysdig [7], unprivileged containers launched with default Docker, containerd, and Kubernetes pod security settings in the versions current as of disclosure retain the ability to open `AF_KEY`, `XFRM` netlink, or `AF_RXRPC` sockets — the socket families that create the preconditions for exploitation. Organizations should verify the default socket-access posture against the specific runtime versions deployed in their environments. An attacker who achieves any foothold inside a container — through application vulnerabilities, stolen credentials, or supply chain compromise — can leverage Dirty Frag to escalate from container-scoped execution to host root, achieving a full container escape with persistent access to the underlying node. This amplifies the blast radius of any container-level compromise in multi-tenant environments. [7]

Cloud-managed Kubernetes services (including managed node pools on major public clouds) are affected wherever the underlying node kernel remains unpatched. Cloud providers have begun issuing patched node images but remediation timelines vary by provider, region, and node pool configuration. Organizations relying on cloud-provider managed nodes should consult their provider's security bulletins and, where auto-upgrade of node pools is not enabled, initiate manual node replacement or kernel updates. [7][11]

Threat Actor Considerations

With a PoC that Microsoft's analysis characterizes as providing more reliable privilege escalation than traditional race-condition-dependent techniques [15], and with exploit code already circulating on GitHub as of disclosure date, the barrier to exploitation is meaningfully lower than for vulnerabilities requiring original exploit development — though it still requires understanding the PoC, adapting it to the target kernel version and configuration, and evading detection. The most realistic exploitation scenarios are post-compromise escalation: an attacker who gains a low-privileged foothold through a web shell, brute-forced SSH credentials, a compromised service account, or phishing then runs the Dirty Frag PoC to silently elevate to root before detection. Red Hat's analysis characterizes this as a post-compromise risk, underscoring that Dirty Frag is most dangerous as a second-stage component of a broader attack chain. [8]

The public PoC also increases the risk of opportunistic mass exploitation once threat actors incorporate the technique into automated attack tooling. While the specific exploitation dynamics of Copy Fail and Dirty Frag differ, the Copy Fail precedent — where CISA confirmed in-the-wild use within days of disclosure — illustrates how rapidly this class of kernel primitive can be operationalized following public release. [14]

Disclosure and Coordination Challenges

The coordinated disclosure process for Dirty Frag unfolded in compressed timeframes that proved insufficient for universal patch availability before the exploit became public. RHEL-based distributions reached production-ready kernels approximately one day after disclosure, while Ubuntu had not released patches within the initial post-disclosure window. [9][12] CVE-2026-43500's kernel-mainline fix also lagged the disclosure, meaning the second attack path remains exploitable even on systems where administrators have applied all available distribution updates. [9] The embargo breach that preceded this timeline — an unrelated third party forcing early disclosure — illustrates one failure mode of coordinated vulnerability disclosure that organizations must plan for even when vendor notification is timely. While the linux-distros coordination process has historically managed many kernel CVEs with adequate advance notice, the Dirty Frag case reinforces that security operations cannot assume coordinated disclosure will always deliver advance notice before exploitation begins.

This asymmetric patching timeline creates a window during which organizations face a known, publicly exploitable vulnerability with no available kernel patch for some of their workloads. It reinforces the importance of module-level mitigations as a bridge control during the gap between disclosure and patch availability.

Recommendations

Immediate Actions

Organizations should treat CVE-2026-43284 and CVE-2026-43500 as requiring emergency response on all Linux infrastructure. Given the public PoC availability and active post-disclosure exploitation risk, CSA recommends treating the following as emergency actions to be executed within 24–48 hours of reading this advisory.

Apply available kernel updates immediately on all RHEL, AlmaLinux, CentOS Stream, and CloudLinux systems using the vendor-supplied errata. For RHEL 9, the target kernel is version `5.14.0-611.54.3.el9_7` or newer; for AlmaLinux 9, the same version applies. For AlmaLinux 10 and corresponding RHEL 10 builds, the target is `6.12.0-124.55.2.el10_1` or newer. [10] Where kernel live-patching is available (e.g., RHEL's kpatch or Canonical's Livepatch), deploy live patches immediately to reduce reboot downtime risk.

On Ubuntu and other distributions awaiting patches, apply module-level mitigations by unloading the vulnerable modules:

```
# Remove ESP/IPsec modules
sudo modprobe -r esp4 esp6

# Remove RxRPC module
sudo modprobe -r rxrpc

# Prevent automatic reload on reboot
echo "install esp4 /bin/false" | sudo tee -a /etc/modprobe.d/dirty-frag.conf
echo "install esp6 /bin/false" | sudo tee -a /etc/modprobe.d/dirty-frag.conf
echo "install rxrpc /bin/false" | sudo tee -a /etc/modprobe.d/dirty-frag.conf
```

This mitigation will break IPsec tunnel termination on the affected host. It should not be applied on hosts that serve as IPsec gateways, strongSwan endpoints, or Libreswan VPN concentrators without first migrating IPsec offload to a patched host. [10][11] AFS client services that depend on RxRPC will similarly require migration or temporary suspension.

For Kubernetes and container environments, prioritize patching or rotating node pool images on any node accepting multi-tenant or untrusted workloads. Apply Pod Security Admission policies or equivalent admission controls that restrict `CAP_NET_ADMIN` and socket-family access in untrusted pods as an additional layer of defense-in-depth, while kernel patching is being completed. [7]

Short-Term Mitigations

Enable audit logging for privilege escalation events across all Linux hosts to support rapid detection if exploitation occurs during the patching window. Specifically, configure auditd rules to capture `setuid` execution, `/etc/shadow` modification, `/etc/sudoers` changes, and unexpected writes to binaries in `/usr/bin` and `/usr/sbin`. Correlate alerts from endpoint detection tools against the Dirty Frag PoC behavioral signature—an abnormal `splice(2)` or `sendfile(2)` call sequence from a low-privileged process followed by execution of a privileged binary. [7]

Review and tighten access controls on hosts that cannot be immediately patched. Reducing the number of user accounts with local shell access limits the population of potential Dirty Frag triggerers. Enforce SSH public-key authentication and disable password-based authentication to shrink the risk surface of initial compromise that precedes privilege escalation. Segment management networks so that lateral movement to unpatched hosts requires explicit credential reuse, making escalation chains more detectable.

Evaluate whether any cloud-provider managed node pools or auto-scaling groups are running kernels in the vulnerable range and, where the provider has issued patched AMIs, node images, or node pool versions, begin a rolling replacement of affected nodes.

Strategic Considerations

Dirty Frag and Copy Fail in rapid succession point toward systemic risk in the Linux kernel networking stack that organizations should treat as a strategic program concern rather than a one-time patch event. The pattern—long-lived page-cache write primitives reachable by unprivileged code through relatively obscure networking subsystems—warrants a broader internal review of which kernel modules are loaded

by default across the organization's fleet and whether each is operationally necessary. A kernel hardening initiative that applies the principle of least privilege to loaded modules reduces future exposure to this class of vulnerability.

Organizations that have not yet implemented a kernel live-patching capability (kpatch, ksplice, or Canonical Livepatch) should evaluate its adoption as a foundational capability. Recent incidents — including Copy Fail and Dirty Frag — reinforce the broader industry observation that the window between high-severity kernel vulnerability disclosure and active exploitation is frequently measured in days to hours; eliminating or dramatically compressing the reboot cycle for kernel updates is a structural response to that reality.

Finally, organizations that operate multi-tenant container platforms or shared Kubernetes clusters should conduct a threat model review of their tenant isolation assumptions in light of this class of kernel LPE. The existence of a kernel LPE with a public PoC on an unpatched host means that container namespace isolation cannot be relied upon to contain privilege escalation, regardless of the container runtime's own security controls. Other isolation controls — network policy, filesystem restrictions, seccomp profiles, SELinux, and AppArmor policies — retain value and should remain in place alongside strict pod security standards as the appropriate long-term posture.

CSA Resource Alignment

Dirty Frag illustrates several risk themes that intersect directly with CSA's published frameworks and ongoing research tracks.

Cloud Controls Matrix (CCM) v4: Organizations responding to this vulnerability should reference CCM controls in the Vulnerability and Patch Management (VPM) domain, particularly VPM-04 (Patch Management) and VPM-06 (Vulnerability Identification), which establish organizational obligations for timely response to disclosed vulnerabilities in infrastructure components. The container-escape dimension of Dirty Frag also activates controls in the Infrastructure and Virtualization Security (IVS) domain, where IVS-07 addresses the isolation and hardening requirements for virtualization and container environments.

Zero Trust Architecture: CSA's Zero Trust guidance is directly relevant to the post-compromise escalation model that makes Dirty Frag most dangerous. Under Zero Trust principles, hosts should be assumed potentially compromised and lateral movement should be detectable and containable. The risk that a single low-privileged foothold escalates to root—and from root to cluster-node compromise—is precisely the lateral escalation scenario that network micro-segmentation, just-in-time access controls,

and continuous host behavioral monitoring are designed to limit. Organizations that have implemented mature Zero Trust controls will be better positioned to detect and contain post-exploitation activity during the patching gap.

Shared Responsibility Model: In cloud environments, the shared responsibility model distributes patching obligations between the cloud provider and the customer. For managed Kubernetes node pools and managed Linux compute offerings, providers bear responsibility for making patched images available; customers bear responsibility for deploying them. Dirty Frag is a concrete reminder that managed infrastructure does not eliminate customer patching obligations—it shifts but does not eliminate them. CSA's STAR program and cloud provider assessments can help organizations understand the provider's contractual commitments around timely vulnerability remediation.

Incident Response and Supply Chain Security: The Dirty Frag embargo breach illustrates one failure mode of coordinated disclosure — third-party leakage — that organizations must plan for even when vendor notification is timely. Security operations should maintain the capability to detect and respond to zero-day exploitation and not assume that coordinated disclosure will always deliver advance notice before active exploitation begins. CSA's AI Safety Initiative and broader cloud security guidance both reinforce that security operations must be capable of responding to zero-day exploitation, not merely patching against known vulnerabilities after coordinated disclosure.

References

[1] Wiz Research. "Dirty Frag (CVE-2026-43284): Linux Privilege Escalation via ESP and RxRPC." Wiz Blog, May 2026.

[2] BleepingComputer. "New Linux 'Dirty Frag' zero-day gives root on all major distros." BleepingComputer, May 2026.

[3] Help Net Security. "Dirty Frag: Unpatched Linux vulnerability delivers root access." Help Net Security, May 8, 2026.

[4] Security Online. "Embargo Broken: Public PoC Released for 'Dirty Frag' Linux Kernel Exploit Granting Instant Root Access." Security Online, May 2026.

[5] The Hacker News. "Linux Kernel Dirty Frag LPE Exploit Enables Root Access Across Major Distributions." The Hacker News, May 2026.

[6] Heise Online. "'Dirty Frag': Linux flaws grant root access." Heise Online, May 2026.

[7] Sysdig. "Dirty Frag (CVE-2026-43284 and CVE-2026-43500): Detecting unpatched local privilege escalation via Linux Kernel ESP and RxRPC." Sysdig Blog, May 2026.

[8] Red Hat. "RHSB-2026-003 Networking subsystem Privilege Escalation – Linux Kernel (Dirty Frag) – (CVE-2026-43284)." Red Hat Customer Portal, May 2026.

[9] Ubuntu Security. "Dirty Frag Linux kernel local privilege escalation vulnerability mitigations." Ubuntu Blog, May 2026.

[10] AlmaLinux. "Dirty Frag (CVE-2026-43284, CVE-2026-43500) Patches Released." AlmaLinux Blog, May 7, 2026.

[11] CloudLinux. "Dirty Frag (CVE-2026-43284, CVE-2026-43500): Mitigation and Kernel Update on CloudLinux." CloudLinux Blog, May 2026.

[12] The Register. "'Dirty Frag' Linux flaw one-ups CopyFail with no patches and public root exploit." The Register, May 8, 2026.

[13] Tenable. "Dirty Frag (CVE-2026-43284, CVE-2026-43500): Linux Kernel Privilege Escalation FAQ." Tenable Blog, May 2026.

[14] Bugcrowd. "What we know about Copy Fail (CVE-2026-31431)." Bugcrowd Blog, May 2026.

[15] Microsoft Security Blog. "Active attack: Dirty Frag Linux vulnerability expands post-compromise risk." Microsoft Security Blog, May 8, 2026.