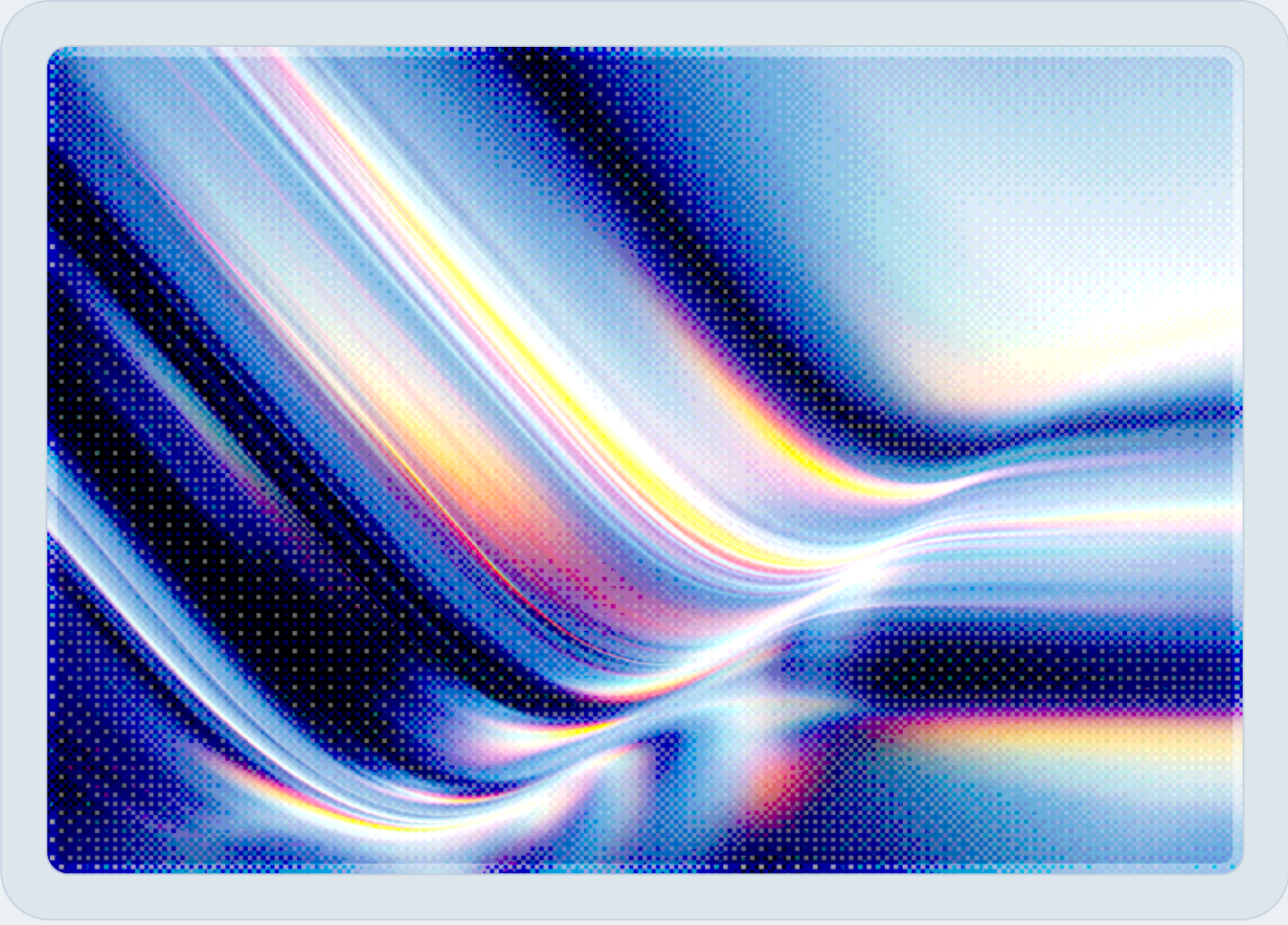
CSAI Foundation | Cloud Security Alliance

CVE-2026-98432: Reported AI-Built Remote Root Exploit Precedes Distribution Fixes

Security Implications and Guidance for the IPv4 TCP Use-After-Free and Its MPTCP Exploit Path

2026-11-01

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

CVE-2026-98432 is a use-after-free race in the Linux kernel's IPv4 TCP handshake code. It has been present since Linux 4.4 (January 2016), and a remote, unauthenticated attacker can trigger it [1][2]. On October 27, 2026, Anthropic's Frontier Red Team reported that an unreleased internal research model had written a two-stage exploit. Against a server with a Multipath TCP (MPTCP) listener, the exploit leaks a kernel pointer to defeat KASLR and then runs an attacker-chosen command as root. Anthropic reported about 48 minutes for the leak and about five more minutes to reach command execution [3]. Anthropic has not published exploit code, the kernel version, or the configuration it used, and no independent public reproduction had appeared as of October 30 [2].

The upstream fix has existed since early October, and it has reached all maintained stable branches [2][3]. Distribution coverage is uneven, though. As of October 30, Debian, Oracle UEK, SUSE (SLES 16.0 and several SLES 15 LTSS service packs), and some Amazon Linux kernels had shipped fixes. Red Hat Enterprise Linux 8–10, its rebuilds, and every supported Ubuntu LTS release still had no fixed production kernel [2][4][5][6][10][11][12][13]. AlmaLinux, departing from its usual practice of following RHEL errata, shipped backported kernels to its testing repository ahead of RHEL [1]. For several days, a publicly described AI-built remote root chain existed while some of the most widely deployed server distributions had nothing to install.

CSA recommends treating this as a priority remediation event. The first step is to patch: install a fixed kernel and reboot, or apply a vendor live patch, as soon as one is available for each platform. Where no patch exists yet, organizations should drop inbound IPv4 TCP packets that carry IP options and disable MPTCP where it is not needed [1][4]. Disabling MPTCP alone is not sufficient, however, because the bug lives in core IPv4 TCP code and the MPTCP setup is only the configuration Anthropic demonstrated [1][7]. Finally, teams should audit their Go services. Go 1.24 and later enable MPTCP on listening sockets by default, so on kernels where MPTCP is enabled (the upstream default), a Go service can expose the demonstrated attack path without anyone choosing it [7][8].

Background

The Vulnerability

The flaw is in how the kernel handles IPv4 options during the TCP three-way handshake. If an incoming SYN carries IPv4 options such as Record Route or Timestamp, the listening socket stores a copy with the pending request [1][2]. When the handshake completes, `tcp_v4_syn_recv_sock()` hands ownership of those options to the newly created child socket without copying them. Meanwhile, the client may retransmit its SYN, and another CPU may be building a SYN-ACK from the same options through `ip_options_build()`. If the child socket is torn down during that window, `inet_sock_destruct()` frees the buffer with a plain `kfree()`. The SYN-ACK path then reads freed memory [2][5]. A second variant affects request-socket migration through `tcp_v4_reqsk_destructor()`. It applies only when `net.ipv4.tcp_migrate_req=1` is set or a `BPF_SK_REUSEPORT_SELECT_OR_MIGRATE` program is attached, and both are off by default [2][5].

The upstream fix is commit `e3f33b0a1d89`, titled "ipv4: free inet_opt and ireq_opt after an RCU grace period." Eric Dumazet wrote it, Kuniyuki Iwashima reviewed it, and Xinyang Ge reported the underlying bug [1][9]. The fix replaces both immediate frees with `kfree_rcu()`, so the memory is not reclaimed until no other CPU can still read it [2][5]. The bug traces back to commit `079096f103fa` ("tcp/dccp: install syn_recv requests into ehash table"). That commit took the listener lock off the SYN-received path to improve performance, and it shipped in Linux 4.4 [2]. IPv6 is not affected by the main path, because `tcp_v6_syn_recv_sock()` copies the options rather than transferring them, although a separate IPv6 migration issue is expected to be fixed in a later patch [2][5].

The AI-Built Exploit

Anthropic's October 27 post on AI-driven vulnerability discovery described CVE-2026-98432 as its one detailed open-source example [3]. According to the post, an unreleased internal research model built a working chain after initial prompting. In the first stage, the attacker repeatedly wins the race so that the kernel over-reads and returns extra bytes. Once one of those replies contains a kernel pointer, the attacker knows the kernel's memory layout, which defeats KASLR. In the second stage, the same bug overflows a buffer and overwrites a callback pointer. The new pointer targets a payload staged earlier through a normal protocol feature, and the payload chains existing kernel functions to run a command as

root [3]. Anthropic noted that a lost race only fails the handshake and does not crash the host, so an attacker can retry as often as needed. This property makes a timing-dependent bug much more practical to exploit [3].

Several points about this claim deserve emphasis. The demonstration targeted servers with an MPTCP listener, yet neither the CVE record nor the fix commit mentions MPTCP [2]. Anthropic has not explained why MPTCP made exploitation feasible, and it has not said whether a comparable chain works against plain TCP listeners [2]. Nor has it said whether its model began from the upstream fix, the original bug report, or independent analysis of the code. The Go Security Team and AlmaLinux both stress that the bug is in ordinary IPv4 TCP code. They warn that systems without MPTCP listeners may still be exploitable through attacks that have not yet been demonstrated [1][7]. Anthropic follows a coordinated disclosure policy that allows 90 days for patching and a further 45 days before it discusses details beyond the public advisory [3]. The public description therefore lays out the chain only in general terms.

Timeline and Severity Ratings

The sequence of events matters for interpreting both the severity ratings and the patch gap discussed below, so the key dates are collected in the following table.

Date (2026)	Event	Source
Oct 5	Upstream fix `e3f33b0a1d89` applied	[2][9]
Oct 15	CVE-2026-98432 published	[5]
Oct 16–22	Debian 12/13 and Oracle UEK ship fixed kernels	[6][10]
Oct 24	Kernel CNA assigns CVSS 9.8; Amazon Linux 2023 kernel6.18 fixed	[1][2] [12]
Oct 27	Anthropic publishes AI-built remote root chain	[3]
Oct 29	Go Security Team advisory on MPTCP defaults; Red Hat raises rating to Important (8.1); Amazon Linux 2023 kernel6.12 fixed	[2][7] [13]
Oct 30	Red Hat bulletin RHSB-2026-016 (Critical); AlmaLinux testing kernels	[1][4]

Vendor severity ratings diverged sharply and shifted over the disclosure period. The kernel CNA scored the flaw 9.8 (AV:N/AC:L/PR:N/UI:N) on October 24, reasoning that an attacker can induce both sides of the race; that score predated Anthropic's report [1][2]. Red Hat first rated the flaw Moderate at 5.9, expecting a crash to be the most likely outcome and seeing no demonstrated confidentiality or integrity impact [2]. Red Hat then raised the rating to Important (8.1) on October 29, two days after Anthropic's post, and to Critical on October 30. Its October 30 bulletin states that a remote, unauthenticated attacker could achieve remote code execution, while noting that exploitation requires precise timing [2] [4]. Oracle, SUSE, and Amazon scored the flaw at 8.1 and rated attack complexity high [2]. As of October 30, NVD had not published its own score [2].

Security Analysis

The Upstream-to-Distribution Patch Gap

The vulnerability class is familiar. What stands out is the gap between the upstream fix and distribution releases. The fix was public in the kernel tree by October 5, and stable releases 5.10.272, 5.15.223, 6.1.190, 6.6.159, 6.12.113, 6.18.56, and 7.2.10 all carry it [2]. Enterprise distributions have often taken several weeks to ship kernel fixes for flaws judged hard to exploit, and many patch programs have tolerated that lag. That tolerance rests on the assumption that a "complex race, probably just a DoS" buys defenders time.

Two separate considerations now weaken that assumption. The first is long established: kernel fixes are public commits, and a capable analyst can study a diff to find the bug it corrects, a process that AI tools could plausibly speed up. The second comes from Anthropic's report itself, which suggests that the time and skill needed to build a working remote exploit for a bug of this kind may be shrinking quickly. Anthropic has not said that its model worked from the fix commit, so this case does not demonstrate a patch-to-exploit workflow. Taken together, though, the two trends suggest the window that a multi-week distribution lag leaves open is becoming more dangerous.

The following table summarizes distribution patch status at the time of writing.

Platform	Status as of Oct 30, 2026	Source
Debian 12 / 13	Fixed (DLA-4841-1, DLA-4850-1, DSA-6568-1)	[6]
Oracle Linux UEK (OL 7 / 8 / 9)	Fixed Oct 16 (ELSA-2026-500397/8/9)	[10]

Platform	Status as of Oct 30, 2026	Source
SLES 16.0, SLE Micro 6.2, SLES 15 SP4–SP6 LTSS	Fixed	[2][11]
SLES 15 SP7	Affected; fix in progress	[2][11]
Amazon Linux 2023 (kernel6.18, kernel6.12)	Fixed (ALAS2023-2026-3251, -3335)	[12][13]
Amazon Linux 2 / default AL2023 kernel	Pending	[2]
RHEL 8 / 9 / 10, OpenShift	Affected; no errata	[4]
AlmaLinux 8 / 9 / 10	Backported kernels in testing	[1]
Ubuntu 22.04 / 24.04 / 26.04 / 26.10	Vulnerable; fix in progress	[5]
Ubuntu 16.04 / 18.04 / 20.04	Vulnerable; no fix version listed	[5]
Ubuntu 14.04 `linux-lts-xenial` (4.4 HWE kernel)	Vulnerable	[5]
RHEL 7 / CentOS 7, Ubuntu 14.04 GA kernel	Not affected	[2][5]

This table reflects a single point in time, and organizations should check their vendor trackers directly, since several rows were marked "in progress" and may have changed since. Version strings alone are also unreliable. Distributions routinely backport fixes without changing the upstream kernel version, so the distribution's security tracker, not `uname -r`, should determine whether a system is patched [2]. Much of the vendor status above was first aggregated by TuxCare [2], which sells commercial live-patching and extended-support products; the primary vendor advisories cited alongside it confirm the rows where they are available.

Why MPTCP Exposure Is Broader Than Expected

The upstream kernel default for `net.mptcp.enabled` is 1, which allows applications to create MPTCP sockets [14]. Enterprise Linux 8–10 ship with MPTCP disabled [1][2], but a kernel that permits MPTCP still needs an application to open an MPTCP listener before Anthropic's demonstrated path is exposed. Go 1.24 changed its default so that `net` listeners on Linux use MPTCP unless told otherwise [8]. As a result, a Go web service, API gateway, or agent runtime built with a recent toolchain and running

on a kernel with MPTCP enabled may expose an MPTCP listener without any deliberate choice by its operator. The Go Security Team confirmed this exposure and offered `GODEBUG=multipathtcp=0` or a `godebug multipathtcp=0` directive in `go.mod` as stopgaps. It also called these measures fragile, because explicit `SetMultipathTCP` calls override them and because they assume every viable exploit depends on MPTCP [7]. The Go project has declined, for now, to change the default [7].

These defaults are especially relevant to cloud-native and AI infrastructure. Much of the control-plane and agent tooling in Kubernetes environments is written in Go, and container images often use newer toolchains than the host operating system. Because the vulnerable code runs in the shared host kernel, a single exposed listener in any container could put the entire node at risk. The MPTCP sysctl is also scoped per network namespace [14], so a host-level setting or audit does not necessarily reflect what each container sees. AlmaLinux further notes that traffic between containers on the same host may bypass a host-level nftables filter [1]. This leaves a lateral-movement path that perimeter controls do not cover.

Implications of AI-Generated Exploitation

This case is one data point, and Anthropic's specific claims have not been independently verified [2]. Even so, it fits the pattern CSA has described in earlier analyses of AI-accelerated vulnerability discovery. Exploit development used to depend on scarce expert effort, and evidence from this and earlier cases suggests that bottleneck is weakening [17]. Anthropic's post reports 62 Chrome CVEs, all rated critical or high, found with its models, alongside exploit chains in Safari and Android and privilege-escalation vulnerabilities in OpenBSD, FreeBSD, and Linux [3]. A timing race that lets the attacker retry after every failure suits automated exploitation particularly well. An AI system does not tire of thousands of attempts, and the first stage took about 48 minutes in Anthropic's single reported demonstration [3]. Defenders should assume that patch diffs for network-reachable kernel bugs will draw automated analysis from malicious actors. Severity triage that discounts "high complexity" races may need recalibrating, because complexity that slows down a human may not slow down a model.

Recommendations

Immediate Actions

The first priority is to identify every Linux host that accepts inbound IPv4 TCP from untrusted networks and to establish its patch status from vendor trackers rather than version strings. Internet-facing systems should be updated first, along with Go-based services and Kubernetes nodes. Where a fixed kernel exists (Debian, Oracle UEK, SUSE, and the fixed Amazon Linux kernels), install it and reboot, or apply a vendor

live patch if one is available [2]. AlmaLinux users with high exposure can consider the testing-repository kernels after validating them [1]. RHEL and Ubuntu users should watch RHSB-2026-016 and the Ubuntu CVE tracker and deploy fixes as soon as they are published [4][5].

Until a fix is installed, the most direct interim control is to drop inbound IPv4 TCP packets that carry IP options. Ordinary TCP options sit in the TCP header and are unaffected by this filter, and AlmaLinux notes that legitimate traffic almost never uses IPv4 options [1][4]. AlmaLinux publishes the following prerouting rule, and Red Hat's bulletin gives an equivalent variant that uses an `inet` table attached to the input hook [1][4]:

```
table ip cve_2026_98432 {
    chain prerouting {
        type filter hook prerouting priority -300; policy accept;
        meta l4proto tcp ip hdrlength > 5 drop
    }
}
```

The rule matches TCP packets whose IPv4 header length exceeds five 32-bit words (20 bytes), which is the case only when IP options are present. It does not survive a reboot unless it is persisted. The prerouting form shown above also drops forwarded packets with options, while Red Hat's input-hook variant does not. Either form breaks networks that depend on CIPSO/NetLabel labeling, and either may miss container-to-container traffic on the same host [1]. Red Hat stresses that filtering reduces exposure but does not replace patching [4]. Test the rule in non-production environments first. On perimeter firewalls, an equivalent drop for IPv4 packets with a header length greater than 20 bytes gives a further layer of defense, subject to the same CIPSO/NetLabel and legitimate-options caveats.

Short-Term Mitigations

Over the next several weeks, reduce the attack surface the demonstrated exploit relies on. Run `ss -lnM` (or `ss -H -lMn`) to inventory MPTCP listeners, and where MPTCP is not needed, set `net.mptcp.enabled=0` persistently under `/etc/sysctl.d/` and restart the affected services [1][2]. Because MPTCP sysctls are scoped per network namespace [14], these host-level steps cover only the host namespace. On container hosts, verify `net.mptcp.enabled` and enumerate listeners inside each container or pod namespace (for example, with `nsenter -n -t <pid> ss -lnM`), or set the value through the container runtime or pod sysctl configuration, and confirm what value new namespaces receive on your distribution.

For Go services, set `GODEBUG=multipathtcp=0` in deployment manifests or add `godebug multipathtcp=0` to `go.mod`, and audit code for explicit `SetMultipathTCP(true)` calls [7]. Confirm that `net.ipv4.tcp_migrate_req` is 0 and that no reuseport migration BPF programs are attached, which closes the secondary path [2][5]. These steps narrow exposure, but none of them removes the underlying bug, so they should be tracked as temporary exceptions that end once the kernel is patched.

Detection is difficult, because a lost race produces only a failed handshake [3]. Teams may still find value in watching for unusual volumes of IPv4 TCP SYNs carrying IP options and for repeated SYN retransmission patterns from single sources, and in alerting on unexpected kernel oops or KASAN reports on exposed hosts. These indicators are heuristic and have not been validated against a real exploit.

Strategic Considerations

This event should prompt a review of how organizations measure kernel patch latency. Programs that track time-to-patch from the distribution's release date miss the window that opens when the upstream fix becomes public. Measuring from the upstream commit, and maintaining relationships with vendors that ship live patches or faster backports, gives a more accurate picture of real exposure. Organizations that depend on a single enterprise distribution should consider in advance what compensating controls they would apply if their vendor is weeks behind upstream on a critical network-reachable flaw.

Severity triage policies should also account for AI-assisted exploitation. Conflicting vendor scores, ranging from 5.9 to 9.8 for the same bug [2], show how much weight assessors place on assumed exploit difficulty and on whether a race is expected to yield only a crash or full code execution. Both assumptions are becoming less reliable. Finally, defaults inherited from language runtimes, such as Go's MPTCP listener behavior, belong in configuration baselines and software bills of materials. Changes in runtime defaults can silently change a service's kernel attack surface.

CSA Resource Alignment

CSA's **Linux Kernel Root Exploits: pedit COW and DirtyClone** research note [15] analyzed two local privilege escalation flaws with public exploits, with an emphasis on multi-tenant cloud, container, and AI/ML environments. CVE-2026-98432 goes further, because it is remotely reachable and needs no prior foothold. The container and node-isolation guidance in that note applies here too, as do its points about shared-kernel risk and prioritizing multi-tenant hosts. Similarly, CSA's **GhostLock** briefing [16]

described a long-latent kernel flaw (a 15-year-old futex use-after-free) enabling container escape. CVE-2026-98432 is another decade-old flaw in heavily used core kernel code, which suggests that code age offers little assurance on its own.

CSA's white paper **AI-Accelerated Vulnerability Discovery and the Patch Debt Crisis** [17] argued that autonomous AI tools are collapsing exploitation windows faster than conventional patch management can respond, and **AI-Accelerated Exploitation and Asymmetric Vulnerability Velocity** [18] examines the same dynamic from the exploitation side. If independently confirmed, this case would be a concrete example of what those papers described: a publicly reported AI-built remote root chain appeared while major distributions had not yet shipped fixes. The patch debt paper's recommendations on exposure-based prioritization and compensating controls map directly to the interim measures above.

The **AI Controls Matrix (AICM) v1.1** [19] provides the control framework for operationalizing this guidance. Its Threat and Vulnerability Management (TVM) domain covers vulnerability identification, prioritization, and remediation timelines, including the use of compensating controls when patches are unavailable. Its Infrastructure and Virtualization Security and Application and Interface Security domains cover the network filtering and configuration hardening recommended here. Organizations operating AI and agent infrastructure on Linux should treat kernel exposure of their serving and orchestration layers as in scope for these controls.

References

[1] AlmaLinux OS Foundation. "CVE-2026-98432: Patched kernels available in testing." AlmaLinux Blog, October 30, 2026.

[2] Chris DeMars, TuxCare. "CVE-2026-98432 Linux Kernel TCP Flaw, Claude's Claimed Root Exploit and Which Distros Are Patched." TuxCare Blog, October 30, 2026.

[3] Anthropic Frontier Red Team. "Discovering and exploiting vulnerabilities in critical, real-world software with Claude." Anthropic, October 27, 2026.

[4] Red Hat. "RHSB-2026-016: Use-after-free in kernel IPv4 (CVE-2026-98432)." Red Hat Customer Portal, October 30, 2026.

[5] Canonical. "CVE-2026-98432." Ubuntu Security, last updated October 30, 2026.

[6] Debian Security Team. "CVE-2026-98432." Debian Security Bug Tracker, accessed November 1, 2026.

[7] Go Security Team. "[security] Protecting against CVE-2026-98432." golang-announce, October 29, 2026.

[8] The Go Authors. "Go, Backwards Compatibility, and GODEBUG." go.dev, accessed November 1, 2026.

[9] Linux Kernel Developers. "ipv4: free inet_opt and ireq_opt after an RCU grace period (commit e3f33b0a1d89)." kernel.org, October 5, 2026.

[10] Oracle. "CVE-2026-98432." Oracle Linux CVE Repository, accessed November 1, 2026.

[11] SUSE. "CVE-2026-98432." SUSE Security, accessed November 1, 2026.

[12] Amazon Web Services. "ALAS2023-2026-3251." Amazon Linux Security Center, October 24, 2026.

[13] Amazon Web Services. "ALAS2023-2026-3335." Amazon Linux Security Center, October 29, 2026.

[14] Linux Kernel Documentation. "MPTCP Sysfs variables." kernel.org, accessed November 1, 2026.

[15] Cloud Security Alliance. "Linux Kernel Root Exploits: pedit COW and DirtyClone." CSA AI Safety Initiative, June 2026.

© 2026 Cloud Security Alliance. AI-assisted Rapid Research.

[16] Cloud Security Alliance. "GhostLock: 15-Year-Old Linux Kernel Flaw Grants Root, Escapes Containers." CSA AI Safety Initiative, July 2026.

[17] Cloud Security Alliance. "AI-Accelerated Vulnerability Discovery and the Patch Debt Crisis." CSA AI Safety Initiative, June 2026.

[18] Cloud Security Alliance. "AI-Accelerated Exploitation and Asymmetric Vulnerability Velocity." CSA AI Safety Initiative, 2026.

[19] Cloud Security Alliance. "AI Controls Matrix (AICM) v1.1." Cloud Security Alliance, 2026.