

CVE-2026-98584

Microsoft net/sched: fix potential stack infoleak in em_text_dump()

In the Linux kernel, the following vulnerability has been resolved: net/sched: fix potential stack infoleak in em_text_dump() em_text_dump() allocates struct tcf_em_text on the stack without zeroing it. strncpy() writes the algorithm name and a NUL terminator into conf.algo[], leaving the remaining bytes uninitialised. nla_put_nohdr() then copies the full struct to the netlink response.

EVRS 0 /100 Informational priority	CVSS — Not scored	EPSS 0.002 9th percentile · FIRST	CISA DEADLINE — No CISA deadline
---	---------------------------------------	---	--

Signals behind the score

☒ counted in the score ☐ not counted

evidence for 2 of 8 signals

☐ CISA KEV	Not currently CISA KEV-listed	☐ Vendor exploitation	Active-exploitation status not yet checked - scored at the population base rate (none) until known
☐ Ransomware	Ransomware association unavailable	☐ KEV recency	CISA KEV listing date unavailable
☐ EPSS	0.002 probability	☐ Pre-patch disclosure	Disclosure status unavailable
☐ CVSS	CVSS base score unavailable	☐ Microsoft Exploitability Index	Not a Microsoft CVE

Remediation

Monitor Untagged reference available; patch status is not confirmed

NVD lists one or more references for this CVE that aren't tagged as a patch or vendor advisory. Review them directly to confirm patch availability and applicability before making a remediation decision.

Basis: NVD reference (untagged)

KMSAN on Linux 7.2-rc6 reports two kernel-infoleak splats from this path, one triggered via "tc filter show" and one via a raw RTM_GETTFILTER dump: BUG: KMSAN: kernel-infoleak in _copy_to_iter+0x1c9/0x2620 nla_put_nohdr+0x83/0x130 em_text_dump+0x291/0x550 Local variable conf created at: em_text_dump+0x5d/0x550 Bytes 168-179 of 199 are uninitialized I am not certain whether this constitutes a real security problem in practice: the test was conducted in a controlled KMSAN environment and the leaked stack bytes may or may not carry sensitive data on actual production kernels. I am reporting it because KMSAN flagged it as a kernel-infoleak and the fix is straightforward. I can provide a userspace reproducer on request. The original code used strncpy() which zero-pads to the destination size. Commit b04202d6065c ("net/sched: replace strncpy with strscpy") replaced it with strscpy(), which does not pad, creating this condition. Zero-initialising the struct closes it.

Affected and fixed versions

PACKAGE	AFFECTED	FIXED IN	SOURCE
Not identified by the connected sources	—	no fixed version listed	NVD

Sources

Reference	git.kernel.org/stable/c/0b160417826d22f24fc31a5b9cfae3b14aa70f76	Additional reference	git.kernel.org/stable/c/9923e1c537f91f3530b52d4efce85f3d9c131bf8
Additional reference	git.kernel.org/stable/c/9c572a83037a7dcd653ba3a9cc468c16b857d0c9	NVD	nvd.nist.gov/vuln/detail/CVE-2026-98584
ENISA EUVD	euvd.enisa.europa.eu/vulnerability/EUVD-2026-103629	Current data	evulnable.com/vulnerability-management/cve/CVE-2026-98584

Snapshot at generation time. EPSS can move quickly. A public exploit module is not evidence of active exploitation.