

CVE-2026-98371

Microsoft xfrm: iptfs: fix runt reassembly panic from short inner tot_len

In the Linux kernel, the following vulnerability has been resolved: xfrm: iptfs: fix runt reassembly panic from short inner tot_len
When the start of an inner packet is split across two outer packets such that fewer than 4 bytes land at the end of the first one, __input_process_payload() saves those bytes as a runt and skips the iphlen/iphlen validation performed for in-place packets.

EVRS 0 /100 Informational priority	CVSS — Not scored	EPSS 0.002 4.9th 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)

When the continuation packet arrives, iptfs_reassem_cont() only requires the declared inner length to be $\geq \text{sizeof}(\text{ra_runt})$ (6) before allocating the reassembly skb with that attacker-controlled length. However, __iptfs_iphlen() always returns the fixed minimum IP header size (20 for IPv4, 40 for IPv6), so for an inner IPv4 tot_len in [6, 19] the header-completion copy writes past the declared packet length, and the subsequent "ipremain -= copylen" underflows to ~4GB, leaving the payload copy length bounded only by blkoff (up to 64KB). At runtime the skb_put() tailroom check turns this into skb_over_panic(), i.e. an unprivileged kernel panic (DoS), reachable locally via users+netns IPTFS SAs and remotely against IPTFS VPN gateways when the decrypted outer skb is linear (e.g. AF_PACKET taps, tun/tap delivery). Align the runt path with the normal path by requiring the declared inner length to cover at least the IP header size. This also subsumes the previous $\geq \text{sizeof}(\text{ra_runt})$ check, since the minimum IP header is always larger than the runt buffer. This issue was found by the autokbug dynamic kernel fuzzer at Tencent Yunding Lab.

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/5b8afb56ccb7c014b0f1ac40341708b200443c2d	Additional reference	git.kernel.org/stable/c/a7018ee0ea8622ed59edc064be4e0f2c26464f53
Additional reference	git.kernel.org/stable/c/dc33262be1fe43d0eb0b84fb58c6ed42e2f64a8c	NVD	nvd.nist.gov/vuln/detail/CVE-2026-98371
ENISA EUVD	euvd.enisa.europa.eu/vulnerability/EUVD-2026-93356	Current data	evulnable.com/vulnerability-management/cve/CVE-2026-98371

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