

CVE-2026-98563

Microsoft net: phylink: record the PHY only once bringup cannot fail

In the Linux kernel, the following vulnerability has been resolved: net: phylink: record the PHY only once bringup cannot fail
phylink_bringup_phy() stores the PHY in pl->phydev before its last fallible step: on a MAC whose phylink ops implement LPI, phy_eee_rx_clock_stop() can fail with a real MDIO error. The callers unwind with phy_detach(), which knows nothing about pl->phydev, so a pointer to a PHY that is no longer attached outlives the failed connect. What that costs depends on how the caller got here.

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)

phylink_connect_phy() goes through phylink_attach_phy(), which refuses to attach while pl->phydev is set, turning a transient MDIO error into a permanent -EBUSY. The SFP path is worse than that: sfp_sm_probe_phy() answers the failure with phy_device_remove() and phy_device_free(), and it assigns sfp->mod_phy only past that error return, so nothing clears pl->phydev and it is left pointing at a freed phy_device that phylink_resolve() and the ethtool helpers go on reading. phylink_fwnode_phy_connect() has no such check, so a later connect overwrites the stale pointer and hides the problem. A disconnect does not: phylink_disconnect_phy() hands that pointer to phy_disconnect(), and the second phy_detach() on the same PHY drops references the first one already released. Found while making a DSA port survive a PHY whose driver arrives after the switch probes: keeping the port across a failed connect and retrying is what makes this window reachable. Publish the pointer after the last call that can fail instead of unwinding it afterwards. Nothing between the two points reads pl->phydev, and the registration that follows cannot fail: phy_request_interrupt() falls back to polling on its own. The PHY-side state keeps the order it had, so no MDIO operation moves relative to another.



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/4e7a4af364c2048bafb213bf0ef4a89dcbe1bfd4	Additional reference	git.kernel.org/stable/c/671a68654a9d319d7f0d53a6321db3d9b391bdef
Additional reference	git.kernel.org/stable/c/a940003f44e7e441c228151dd212642152700ec8	NVD	nvd.nist.gov/vuln/detail/CVE-2026-98563
ENISA EUVD	euvd.enisa.europa.eu/vulnerability/EUVD-2026-103608	Current data	evulnable.com/vulnerability-management/cve/CVE-2026-98563

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