

CVE-2026-98844

Microsoft netfs: clear post-EOF pagecache when extending a file via write

In the Linux kernel, the following vulnerability has been resolved: netfs: clear post-EOF pagecache when extending a file via write Fix netfs to erase the contents of a hole created after the EOF by an ordinary write if dirty data has been previously left there by writes through an mmapmed region. Neither the buffered nor the unbuffered/DIO write path clears that stale pagecache. Zero the tail of the folio straddling the EOF before an extending write.

EVRS 0 /100 Informational priority	CVSS — Not scored	EPSS 0.002 4.4th 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)

That is the only folio that can hold data written past the EOF through an mmap, as pages wholly beyond the EOF can't be faulted in. The folio is zeroed rather than dropped so a concurrent extending write can't lose data. Both write paths downgrade the `i_rwsem` to shared, so extending writes can run concurrently and the `i_size` read by the caller may be stale by the time the folio is locked. Re-read `i_size` under the folio lock and clamp the zeroed range up to it, so a racing write that already put data into the folio isn't clobbered. Wait for any writeback on the folio to finish before zeroing it so that the pagecache isn't modified while it may still be read by the transport during transmission. Honour `IOCB_NOWAIT` by returning `-EAGAIN` rather than blocking on the folio lock, on writeback, or in `folio_mkclean()`'s rmap walk when the folio is mapped. `truncate_pagecache()` can't be used here: it must be called with the `i_rwsem` held exclusively, but these write paths only hold it shared, and it would block unconditionally, breaking `IOCB_NOWAIT`. Callers that hold `i_rwsem` exclusively for the whole resize (`truncate`, `setattr`, `fallocate`, `clone`) exclude any genuine concurrent buffered writer, so staleness can instead be decided from the folio's dirty state, as `pagecache_isize_extended()` already does for filesystems that serialise writes against `truncate/setattr` via a single `i_rwsem`. Export `netfs_clear_stale_post_isize()` helper to handle such case. The helper is required by the CIFS client to fix generic/363.



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/69bf14300b50631e7c2271289d347bbd1b19a06c	Additional reference	git.kernel.org/stable/c/98f6a4589e0f5f6cf819b8dd45605efc481c06ce
Additional reference	git.kernel.org/stable/c/a7036a73671fc83d299bc770710fe996d9fef9ca	Additional reference	git.kernel.org/stable/c/f8f7d86b3537e885ee620898a38b749fba8a8456
NVD	nvd.nist.gov/vuln/detail/CVE-2026-98844	ENISA EUVD	euvd.enisa.europa.eu/vulnerability/EUVD-2026-103784
Current data	evulnable.com/vulnerability-management/cve/CVE-2026-98844		

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