Security Bulletin

Bulletin ID: RITTAL-SA-2026-0001
Project: RITTAL ReOS1 - CIOC1
Version: v10.1.2
Compared against: v10.0.2
Generated on: 11.06.2026
Classification: TLP:CLEAR

Summary

This bulletin documents 927 security vulnerabilities (22 Critical, 254 High, 644 Medium, 7 Low) that were fixed in RITTAL ReOS1 - CIOC1 version v10.1.2 compared to version v10.0.2. Rittal recommends updating to the latest firmware version.

Table of Contents

1. Affected Products
2. Fixed Vulnerabilities Overview
3. Fixed Vulnerabilities
4. Recommended Actions
5. Contact
6. Disclaimer

Affected Products (Article Numbers)

Article Number	Description
3313.610	Liquid Cooling Unit LCU CW, L24/W15: 7,9 kW
3314.020	SK Aktivmodul für LCP RD CW, für Höhe =2000 mm, RAL 9005 Feinstruktur matt
3314.025	SK Aktivmodul für LCP RD CW, für Höhe =2200 mm, RAL 9005 Feinstruktur matt
3314.130	LCP Rack CW, 30 kW, RAL 7035, BHT: 300x2000x1000 mm
3314.230	LCP Rack CW, 30 kW, RAL 7035, BHT: 300x2000x1200 mm
3314.238	SK LCP Rack CW, 30 kW, UL
3314.250	LCP Rack CWG, 44 kW, RAL 7035, BHT: 300x2000x1200 mm
3314.260	LCP Rack CW, 53 kW, RAL 7035, BHT: 300x2000x1200 mm
3314.268	SK LCP Rack CW, 53 kW, UL
3314.530	LCP Inline CW, 30 kW, RAL 7035, BHT: 300x2000x1200 mm
3314.538	SK LCP Inline CW vorgezogen, 30 kW, UL
3314.540	LCP Inline CW, 30 kW, RAL 7035, BHT: 300x2000x1200 mm
3314.542	LCP Inline CW, 30 kW, RAL 7035, BHT: 300x2200x1200 mm
3314.548	SK LCP Inline CW bündig, 30 kW, UL
3314.550	LCP Inline CWG, 35 kW, RAL 7035, BHT: 300x2000x1200 mm
3314.560	LCP Inline CW, 53 kW, RAL 7035, BHT: 300x2000x1200 mm
3314.568	SK LCP Inline CW vorgezogen, 53 kW, UL
3314.570	LCP Inline CWG, 44 kW, RAL 7035, BHT: 300x2000x1200 mm
3343.200	CDU In-Row, 1000 kW bei 4K ATD
3343.230	CDU In-Rack, 150 kW bei 6K ATD

Fixed Vulnerabilities Overview

All	Critical	High	Medium	Low
927	22	254	644	7

Fixed Vulnerabilities

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-57052	CRITICAL	9.8	CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H	cjson 1.7.18	cJSON 1.5.0 through 1.7.18 allows out-of-bounds access via the decode_array_index_from_pointer function in cJSON_Utils.c, allowing remote attackers to bypass array bounds checking and access restricted data via malformed JSON pointer strings containing alphanumeric characters.
CVE-2026-23112	CRITICAL	9.8	CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: nvmet-tcp: add bounds checks in nvmet_tcp_build_pdu_iovec nvmet_tcp_build_pdu_iovec() could walk past cmd->req.sg when a PDU length or offset exceeds sg_cnt and then use bogus sg->length/offset values, leading to _copy_to_iter() GPF/KASAN. Guard sg_idx, remaining entries, and sg->length/offset before building the bvec.
CVE-2025-68615	CRITICAL	9.8	CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H	net-snmp 5.9.4-rittal	net-snmp is a SNMP application library, tools and daemon. Prior to versions 5.9.5 and 5.10.pre2, a specially crafted packet to an net-snmp snmptrapd daemon can cause a buffer overflow and the daemon to crash. This issue has been patched in versions 5.9.5 and 5.10.pre2.
CVE-2026-4176	CRITICAL	9.8	CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H	perl 5.38.4	Perl versions from 5.9.4 before 5.40.4-RC1, from 5.41.0 before 5.42.2-RC1, from 5.43.0 before 5.43.9 contain a vulnerable version of Compress::Raw::Zlib. Compress::Raw::Zlib is included in the Perl package as a dual-life core module, and is vulnerable to CVE-2026-3381 due to a vendored version of zlib which has several vulnerabilities, including CVE-2026-27171. The bundled Compress::Raw::Zlib was updated to version 2.221 in Perl blead commit c75ae9cc164205e1b6d6dbd57bd2c65c8593fe94.
CVE-2019-14197	CRITICAL	9.1	CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:N/A:H	u-boot-rittal 2018.09	An issue was discovered in Das U-Boot through 2019.07. There is a read of out-of-bounds data at nfs_read_reply.
CVE-2019-11059	CRITICAL	9.8	CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H	u-boot-rittal 2018.09	Das U-Boot 2016.11-rc1 through 2019.04 mishandles the ext4 64-bit extension, resulting in a buffer overflow.
CVE-2019-14199	CRITICAL	9.8	CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H	u-boot-rittal 2018.09	An issue was discovered in Das U-Boot through 2019.07. There is an unbounded memcpy when parsing a UDP packet due to a net_process_received_packet integer underflow during an *udp_packet_handler call.
CVE-2019-14192	CRITICAL	9.8	CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H	u-boot-rittal 2018.09	An issue was discovered in Das U-Boot through 2019.07. There is an unbounded memcpy when parsing a UDP packet due to a net_process_received_packet integer underflow during an nc_input_packet call.
CVE-2019-14201	CRITICAL	9.8	CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H	u-boot-rittal 2018.09	An issue was discovered in Das U-Boot through 2019.07. There is a stack-based buffer overflow in this nfs_handler reply helper function: nfs_lookup_reply.
CVE-2019-14203	CRITICAL	9.8	CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H	u-boot-rittal 2018.09	An issue was discovered in Das U-Boot through 2019.07. There is a stack-based buffer overflow in this nfs_handler reply helper function: nfs_mount_reply.
CVE-2019-14202	CRITICAL	9.8	CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H	u-boot-rittal 2018.09	An issue was discovered in Das U-Boot through 2019.07. There is a stack-based buffer overflow in this nfs_handler reply helper function: nfs_readlink_reply.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2019-14204	CRITICAL	9.8	CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H	u-boot-rittal 2018.09	An issue was discovered in Das U-Boot through 2019.07. There is a stack-based buffer overflow in this nfs_handler reply helper function: nfs_umountall_reply.
CVE-2019-14200	CRITICAL	9.8	CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H	u-boot-rittal 2018.09	An issue was discovered in Das U-Boot through 2019.07. There is a stack-based buffer overflow in this nfs_handler reply helper function: rpc_lookup_reply.
CVE-2019-14196	CRITICAL	9.8	CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H	u-boot-rittal 2018.09	An issue was discovered in Das U-Boot through 2019.07. There is an unbounded memcpy with a failed length check at nfs_lookup_reply.
CVE-2019-14194	CRITICAL	9.8	CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H	u-boot-rittal 2018.09	An issue was discovered in Das U-Boot through 2019.07. There is an unbounded memcpy with a failed length check at nfs_read_reply when calling store_block in the NFSv2 case.
CVE-2019-14198	CRITICAL	9.8	CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H	u-boot-rittal 2018.09	An issue was discovered in Das U-Boot through 2019.07. There is an unbounded memcpy with a failed length check at nfs_read_reply when calling store_block in the NFSv3 case.
CVE-2019-14193	CRITICAL	9.8	CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H	u-boot-rittal 2018.09	An issue was discovered in Das U-Boot through 2019.07. There is an unbounded memcpy with an unvalidated length at nfs_readlink_reply, in the "if" block after calculating the new path length.
CVE-2019-14195	CRITICAL	9.8	CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H	u-boot-rittal 2018.09	An issue was discovered in Das U-Boot through 2019.07. There is an unbounded memcpy with unvalidated length at nfs_readlink_reply in the "else" block after calculating the new path length.
CVE-2022-30767	CRITICAL	9.8	CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H	u-boot-rittal 2018.09	nfs_lookup_reply in net/nfs.c in Das U-Boot through 2022.04 (and through 2022.07-rc2) has an unbounded memcpy with a failed length check, leading to a buffer overflow. NOTE: this issue exists because of an incorrect fix for CVE-2019-14196.
CVE-2022-34835	CRITICAL	9.8	CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H	u-boot-rittal 2018.09	In Das U-Boot through 2022.07-rc5, an integer signedness error and resultant stack-based buffer overflow in the "i2c md" command enables the corruption of the return address pointer of the do_i2c_md function.
CVE-2018-18439	CRITICAL	9.8	CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H	u-boot-rittal 2018.09	DENX U-Boot through 2018.09-rc1 has a remotely exploitable buffer overflow via a malicious TFTP server because TFTP traffic is mishandled. Also, local exploitation can occur via a crafted kernel image.
CVE-2020-8432	CRITICAL	9.8	CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H	u-boot-rittal 2018.09	In Das U-Boot through 2020.01, a double free has been found in the cmd/gpt.c do_rename_gpt_parts() function. Double freeing may result in a write-what-where condition, allowing an attacker to execute arbitrary code. NOTE: this vulnerablity was introduced when attempting to fix a memory leak identified by static analysis.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-39749	HIGH	7.0	CVSS:3.1/AV:L/AC:H/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: rcu: Protect ->defer_qs_iw_pending from data race On kernels built with CONFIG_IRQ_WORK=y, when rcu_read_unlock() is invoked within an interrupts-disabled region of code [1], it will invoke rcu_read_unlock_special(), which uses an irq-work handler to force the system to notice when the RCU read-side critical section actually ends. That end won't happen until interrupts are enabled at the soonest. In some kernels, such as those booted with rcutree.use_softirq=y, the irq-work handler is used unconditionally. The per-CPU rcu_data structure's ->defer_qs_iw_pending field is updated by the irq-work handler and is both read and updated by rcu_read_unlock_special(). This resulted in the following KCSAN splat: ---------- BUG: KCSAN: data-race in rcu_preempt_deferred_qs_handler / rcu_read_unlock_special read to 0xffff96b95f42d8d8 of 1 bytes by task 90 on cpu 8: rcu_read_unlock_special+0x175/0x260 __rcu_read_unlock+0x92/0xa0 rt_spin_unlock+0x9b/0xc0 __local_bh_enable+0x10d/0x170 __local_bh_enable_ip+0xfb/0x150 rcu_do_batch+0x595/0xc40 rcu_cpu_kthread+0x4e9/0x830 smpboot_thread_fn+0x24d/0x3b0 kthread+0x3bd/0x410 ret_from_fork+0x35/0x40 ret_from_fork_asm+0x1a/0x30 write to 0xffff96b95f42d8d8 of 1 bytes by task 88 on cpu 8: rcu_preempt_deferred_qs_handler+0x1e/0x30 irq_work_single+0xaf/0x160 run_irq_workd+0x91/0xc0 smpboot_thread_fn+0x24d/0x3b0 kthread+0x3bd/0x410 ret_from_fork+0x35/0x40 ret_from_fork_asm+0x1a/0x30 no locks held by irq_work/8/88. irq event stamp: 200272 hardirqs last enabled at (200272): [] finish_task_switch+0x131/0x320 hardirqs last disabled at (200271): [] __schedule+0x129/0xd70 softirqs last enabled at (0): [] copy_process+0x4df/0x1cc0 softirqs last disabled at (0): [<0000000000000000>] 0x0 ---------- The problem is that irq-work handlers run with interrupts enabled, which means that rcu_preempt_deferred_qs_handler() could be interrupted, and that interrupt handler might contain an RCU read-side critical section, which might invoke rcu_read_unlock_special(). In the strict KCSAN mode of operation used by RCU, this constitutes a data race on the ->defer_qs_iw_pending field. This commit therefore disables interrupts across the portion of the rcu_preempt_deferred_qs_handler() that updates the ->defer_qs_iw_pending field. This suffices because this handler is not a fast path.
CVE-2025-71203	HIGH	7.0	CVSS:3.1/AV:L/AC:H/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: riscv: Sanitize syscall table indexing under speculation The syscall number is a user-controlled value used to index into the syscall table. Use array_index_nospec() to clamp this value after the bounds check to prevent speculative out-of-bounds access and subsequent data leakage via cache side channels.
CVE-2025-39702	HIGH	7.0	CVSS:3.1/AV:L/AC:H/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ipv6: sr: Fix MAC comparison to be constant-time To prevent timing attacks, MACs need to be compared in constant time. Use the appropriate helper function for this.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-39759	HIGH	7.0	CVSS:3.1/AV:L/AC:H/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: btrfs: qgroup: fix race between quota disable and quota rescan ioctl There's a race between a task disabling quotas and another running the rescan ioctl that can result in a use-after-free of qgroup records from the fs_info->qgroup_tree rbtree. This happens as follows: 1) Task A enters btrfs_ioctl_quota_rescan() -> btrfs_qgroup_rescan(); 2) Task B enters btrfs_quota_disable() and calls btrfs_qgroup_wait_for_completion(), which does nothing because at that point fs_info->qgroup_rescan_running is false (it wasn't set yet by task A); 3) Task B calls btrfs_free_qgroup_config() which starts freeing qgroups from fs_info->qgroup_tree without taking the lock fs_info->qgroup_lock; 4) Task A enters qgroup_rescan_zero_tracking() which starts iterating the fs_info->qgroup_tree tree while holding fs_info->qgroup_lock, but task B is freeing qgroup records from that tree without holding the lock, resulting in a use-after-free. Fix this by taking fs_info->qgroup_lock at btrfs_free_qgroup_config(). Also at btrfs_qgroup_rescan() don't start the rescan worker if quotas were already disabled.
CVE-2026-23191	HIGH	7.0	CVSS:3.1/AV:L/AC:H/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ALSA: aloop: Fix racy access at PCM trigger The PCM trigger callback of aloop driver tries to check the PCM state and stop the stream of the tied substream in the corresponding cable. Since both check and stop operations are performed outside the cable lock, this may result in UAF when a program attempts to trigger frequently while opening/closing the tied stream, as spotted by fuzzers. For addressing the UAF, this patch changes two things: - It covers the most of code in loopback_check_format() with cable->lock spinlock, and add the proper NULL checks. This avoids already some racy accesses. - In addition, now we try to check the state of the capture PCM stream that may be stopped in this function, which was the major pain point leading to UAF.
CVE-2025-39826	HIGH	7.0	CVSS:3.1/AV:L/AC:H/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net: rose: convert 'use' field to refcount_t The 'use' field in struct rose_neigh is used as a reference counter but lacks atomicity. This can lead to race conditions where a rose_neigh structure is freed while still being referenced by other code paths. For example, when rose_neigh->use becomes zero during an ioctl operation via rose_rt_ioctl(), the structure may be removed while its timer is still active, potentially causing use-after-free issues. This patch changes the type of 'use' from unsigned short to refcount_t and updates all code paths to use rose_neigh_hold() and rose_neigh_put() which operate reference counts atomically.
CVE-2025-39687	HIGH	7.1	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: iio: light: as73211: Ensure buffer holes are zeroed Given that the buffer is copied to a kfifo that ultimately user space can read, ensure we zero it.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-39682	HIGH	7.1	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: tls: fix handling of zero-length records on the rx_list Each recvmsg() call must process either - only contiguous DATA records (any number of them) - one non-DATA record If the next record has different type than what has already been processed we break out of the main processing loop. If the record has already been decrypted (which may be the case for TLS 1.3 where we don't know type until decryption) we queue the pending record to the rx_list. Next recvmsg() will pick it up from there. Queuing the skb to rx_list after zero-copy decrypt is not possible, since in that case we decrypted directly to the user space buffer, and we don't have an skb to queue (darg.skb points to the ciphertext skb for access to metadata like length). Only data records are allowed zero-copy, and we break the processing loop after each non-data record. So we should never zero-copy and then find out that the record type has changed. The corner case we missed is when the initial record comes from rx_list, and it's zero length.
CVE-2025-39891	HIGH	7.1	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: wifi: mwifiex: Initialize the chan_stats array to zero The adapter->chan_stats[] array is initialized in mwifiex_init_channel_scan_gap() with vmalloc(), which doesn't zero out memory. The array is filled in mwifiex_update_chan_statistics() and then the user can query the data in mwifiex_cfg80211_dump_survey(). There are two potential issues here. What if the user calls mwifiex_cfg80211_dump_survey() before the data has been filled in. Also the mwifiex_update_chan_statistics() function doesn't necessarily initialize the whole array. Since the array was not initialized at the start that could result in an information leak. Also this array is pretty small. It's a maximum of 900 bytes so it's more appropriate to use kcalloc() instead vmalloc().
CVE-2026-23076	HIGH	7.1	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ALSA: ctxfi: Fix potential OOB access in audio mixer handling In the audio mixer handling code of ctxfi driver, the conf field is used as a kind of loop index, and it's referred in the index callbacks (amixer_index() and sum_index()). As spotted recently by fuzzers, the current code causes OOB access at those functions. | UBSAN: array-index-out-of-bounds in /build/reproducible-path/linux-6.17.8/sound/pci/ctxfi/ctamixer.c:347:48 | index 8 is out of range for type 'unsigned char [8]' After the analysis, the cause was found to be the lack of the proper (re-)initialization of conj field. This patch addresses those OOB accesses by adding the proper initializations of the loop indices.
CVE-2025-39757	HIGH	7.1	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ALSA: usb-audio: Validate UAC3 cluster segment descriptors UAC3 class segment descriptors need to be verified whether their sizes match with the declared lengths and whether they fit with the allocated buffer sizes, too. Otherwise malicious firmware may lead to the unexpected OOB accesses.
CVE-2025-38556	HIGH	7.1	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: HID: core: Harden s32ton() against conversion to 0 bits Testing by the syzbot fuzzer showed that the HID core gets a shift-out-of-bounds exception when it tries to convert a 32-bit quantity to a 0-bit quantity. Ideally this should never occur, but there are buggy devices and some might have a report field with size set to zero; we shouldn't reject the report or the device just because of that. Instead, harden the s32ton() routine so that it returns a reasonable result instead of crashing when it is called with the number of bits set to 0 -- the same as what snto32() does.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-39806	HIGH	7.1	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: HID: multitouch: fix slab out-of-bounds access in mt_report_fixup() A malicious HID device can trigger a slab out-of-bounds during mt_report_fixup() by passing in report descriptor smaller than 607 bytes. mt_report_fixup() attempts to patch byte offset 607 of the descriptor with 0x25 by first checking if byte offset 607 is 0x15 however it lacks bounds checks to verify if the descriptor is big enough before conducting this check. Fix this bug by ensuring the descriptor size is at least 608 bytes before accessing it. Below is the KASAN splat after the out of bounds access happens: [13.671954] ========== [13.672667] BUG: KASAN: slab-out-of-bounds in mt_report_fixup+0x103/0x110 [13.673297] Read of size 1 at addr ffff888103df39df by task kworker/0:1/10 [13.673297] [13.673297] CPU: 0 UID: 0 PID: 10 Comm: kworker/0:1 Not tainted 6.15.0-00005-gec5d573d83f4-dirty #3 [13.673297] Hardware name: QEMU Standard PC (i440FX + PIIX, 1996), BIOS 1.16.2-debian-1.16.2-1 04/04 [13.673297] Call Trace: [13.673297] [13.673297] dump_stack_lvl+0x5f/0x80 [13.673297] print_report+0xd1/0x660 [13.673297] kasan_report+0xe5/0x120 [13.673297] __asan_report_load1_noabort+0x18/0x20 [13.673297] mt_report_fixup+0x103/0x110 [13.673297] hid_open_report+0x1ef/0x810 [13.673297] mt_probe+0x422/0x960 [13.673297] hid_device_probe+0x2e2/0x6f0 [13.673297] really_probe+0x1c6/0x6b0 [13.673297] __driver_probe_device+0x24f/0x310 [13.673297] driver_probe_device+0x4e/0x220 [13.673297] __device_attach_driver+0x169/0x320 [13.673297] bus_for_each_drv+0x11d/0x1b0 [13.673297] __device_attach+0x1b8/0x3e0 [13.673297] device_initial_probe+0x12/0x20 [13.673297] bus_probe_device+0x13d/0x180 [13.673297] device_add+0xe3a/0x1670 [13.673297] hid_add_device+0x31d/0xa40 [...]

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-71133	HIGH	7.1	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: RDMA/irdma: avoid invalid read in irdma_net_event irdma_net_event() should not dereference anything from "neigh" (alias "ptr") until it has checked that the event is NETEVENT_NEIGH_UPDATE. Other events come with different structures pointed to by "ptr" and they may be smaller than struct neighbour. Move the read of neigh->dev under the NETEVENT_NEIGH_UPDATE case. The bug is mostly harmless, but it triggers KASAN on debug kernels: BUG: KASAN: stack-out-of-bounds in irdma_net_event+0x32e/0x3b0 [irdma] Read of size 8 at addr ffffc900075e07f0 by task kworker/27:2/542554 CPU: 27 PID: 542554 Comm: kworker/27:2 Kdump: loaded Not tainted 5.14.0-630.el9.x86_64+debug #1 Hardware name: [...] Workqueue: events rt6_probe_deferred Call Trace: dump_stack_lvl+0x60/0xb0 print_address_description.constprop.0+0x2c/0x3f0 print_report+0xb4/0x270 kasan_report+0x92/0xc0 irdma_net_event+0x32e/0x3b0 [irdma] notifier_call_chain+0x9e/0x180 atomic_notifier_call_chain+0x5c/0x110 rt6_do_redirect+0xb91/0x1080 tcp_v6_err+0xe9b/0x13e0 icmpv6_notify+0x2b2/0x630 ndisc_redirect_rcv+0x328/0x530 icmpv6_rcv+0xc16/0x1360 ip6_protocol_deliver_rcu+0xb84/0x12e0 ip6_input_finish+0x117/0x240 ip6_input+0xc4/0x370 ipv6_rcv+0x420/0x7d0 __netif_receive_skb_one_core+0x118/0x1b0 process_backlog+0xd1/0x5d0 __napi_poll.constprop.0+0xa3/0x440 net_rx_action+0x78a/0xba0 handle_softirqs+0x2d4/0x9c0 do_softirq+0xad/0xe0
CVE-2025-39839	HIGH	7.1	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: batman-adv: fix OOB read/write in network-coding decode batadv_nc_skb_decode_packet() trusts coded_len and checks only against skb->len. XOR starts at sizeof(struct batadv_unicast_packet), reducing payload headroom, and the source skb length is not verified, allowing an out-of-bounds read and a small out-of-bounds write. Validate that coded_len fits within the payload area of both destination and source sk_buffs before XORing.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2026-23099	HIGH	7.1	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: bonding: limit BOND_MODE_8023AD to Ethernet devices BOND_MODE_8023AD makes sense for ARPHRD_ETHER only. syzbot reported: BUG: KASAN: global-out-of-bounds in __hw_addr_create net/core/dev_addr_lists.c:63 [inline] BUG: KASAN: global-out-of-bounds in __hw_addr_add_ex+0x25d/0x760 net/core/dev_addr_lists.c:118 Read of size 16 at addr ffffffff8bf94040 by task syz.1.3580/19497 CPU: 1 UID: 0 PID: 19497 Comm: syz.1.3580 Tainted: G L syzkaller #0 PREEMPT(full) Tainted: [L]=SOFTLOCKUP Hardware name: Google Google Compute Engine/Google Compute Engine, BIOS Google 10/25/2025 Call Trace: dump_stack_lvl+0xe8/0x150 lib/dump_stack.c:120 print_address_description mm/kasan/report.c:378 [inline] print_report+0xca/0x240 mm/kasan/report.c:482 kasan_report+0x118/0x150 mm/kasan/report.c:595 check_region_inline mm/kasan/generic.c:-1 [inline] kasan_check_range+0x2b0/0x2c0 mm/kasan/generic.c:200 __asan_memcpy+0x29/0x70 mm/kasan/shadow.c:105 __hw_addr_create net/core/dev_addr_lists.c:63 [inline] __hw_addr_add_ex+0x25d/0x760 net/core/dev_addr_lists.c:118 __dev_mc_add net/core/dev_addr_lists.c:868 [inline] dev_mc_add+0xa1/0x120 net/core/dev_addr_lists.c:886 bond_enslave+0x2b8b/0x3ac0 drivers/net/bonding/bond_main.c:2180 do_set_master+0x533/0x6d0 net/core/rtnetlink.c:2963 do_setlink+0xcf0/0x41c0 net/core/rtnetlink.c:3165 rtnl_changelink net/core/rtnetlink.c:3776 [inline] __rtnl_newlink net/core/rtnetlink.c:3935 [inline] rtnl_newlink+0x161c/0x1c90 net/core/rtnetlink.c:4072 rtnetlink_rcv_msg+0x7cf/0xb70 net/core/rtnetlink.c:6958 netlink_rcv_skb+0x208/0x470 net/netlink/af_netlink.c:2550 netlink_unicast_kernel net/netlink/af_netlink.c:1318 [inline] netlink_unicast+0x82f/0x9e0 net/netlink/af_netlink.c:1344 netlink_sendmsg+0x805/0xb30 net/netlink/af_netlink.c:1894 sock_sendmsg_nosec net/socket.c:727 [inline] __sock_sendmsg+0x21c/0x270 net/socket.c:742 ____sys_sendmsg+0x505/0x820 net/socket.c:2592 ___sys_sendmsg+0x21f/0x2a0 net/socket.c:2646 __sys_sendmsg+0x164/0x220 net/socket.c:2678 do_syscall_32_irqs_on arch/x86/entry/syscall_32.c:83 [inline] __do_fast_syscall_32+0x1dc/0x560 arch/x86/entry/syscall_32.c:307 do_fast_syscall_32+0x34/0x80 arch/x86/entry/syscall_32.c:332 entry_SYSENTER_compat_after_hwframe+0x84/0x8e The buggy address belongs to the variable: lacpdu_mcast_addr+0x0/0x40

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38502	HIGH	7.1	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: bpf: Fix oob access in cgroup local storage Lonial reported that an out-of-bounds access in cgroup local storage can be crafted via tail calls. Given two programs each utilizing a cgroup local storage with a different value size, and one program doing a tail call into the other. The verifier will validate each of the indivial programs just fine. However, in the runtime context the bpf_cg_run_ctx holds an bpf_prog_array_item which contains the BPF program as well as any cgroup local storage flavor the program uses. Helpers such as bpf_get_local_storage() pick this up from the runtime context: ctx = container_of(current->bpf_ctx, struct bpf_cg_run_ctx, run_ctx); storage = ctx->prog_item->cgroup_storage[stype]; if (stype == BPF_CGROUP_STORAGE_SHARED) ptr = &READ_ONCE(storage->buf)->data[0]; else ptr = this_cpu_ptr(storage->percpu_buf); For the second program which was called from the originally attached one, this means bpf_get_local_storage() will pick up the former program's map, not its own. With mismatching sizes, this can result in an unintended out-of-bounds access. To fix this issue, we need to extend bpf_map_owner with an array of storage_cookie[] to match on i) the exact maps from the original program if the second program was using bpf_get_local_storage(), or ii) allow the tail call combination if the second program was not using any of the cgroup local storage maps.
CVE-2025-38529	HIGH	7.1	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: comedi: aio_iiro_16: Fix bit shift out of bounds When checking for a supported IRQ number, the following test is used: if ((1 << it->options[1]) & 0xdcfc) { However, `it->options[i]` is an unchecked `int` value from userspace, so the shift amount could be negative or out of bounds. Fix the test by requiring `it->options[1]` to be within bounds before proceeding with the original test. Valid `it->options[1]` values that select the IRQ will be in the range [1,15]. The value 0 explicitly disables the use of interrupts.
CVE-2025-38483	HIGH	7.1	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: comedi: das16m1: Fix bit shift out of bounds When checking for a supported IRQ number, the following test is used: /* only irqs 2, 3, 4, 5, 6, 7, 10, 11, 12, 14, and 15 are valid */ if ((1 << it->options[1]) & 0xdcfc) { However, `it->options[i]` is an unchecked `int` value from userspace, so the shift amount could be negative or out of bounds. Fix the test by requiring `it->options[1]` to be within bounds before proceeding with the original test.
CVE-2025-38482	HIGH	7.1	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: comedi: das6402: Fix bit shift out of bounds When checking for a supported IRQ number, the following test is used: /* IRQs 2,3,5,6,7, 10,11,15 are valid for "enhanced" mode */ if ((1 << it->options[1]) & 0x8cec) { However, `it->options[i]` is an unchecked `int` value from userspace, so the shift amount could be negative or out of bounds. Fix the test by requiring `it->options[1]` to be within bounds before proceeding with the original test. Valid `it->options[1]` values that select the IRQ will be in the range [1,15]. The value 0 explicitly disables the use of interrupts.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-39685	HIGH	7.1	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: comedi: pcl726: Prevent invalid irq number The reproducer passed in an irq number(0x80008000) that was too large, which triggered the oob. Added an interrupt number check to prevent users from passing in an irq number that was too large. If `it->options[1]` is 31, then `1 << it->options[1]` is still invalid because it shifts a 1-bit into the sign bit (which is UB in C). Possible solutions include reducing the upper bound on the `it->options[1]` value to 30 or lower, or using `1U << it->options[1]`. The old code would just not attempt to request the IRQ if the `options[1]` value were invalid. And it would still configure the device without interrupts even if the call to `request_irq` returned an error. So it would be better to combine this test with the test below.
CVE-2025-38530	HIGH	7.1	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: comedi: pcl812: Fix bit shift out of bounds When checking for a supported IRQ number, the following test is used: if ((1 << it->options[1]) & board->irq_bits) { However, `it->options[i]` is an unchecked `int` value from userspace, so the shift amount could be negative or out of bounds. Fix the test by requiring `it->options[1]` to be within bounds before proceeding with the original test. Valid `it->options[1]` values that select the IRQ will be in the range [1,15]. The value 0 explicitly disables the use of interrupts.
CVE-2025-39869	HIGH	7.1	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: dmaengine: ti: edma: Fix memory allocation size for queue_priority_map Fix a critical memory allocation bug in edma_setup_from_hw() where queue_priority_map was allocated with insufficient memory. The code declared queue_priority_map as s8 (*)[2] (pointer to array of 2 s8), but allocated memory using sizeof(s8) instead of the correct size. This caused out-of-bounds memory writes when accessing: queue_priority_map[i][0] = i; queue_priority_map[i][1] = i; The bug manifested as kernel crashes with "Oops - undefined instruction" on ARM platforms (BeagleBoard-X15) during EDMA driver probe, as the memory corruption triggered kernel hardening features on Clang. Change the allocation to use sizeof(*queue_priority_map) which automatically gets the correct size for the 2D array structure.
CVE-2025-21985	HIGH	7.1	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: drm/amd/display: Fix out-of-bound accesses [WHAT & HOW] hpo_stream_to_link_encoder_mapping has size MAX_HPO_DP2_ENCODERS(=4), but location can have size up to 6. As a result, it is necessary to check location against MAX_HPO_DP2_ENCODERS. Similiarly, disp_cfg_stream_location can be used as an array index which should be 0..5, so the ASSERT's conditions should be less without equal.
CVE-2025-71093	HIGH	7.1	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: e1000: fix OOB in e1000_tbi_should_accept() In e1000_tbi_should_accept() we read the last byte of the frame via 'data[length - 1]' to evaluate the TBI workaround. If the descriptor- reported length is zero or larger than the actual RX buffer size, this read goes out of bounds and can hit unrelated slab objects. The issue is observed from the NAPI receive path (e1000_clean_rx_irq): ================================== ========================== BUG: KASAN: slab-out-of-bounds in e1000_tbi_should_accept+0x610/0x790 Read of size 1 at addr ffff888014114e54 by task sshd/363 CPU: 0 PID: 363 Comm: sshd Not tainted 5.18.0-rc1 #1 Hardware name: QEMU Standard PC (i440FX + PIIX, 1996), BIOS rel-1.12.0-59-

CVE	Severity	CVSS	CVSS Vector	Component	Description
					gc9ba5276e321-prebuilt.qemu.org 04/01/2014 Call Trace: dump_stack_lvl+0x5a/0x74 print_address_description+0x7b/0x440 print_report+0x101/0x200 kasan_report+0xc1/0xf0 e1000_tbi_should_accept+0x610/0x790 e1000_clean_rx_irq+0xa8c/0x1110 e1000_clean+0xde2/0x3c10 __napi_poll+0x98/0x380 net_rx_action+0x491/0xa20 __do_softirq+0x2c9/0x61d do_softirq+0xd1/0x120 __local_bh_enable_ip+0xfe/0x130 ip_finish_output2+0x7d5/0xb00 __ip_queue_xmit+0xe24/0x1ab0 __tcp_transmit_skb+0x1bcb/0x3340 tcp_write_xmit+0x175d/0x6bd0 __tcp_push_pending_frames+0x7b/0x280 tcp_sendmsg_locked+0x2e4f/0x32d0 tcp_sendmsg+0x24/0x40 sock_write_iter+0x322/0x430 vfs_write+0x56c/0xa60 ksys_write+0xd1/0x190 do_syscall_64+0x43/0x90 entry_SYSCALL_64_after_hwframe+0x44/0xae RIP: 0033:0x7f511b476b10 Code: 73 01 c3 48 8b 0d 88 d3 2b 00 f7 d8 64 89 01 48 83 c8 ff c3 66 0f 1f 44 00 00 83 3d f9 2b 2c 00 00 75 10 b8 01 00 00 00 0f 05 <48> 3d 01 f0 ff ff 73 31 c3 48 83 ec 08 e8 8e 9b 01 00 48 89 04 24 RSP: 002b:00007ffc9211d4e8 EFLAGS: 00000246 ORIG_RAX: 0000000000000001 RAX: ffffffffffffffda RBX: 0000000000004024 RCX: 00007f511b476b10 RDX: 0000000000004024 RSI: 0000559a9385962c RDI: 0000000000000003 RBP: 0000559a9383a400 R08: ffffffffffffffff0 R09: 0000000000004f00 R10: 0000000000000070 R11: 0000000000000246 R12: 0000000000000000 R13: 00007ffc9211d57f R14: 0000559a9347bde7 R15: 0000000000000003 Allocated by task 1: __kasan_krealloc+0x131/0x1c0 krealloc+0x90/0xc0 add_sysfs_param+0xcb/0x8a0 kernel_add_sysfs_param+0x81/0xd4 param_sysfs_builtin+0x138/0x1a6 param_sysfs_init+0x57/0x5b do_one_initcall+0x104/0x250 do_initcall_level+0x102/0x132 do_initcalls+0x46/0x74 kernel_init_freeable+0x28f/0x393 kernel_init+0x14/0x1a0 ret_from_fork+0x22/0x30 The buggy address belongs to the object at ffff888014114000 which belongs to the cache kmalloc-2k of size 2048 The buggy address is located 1620 bytes to the right of 2048-byte region [ffff888014114000, ffff888014114800] The buggy address belongs to the physical page: page:ffffea0000504400 refcount:1 mapcount:0 mapping:0000000000000000 index0x0 pfn:0x14110 head:ffffea0000504400 order:3 compound_mapcount:0 compound_pincount:0 flags: 0x100000000010200(slab|head|node=0|zone=1) raw: 0100000000010200 0000000000000000 dead000000000001 ffff888013442000 raw: 0000000000000000 0000000000080008 00000001ffffffff 0000000000000000 page dumped because: kasan: bad access detected ========== ========== This happens because the TBI check unconditionally dereferences the last byte without validating the reported length first: u8 last_byte = *(data + length - 1); Fix by rejecting the frame early if the length is zero, or if it exceeds adapter->rx_buffer_len. This preserves the TBI workaround semantics for valid frames and prevents touching memory beyond the RX buffer.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-39817	HIGH	7.1	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: efivarfs: Fix slab-out-of-bounds in efivarfs_d_compare Observed on kernel 6.6 (present on master as well): BUG: KASAN: slab-out-of-bounds in memcmp+0x98/0xd0 Call trace: kasan_check_range+0xe8/0x190 __asan_loadN+0x1c/0x28 memcmp+0x98/0xd0 efivarfs_d_compare+0x68/0xd8 __d_lookup_rcu_op_compare+0x178/0x218 __d_lookup_rcu+0x1f8/0x228 d_alloc_parallel+0x150/0x648 lookup_open.isra.0+0x5f0/0x8d0 open_last_lookups+0x264/0x828 path_openat+0x130/0x3f8 do_filp_open+0x114/0x248 do_sys_openat2+0x340/0x3c0 __arm64_sys_openat+0x120/0x1a0 If dentry->d_name.len < EFI_VARIABLE_GUID_LEN , 'guid' can become negative, leadings to oob. The issue can be triggered by parallel lookups using invalid filename: T1 T2 lookup_open ->lookup simple_lookup d_add // invalid dentry is added to hash list lookup_open d_alloc_parallel __d_lookup_rcu __d_lookup_rcu_op_compare hlist_bl_for_each_entry_rcu // invalid dentry can be retrieved ->d_compare efivarfs_d_compare // oob Fix it by checking 'guid' before cmp.
CVE-2026-23235	HIGH	7.1	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: f2fs: fix out-of-bounds access in sysfs attribute read/write Some f2fs sysfs attributes suffer from out-of-bounds memory access and incorrect handling of integer values whose size is not 4 bytes. For example: vm:~# echo 65537 > /sys/fs/f2fs/vde/carve_out vm:~# cat /sys/fs/f2fs/vde/carve_out 65537 vm:~# echo 4294967297 > /sys/fs/f2fs/vde/atgc_age_threshold vm:~# cat /sys/fs/f2fs/vde/atgc_age_threshold 1 carve_out maps to {struct f2fs_sb_info}->carve_out, which is a 8-bit integer. However, the sysfs interface allows setting it to a value larger than 255, resulting in an out-of-range update. atgc_age_threshold maps to {struct atgc_management}->age_threshold, which is a 64-bit integer, but its sysfs interface cannot correctly set values larger than UINT_MAX. The root causes are: 1. __sbi_store() treats all default values as unsigned int, which prevents updating integers larger than 4 bytes and causes out-of-bounds writes for integers smaller than 4 bytes. 2. f2fs_sbi_show() also assumes all default values are unsigned int, leading to out-of-bounds reads and incorrect access to integers larger than 4 bytes. This patch introduces {struct f2fs_attr}->size to record the actual size of the integer associated with each sysfs attribute. With this information, sysfs read and write operations can correctly access and update values according to their real data size, avoiding memory corruption and truncation.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38652	HIGH	7.1	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: f2fs: fix to avoid out-of-boundary access in devs.path - touch /mnt/f2fs/0123456789012345678901234567890123456789012345678901234567890123 - truncate -s $((1024*1024*1024)) \ /mnt/f2fs/0123456789012345678901234567890123456789012345678901234567890123 - touch /mnt/f2fs/file - truncate -s $((1024*1024*1024)) /mnt/f2fs/file - mkfs.f2fs /mnt/f2fs/0123456789012345678901234567890123456789012345678901234567890123 \ -c /mnt/f2fs/file - mount /mnt/f2fs/0123456789012345678901234567890123456789012345678901234567890123 \ /mnt/f2fs/loop [16937.192225] F2FS-fs (loop0): Mount Device [0]: /mnt/f2fs/0123456789012345678901234567890123456789012345678901234567890123\xff\x01, 511, 0 - 3ffff [16937.192268] F2FS-fs (loop0): Failed to find devices If device path length equals to MAX_PATH_LEN, sbi->devs.path[] may not end up w/ null character due to path array is fully filled, So accidently, fields locate after path[] may be treated as part of device path, result in parsing wrong device path. struct f2fs_dev_info { ... char path[MAX_PATH_LEN]; ... }; Let's add one byte space for sbi->devs.path[] to store null character of device path string.
CVE-2025-38677	HIGH	7.1	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: f2fs: fix to avoid out-of-boundary access in dnode page As Jiaming Zhang reported: __dump_stack lib/dump_stack.c:94 [inline] dump_stack_lvl+0x1c1/0x2a0 lib/dump_stack.c:120 print_address_description mm/kasan/report.c:378 [inline] print_report+0x17e/0x800 mm/kasan/report.c:480 kasan_report+0x147/0x180 mm/kasan/report.c:593 data_blkaddr fs/f2fs/f2fs.h:3053 [inline] f2fs_data_blkaddr fs/f2fs/f2fs.h:3058 [inline] f2fs_get_dnode_of_data+0x1a09/0x1c40 fs/f2fs/node.c:855 f2fs_reserve_block+0x53/0x310 fs/f2fs/data.c:1195 prepare_write_begin fs/f2fs/data.c:3395 [inline] f2fs_write_begin+0xf39/0x2190 fs/f2fs/data.c:3594 generic_perform_write+0x2c7/0x910 mm/filemap.c:4112 f2fs_buffered_write_iter fs/f2fs/file.c:4988 [inline] f2fs_file_write_iter+0x1ec8/0x2410 fs/f2fs/file.c:5216 new_sync_write fs/read_write.c:593 [inline] vfs_write+0x546/0xa90 fs/read_write.c:686 ksys_write+0x149/0x250 fs/read_write.c:738 do_syscall_x64 arch/x86/entry/syscall_64.c:63 [inline] do_syscall_64+0xf3/0x3d0 arch/x86/entry/syscall_64.c:94 entry_SYSCALL_64_after_hwframe+0x77/0x7f The root cause is in the corrupted image, there is a dnode has the same node id w/ its inode, so during f2fs_get_dnode_of_data(), it tries to access block address in dnode at offset 934, however it parses the dnode as inode node, so that get_dnode_addr() returns 360, then it tries to access page address from 360 + 934 * 4 = 4096 w/ 4 bytes. To fix this issue, let's add sanity check for node id of all direct nodes during f2fs_get_dnode_of_data().

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38715	HIGH	7.1	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: hfs: fix slab-out-of-bounds in hfs_bnode_read() This patch introduces is_bnode_offset_valid() method that checks the requested offset value. Also, it introduces check_and_correct_requested_length() method that checks and correct the requested length (if it is necessary). These methods are used in hfs_bnode_read(), hfs_bnode_write(), hfs_bnode_clear(), hfs_bnode_copy(), and hfs_bnode_move() with the goal to prevent the access out of allocated memory and triggering the crash.
CVE-2025-38714	HIGH	7.1	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: hfsplus: fix slab-out-of-bounds in hfsplus_bnode_read() The hfsplus_bnode_read() method can trigger the issue: [174.852007][T9784] ========== [174.852709][T9784] BUG: KASAN: slab-out-of-bounds in hfsplus_bnode_read+0x2f4/0x360 [174.853412][T9784] Read of size 8 at addr ffff88810b5fc6c0 by task repro/9784 [174.854059][T9784] [174.854272][T9784] CPU: 1 UID: 0 PID: 9784 Comm: repro Not tainted 6.16.0-rc3 #7 PREEMPT(full) [174.854281][T9784] Hardware name: QEMU Ubuntu 24.04 PC (i440FX + PIIX, 1996), BIOS 1.16.3-debian-1.16.3-2 04/01/2014 [174.854286][T9784] Call Trace: [174.854289][T9784] [174.854292][T9784] dump_stack_lvl+0x10e/0x1f0 [174.854305][T9784] print_report+0xd0/0x660 [174.854315][T9784] ? __virt_addr_valid+0x81/0x610 [174.854323][T9784] ? __phys_addr+0xe8/0x180 [174.854330][T9784] ? hfsplus_bnode_read+0x2f4/0x360 [174.854337][T9784] kasan_report+0xc6/0x100 [174.854346][T9784] ? hfsplus_bnode_read+0x2f4/0x360 [174.854354][T9784] hfsplus_bnode_read+0x2f4/0x360 [174.854362][T9784] hfsplus_bnode_dump+0x2ec/0x380 [174.854370][T9784] ? __pfx_hfsplus_bnode_dump+0x10/0x10 [174.854377][T9784] ? hfsplus_bnode_write_u16+0x83/0xb0 [174.854385][T9784] ? srcu_gp_start+0xd0/0x310 [174.854393][T9784] ? __mark_inode_dirty+0x29e/0xe40 [174.854402][T9784] hfsplus_brec_remove+0x3d2/0x4e0 [174.854411][T9784] __hfsplus_delete_attr+0x290/0x3a0 [174.854419][T9784] ? __pfx_hfs_find_1st_rec_by_cnid+0x10/0x10 [174.854427][T9784] ? __pfx___hfsplus_delete_attr+0x10/0x10 [174.854436][T9784] ? __asan_memset+0x23/0x50 [174.854450][T9784] hfsplus_delete_all_attrs+0x262/0x320 [174.854459][T9784] ? __pfx_hfsplus_delete_all_attrs+0x10/0x10 [174.854469][T9784] ? rcu_is_watching+0x12/0xc0 [174.854476][T9784] ? __mark_inode_dirty+0x29e/0xe40 [174.854483][T9784] hfsplus_delete_cat+0x845/0xde0 [174.854493][T9784] ? __pfx_hfsplus_delete_cat+0x10/0x10 [174.854507][T9784] hfsplus_unlink+0x1ca/0x7c0 [174.854516][T9784] ? __pfx_hfsplus_unlink+0x10/0x10 [174.854525][T9784] ? down_write+0x148/0x200 [174.854532][T9784] ? __pfx_down_write+0x10/0x10 [174.854540][T9784] vfs_unlink+0x2fe/0x9b0 [174.854549][T9784] do_unlinkat+0x490/0x670 [174.854557][T9784] ? __pfx_do_unlinkat+0x10/0x10 [174.854565][T9784] ? __might_fault+0xbc/0x130 [174.854576][T9784] ? getname_flags.part.0+0x1c5/0x550 [174.854584][T9784] __x64_sys_unlink+0xc5/0x110 [174.854592][T9784] do_syscall_64+0xc9/0x480 [174.854600][T9784]

CVE	Severity	CVSS	CVSS Vector	Component	Description
					entry_SYSCALL_64_after_hwframe+0x77/0x7f [174.854608][T9784] RIP: 0033:0x7f6fdf4c3167 [174.854614][T9784] Code: f0 ff ff 73 01 c3 48 8b 0d 26 0d 0e 00 f7 d8 64 89 01 48 83 c8 ff c3 66 2e 0f 1f 84 00 00 00 00 08 [174.854622][T9784] RSP: 002b:00007ffcb948bca8 EFLAGS: 00000206 ORIG_RAX: 0000000000000057 [174.854630][T9784] RAX: ffffffffffffffda RBX: 0000000000000000 RCX: 00007f6fdf4c3167 [174.854636][T9784] RDX: 00007ffcb948bcc0 RSI: 00007ffcb948bcc0 RDI: 00007ffcb948bd50 [174.854641][T9784] RBP: 00007ffcb948cd90 R08: 0000000000000001 R09: 00007ffcb948bb40 [174.854645][T9784] R10: 00007f6fdf564fc0 R11: 0000000000000206 R12: 0000561e1bc9c2d0 [174.854650][T9784] R13: 0000000000000000 R14: 0000000000000000 R15: 0000000000000000 [174.854658][T9784] [174.854661][T9784] [174.879281][T9784] Allocated by task 9784: [174.879664][T9784] kasan_save_stack+0x20/0x40 [174.880082][T9784] kasan_save_track+0x14/0x30 [174.880500][T9784] __kasan_kmalloc+0xaa/0xb0 [174.880908][T9784] __kmalloc_noprof+0x205/0x550 [174.881337][T9784] __hfs_bnode_create+0x107/0x890 [174.881779] [T9784] hfsplus_bnode_find+0x2d0/0xd10 [174.882222][T9784] hfsplus_brec_find+0x2b0/0x520 [174.882659][T9784] hfsplus_delete_all_attrs+0x23b/0x3 ---truncated---
CVE-2025-38713	HIGH	7.1	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: hfsplus: fix slab-out-of-bounds read in hfsplus_uni2asc() The hfsplus_readdir() method is capable to crash by calling hfsplus_uni2asc(): [667.121659][T9805] == [667.122651][T9805] BUG: KASAN: slab-out-of-bounds in hfsplus_uni2asc+0x902/0xa10 [667.123627][T9805] Read of size 2 at addr ffff88802592f40c by task repro/9805 [667.124578][T9805] [667.124876][T9805] CPU: 3 UID: 0 PID: 9805 Comm: repro Not tainted 6.16.0-rc3 #1 PREEMPT(full) [667.124886][T9805] Hardware name: QEMU Ubuntu 24.04 PC (i440FX + PIIX, 1996), BIOS 1.16.3-debian-1.16.3-2 04/01/2014 [667.124890][T9805] Call Trace: [667.124893][T9805] [667.124896][T9805] dump_stack_lvl+0x10e/0x1f0 [667.124911][T9805] print_report+0xd0/0x660 [667.124920][T9805] ? __virt_addr_valid+0x81/0x610 [667.124928][T9805] ? __phys_addr+0xe8/0x180 [667.124934][T9805] ? hfsplus_uni2asc+0x902/0xa10 [667.124942][T9805] kasan_report+0xc6/0x100 [667.124950][T9805] ? hfsplus_uni2asc+0x902/0xa10 [667.124959][T9805] hfsplus_uni2asc+0x902/0xa10 [667.124966][T9805] ? hfsplus_bnode_read+0x14b/0x360 [667.124974][T9805] hfsplus_readdir+0x845/0xfc0 [667.124984][T9805] ? __pfx_hfsplus_readdir+0x10/0x10 [667.124994][T9805] ? stack_trace_save+0x8e/0xc0 [667.125008][T9805] ? iterate_dir+0x18b/0xb20 [667.125015][T9805] ? trace_lock_acquire+0x85/0xd0 [667.125022][T9805] ? lock_acquire+0x30/0x80 [667.125029][T9805] ? iterate_dir+0x18b/0xb20 [667.125037][T9805] ? down_read_killable+0x1ed/0x4c0 [667.125044][T9805] ? putname+0x154/0x1a0 [667.125051][T9805] ? __pfx_down_read_killable+0x10/0x10 [667.125058][T9805] ? apparmor_file_permission+0x239/0x3e0 [667.125069][T9805] iterate_dir+0x296/0xb20 [667.125076][T9805] __x64_sys_getdents64+0x13c/0x2c0 [667.125084][T9805] ? __pfx___x64_sys_getdents64+0x10/0x10 [667.125091][T9805] ? __x64_sys_openat+0x141/0x200 [667.125126][T9805] ? __pfx_filldir64+0x10/0x10 [667.125134] [T9805] ? do_user_addr_fault+0x7fe/0x12f0 [

CVE	Severity	CVSS	CVSS Vector	Component	Description
					667.125143][T9805] do_syscall_64+0xc9/0x480 [667.125151][T9805] entry_SYSCALL_64_after_hwframe+0x77/0x7f [667.125158][T9805] RIP: 0033:0x7fa8753b2fc9 [667.125164][T9805] Code: 00 c3 66 2e 0f 1f 84 00 00 00 00 00 0f 1f 44 00 00 48 89 f8 48 89 f7 48 89 d6 48 89 ca 4d 89 c2 48 [667.125172][T9805] RSP: 002b:00007ffe96f8e0f8 EFLAGS: 00000217 ORIG_RAX: 00000000000000d9 [667.125181][T9805] RAX: ffffffffffffffda RBX: 0000000000000000 RCX: 00007fa8753b2fc9 [667.125185][T9805] RDX: 0000000000000400 RSI: 00002000000063c0 RDI: 0000000000000004 [667.125190][T9805] RBP: 00007ffe96f8e110 R08: 00007ffe96f8e110 R09: 00007ffe96f8e110 [667.125195][T9805] R10: 0000000000000000 R11: 0000000000000217 R12: 0000556b1e3b4260 [667.125199][T9805] R13: 0000000000000000 R14: 0000000000000000 R15: 0000000000000000 [667.125207][T9805] [667.125210][T9805] [667.145632][T9805] Allocated by task 9805: [667.145991][T9805] kasan_save_stack+0x20/0x40 [667.146352][T9805] kasan_save_track+0x14/0x30 [667.146717][T9805] __kasan_kmalloc+0xaa/0xb0 [667.147065][T9805] __kmalloc_noprof+0x205/0x550 [667.147448][T9805] hfsplus_find_init+0x95/0x1f0 [667.147813][T9805] hfsplus_readdir+0x220/0xfc0 [667.148174][T9805] iterate_dir+0x296/0xb20 [667.148549][T9805] __x64_sys_getdents64+0x13c/0x2c0 [667.148937][T9805] do_syscall_64+0xc9/0x480 [667.149291][T9805] entry_SYSCALL_64_after_hwframe+0x77/0x7f [667.149809][T9805] [667.150030][T9805] The buggy address belongs to the object at ffff88802592f000 [667.150030][T9805] which belongs to the cache kmalloc-2k of size 2048 [667.151282][T9805] The buggy address is located 0 bytes to the right of [667.151282][T9805] allocated 1036-byte region [ffff88802592f000, ffff88802592f40c) [667.1 ---truncated---
CVE-2025-39853	HIGH	7.1	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: i40e: Fix potential invalid access when MAC list is empty list_first_entry() never returns NULL - if the list is empty, it still returns a pointer to an invalid object, leading to potential invalid memory access when dereferenced. Fix this by using list_first_entry_or_null instead of list_first_entry.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-39901	HIGH	7.1	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: i40e: remove read access to debugfs files The 'command' and 'netdev_ops' debugfs files are a legacy debugging interface supported by the i40e driver since its early days by commit 02e9c290814c ("i40e: debugfs interface"). Both of these debugfs files provide a read handler which is mostly useless, and which is implemented with questionable logic. They both use a static 256 byte buffer which is initialized to the empty string. In the case of the 'command' file this buffer is literally never used and simply wastes space. In the case of the 'netdev_ops' file, the last command written is saved here. On read, the files contents are presented as the name of the device followed by a colon and then the contents of their respective static buffer. For 'command' this will always be ": ". For 'netdev_ops', this will be ": ". But note the buffer is shared between all devices operated by this module. At best, it is mostly meaningless information, and at worse it could be accessed simultaneously as there doesn't appear to be any locking mechanism. We have also recently received multiple reports for both read functions about their use of snprintf and potential overflow that could result in reading arbitrary kernel memory. For the 'command' file, this is definitely impossible, since the static buffer is always zero and never written to. For the 'netdev_ops' file, it does appear to be possible, if the user carefully crafts the command input, it will be copied into the buffer, which could be large enough to cause snprintf to truncate, which then causes the copy_to_user to read beyond the length of the buffer allocated by kzalloc. A minimal fix would be to replace snprintf() with scnprintf() which would cap the return to the number of bytes written, preventing an overflow. A more involved fix would be to drop the mostly useless static buffers, saving 512 bytes and modifying the read functions to stop needing those as input. Instead, lets just completely drop the read access to these files. These are debug interfaces exposed as part of debugfs, and I don't believe that dropping read access will break any script, as the provided output is pretty useless. You can find the netdev name through other more standard interfaces, and the 'netdev_ops' interface can easily result in garbage if you issue simultaneous writes to multiple devices at once. In order to properly remove the i40e_dbg_netdev_ops_buf, we need to refactor its write function to avoid using the static buffer. Instead, use the same logic as the i40e_dbg_command_write, with an allocated buffer. Update the code to use this instead of the static buffer, and ensure we free the buffer on exit. This fixes simultaneous writes to 'netdev_ops' on multiple devices, and allows us to remove the now unused static buffer along with removing the read access.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-22104	HIGH	7.1	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ibmvnic: Use kernel helpers for hex dumps Previously, when the driver was printing hex dumps, the buffer was cast to an 8 byte long and printed using string formatters. If the buffer size was not a multiple of 8 then a read buffer overflow was possible. Therefore, create a new ibmvnic function that loops over a buffer and calls hex_dump_to_buffer instead. This patch address KASAN reports like the one below: ibmvnic 30000003 env3: Login Buffer: ibmvnic 30000003 env3: 01000000af000000 <...> ibmvnic 30000003 env3: 2e6d62692e736261 ibmvnic 30000003 env3: 65050003006d6f63 == BUG: KASAN: slab-out-of-bounds in ibmvnic_login+0xacc/0xffc [ibmvnic] Read of size 8 at addr c0000001331a9aa8 by task ip/17681 <...> Allocated by task 17681: <...> ibmvnic_login+0x2f0/0xffc [ibmvnic] ibmvnic_open+0x148/0x308 [ibmvnic] __dev_open+0x1ac/0x304 <...> The buggy address is located 168 bytes inside of allocated 175-byte region [c0000001331a9a00, c0000001331a9aaf) <...> == ibmvnic 30000003 env3: 000000000033766e
CVE-2025-39719	HIGH	7.1	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: iio: imu: bno055: fix OOB access of hw_xlate array Fix a potential out-of-bounds array access of the hw_xlate array in bno055.c. In bno055_get_regmask(), hw_xlate was iterated over the length of the vals array instead of the length of the hw_xlate array. In the case of bno055_gyr_scale, the vals array is larger than the hw_xlate array, so this could result in an out-of-bounds access. In practice, this shouldn't happen though because a match should always be found which breaks out of the for loop before it iterates beyond the end of the hw_xlate array. By adding a new hw_xlate_len field to the bno055_sysfs_attr, we can be sure we are iterating over the correct length.
CVE-2025-38204	HIGH	7.1	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: jfs: fix array-index-out-of-bounds read in add_missing_indices stbl is s8 but it must contain offsets into slot which can go from 0 to 127. Added a bound check for that error and return -EIO if the check fails. Also make jfs_readdir return with error if add_missing_indices returns with an error.
CVE-2024-26954	HIGH	7.1	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ksmbd: fix slab-out-of-bounds in smb_strndup_from_utf16() If ->NameOffset of smb2_create_req is smaller than Buffer offset of smb2_create_req, slab-out-of-bounds read can happen from smb2_open. This patch set the minimum value of the name offset to the buffer offset to validate name length of smb2_create_req().
CVE-2025-39943	HIGH	7.1	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ksmbd: smbdirect: validate data_offset and data_length field of smb_direct_data_transfer If data_offset and data_length of smb_direct_data_transfer struct are invalid, out of bounds issue could happen. This patch validate data_offset and data_length field in recv_done.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-71116	HIGH	7.1	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: libceph: make decode_pool() more resilient against corrupted osdmaps If the osdmap is (maliciously) corrupted such that the encoded length of ceph_pg_pool envelope is less than what is expected for a particular encoding version, out-of-bounds reads may ensue because the only bounds check that is there is based on that length value. This patch adds explicit bounds checks for each field that is decoded or skipped.
CVE-2026-22984	HIGH	7.1	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: libceph: prevent potential out-of-bounds reads in handle_auth_done() Perform an explicit bounds check on payload_len to avoid a possible out-of-bounds access in the callout. [idryomov: changelog]
CVE-2025-38445	HIGH	7.1	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: md/raid1: Fix stack memory use after return in raid1_reshape In the raid1_reshape function, newpool is allocated on the stack and assigned to conf->r1bio_pool. This results in conf->r1bio_pool.wait.head pointing to a stack address. Accessing this address later can lead to a kernel panic. Example access path: raid1_reshape() { // newpool is on the stack mempool_t newpool, oldpool; // initialize newpool.wait.head to stack address mempool_init(&newpool, ...); conf->r1bio_pool = newpool; } raid1_read_request() or raid1_write_request() { alloc_r1bio() { mempool_alloc() { // if pool->alloc fails remove_element() { --pool->curr_nr; } } } } mempool_free() { if (pool->curr_nr < pool->min_nr) { // pool->wait.head is a stack address // wake_up() will try to access this invalid address // which leads to a kernel panic return; wake_up(&pool->wait); } } Fix: reinit conf->r1bio_pool.wait after assigning newpool.
CVE-2025-71136	HIGH	7.1	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: media: adv7842: Avoid possible out-of-bounds array accesses in adv7842_cp_log_status() It's possible for cp_read() and hdmi_read() to return -EIO. Those values are further used as indexes for accessing arrays. Fix that by checking return values where it's needed. Found by Linux Verification Center (linuxtesting.org) with SVACE.
CVE-2025-38680	HIGH	7.1	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: media: uvcvideo: Fix 1-byte out-of-bounds read in uvc_parse_format() The buffer length check before calling uvc_parse_format() only ensured that the buffer has at least 3 bytes (buflen > 2), buf the function accesses buffer[3], requiring at least 4 bytes. This can lead to an out-of-bounds read if the buffer has exactly 3 bytes. Fix it by checking that the buffer has at least 4 bytes in uvc_parse_format().
CVE-2025-39710	HIGH	7.1	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: media: venus: Add a check for packet size after reading from shared memory Add a check to ensure that the packet size does not exceed the number of available words after reading the packet header from shared memory. This ensures that the size provided by the firmware is safe to process and prevent potential out-of-bounds memory access.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38679	HIGH	7.1	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: media: venus: Fix OOB read due to missing payload bound check Currently, The event_seq_changed() handler processes a variable number of properties sent by the firmware. The number of properties is indicated by the firmware and used to iterate over the payload. However, the payload size is not being validated against the actual message length. This can lead to out-of-bounds memory access if the firmware provides a property count that exceeds the data available in the payload. Such a condition can result in kernel crashes or potential information leaks if memory beyond the buffer is accessed. Fix this by properly validating the remaining size of the payload before each property access and updating bounds accordingly as properties are parsed. This ensures that property parsing is safely bounded within the received message buffer and protects against malformed or malicious firmware behavior.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-39883	HIGH	7.1	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: mm/memory-failure: fix VM_BUG_ON_PAGE(PagePoisoned(page)) when unpoison memory When I did memory failure tests, below panic occurs: page dumped because: VM_BUG_ON_PAGE(PagePoisoned(page)) kernel BUG at include/linux/page-flags.h:616! Oops: invalid opcode: 0000 [#1] PREEMPT SMP NOPTI CPU: 3 PID: 720 Comm: bash Not tainted 6.10.0-rc1-00195-g148743902568 #40 RIP: 0010:unpoison_memory+0x2f3/0x590 RSP: 0018:ffffa57fc8787d60 EFLAGS: 00000246 RAX: 0000000000000037 RBX: 0000000000000009 RCX: ffff9be25fcdc9c8 RDX: 0000000000000000 RSI: 0000000000000027 RDI: ffff9be25fcdc9c0 RBP: 0000000000300000 R08: ffffffffb4956f88 R09: 00000000000009ffb R10: 0000000000000284 R11: ffffffffb4926fa0 R12: ffffe6b00c000000 R13: ffff9bdb453dfd00 R14: 0000000000000000 R15: fffffffffffffffe FS: 00007f08f04e4740(0000) GS:ffff9be25fcc0000(0000) knlGS:0000000000000000 CS: 0010 DS: 0000 ES: 0000 CR0: 0000000080050033 CR2: 0000564787a30410 CR3: 000000010d4e2000 CR4: 00000000000006f0 Call Trace: unpoison_memory+0x2f3/0x590 simple_attr_write_xsigned.constprop.0.isra.0+0xb3/0x110 debugfs_attr_write+0x42/0x60 full_proxy_write+0x5b/0x80 vfs_write+0xd5/0x540 ksys_write+0x64/0xe0 do_syscall_64+0xb9/0x1d0 entry_SYSCALL_64_after_hwframe+0x77/0x7f RIP: 0033:0x7f08f0314887 RSP: 002b:00007ffece710078 EFLAGS: 00000246 ORIG_RAX: 0000000000000001 RAX: ffffffffffffffda RBX: 0000000000000009 RCX: 00007f08f0314887 RDX: 0000000000000009 RSI: 0000564787a30410 RDI: 0000000000000001 RBP: 0000564787a30410 R08: 000000000000fefe R09: 000000007fffffff R10: 0000000000000000 R11: 0000000000000246 R12: 0000000000000009 R13: 00007f08f041b780 R14: 00007f08f0417600 R15: 00007f08f0416a00 Modules linked in: hwpoison_inject ---[end trace 0000000000000000]--- RIP: 0010:unpoison_memory+0x2f3/0x590 RSP: 0018:ffffa57fc8787d60 EFLAGS: 00000246 RAX: 0000000000000037 RBX: 0000000000000009 RCX: ffff9be25fcdc9c8 RDX: 0000000000000000 RSI: 0000000000000027 RDI: ffff9be25fcdc9c0 RBP: 0000000000300000 R08: ffffffffb4956f88 R09: 00000000000009ffb R10: 0000000000000284 R11: ffffffffb4926fa0 R12: ffffe6b00c000000 R13: ffff9bdb453dfd00 R14: 0000000000000000 R15: fffffffffffffffe FS: 00007f08f04e4740(0000) GS:ffff9be25fcc0000(0000) knlGS:0000000000000000 CS: 0010 DS: 0000 ES: 0000 CR0: 0000000080050033 CR2: 0000564787a30410 CR3: 000000010d4e2000 CR4: 00000000000006f0 Kernel panic - not syncing: Fatal exception Kernel Offset: 0x31c00000 from 0xffffffff81000000 (relocation range: 0xffffffff80000000-0xffffffffbfffffff) ---[end Kernel panic - not syncing: Fatal exception]--- The root cause is that unpoison_memory() tries to check the PG_HWPoison flags of an uninitialized page. So VM_BUG_ON_PAGE(PagePoisoned(page)) is triggered. This can be reproduced by below steps: 1.Offline memory block: echo offline > /sys/devices/system/memory/memory12/state 2.Get offlined memory pfn: page-types -b n -rlN 3.Write pfn to unpoison-pfn echo > /sys/kernel/debug/hwpoison/unpoison-pfn This scenario can be identified by pfn_to_online_page() returning NULL. And ZONE_DEVICE pages are never expected, so we can simply fail if pfn_to_online_page() == NULL to fix the bug.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2026-23204	HIGH	7.1	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net/sched: cls_u32: use skb_header_pointer_careful() skb_header_pointer() does not fully validate negative @offset values. Use skb_header_pointer_careful() instead. GangMin Kim provided a report and a repro fooling u32_classify(): BUG: KASAN: slab-out-of-bounds in u32_classify+0x1180/0x11b0 net/sched/cls_u32.c:221
CVE-2025-22107	HIGH	7.1	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net: dsa: sja1105: fix kasan out-of-bounds warning in sja1105_table_delete_entry() There are actually 2 problems: - deleting the last element doesn't require the memmove of elements [i + 1, end) over it. Actually, element i+1 is out of bounds. - The memmove itself should move size - i - 1 elements, because the last element is out of bounds. The out-of-bounds element still remains out of bounds after being accessed, so the problem is only that we touch it, not that it becomes in active use. But I suppose it can lead to issues if the out-of-bounds element is part of an unmapped page.
CVE-2025-71112	HIGH	7.1	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net: hns3: add VLAN id validation before using Currently, the VLAN id may be used without validation when receive a VLAN configuration mailbox from VF. The length of vlan_del_fail_bmap is BITS_TO_LONGS(VLAN_N_VID). It may cause out-of-bounds memory access once the VLAN id is bigger than or equal to VLAN_N_VID. Therefore, VLAN id needs to be checked to ensure it is within the range of VLAN_N_VID.
CVE-2025-71101	HIGH	7.1	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: platform/x86: hp-bioscfg: Fix out-of-bounds array access in ACPI package parsing The hp_populate_*_elements_from_package() functions in the hp-bioscfg driver contain out-of-bounds array access vulnerabilities. These functions parse ACPI packages into internal data structures using a for loop with index variable 'elem' that iterates through enum_obj/integer_obj/order_obj/password_obj/string_obj arrays. When processing multi-element fields like PREREQUISITES and ENUM_POSSIBLE_VALUES, these functions read multiple consecutive array elements using expressions like 'enum_obj[elem + reqs]' and 'enum_obj[elem + pos_values]' within nested loops. The bug is that the bounds check only validated elem, but did not consider the additional offset when accessing elem + reqs or elem + pos_values. The fix changes the bounds check to validate the actual accessed index.
CVE-2026-23187	HIGH	7.1	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: pmdomain: imx8m-blk-ctrl: fix out-of-range access of bc->domains Fix out-of-range access of bc->domains in imx8m_blk_ctrl_remove().

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-39744	HIGH	7.1	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: rcu: Fix rcu_read_unlock() deadloop due to IRQ work During rcu_read_unlock_special(), if this happens during irq_exit(), we can lockup if an IPI is issued. This is because the IPI itself triggers the irq_exit() path causing a recursive lock up. This is precisely what Xiongfeng found when invoking a BPF program on the trace_tick_stop() tracepoint As shown in the trace below. Fix by managing the irq_work state correctly. irq_exit() __irq_exit_rcu() /* in_hardirq() returns false after this */ preempt_count_sub(HARDIRQ_OFFSET) tick_irq_exit() tick_nohz_irq_exit() tick_nohz_stop_sched_tick() trace_tick_stop() /* a bpf prog is hooked on this trace point */ __bpf_trace_tick_stop() bpf_trace_run2() rcu_read_unlock_special() /* will send a IPI to itself */ irq_work_queue_on(&rdp->defer_qs_iw, rdp->cpu); A simple reproducer can also be obtained by doing the following in tick_irq_exit(). It will hang on boot without the patch: static inline void tick_irq_exit(void) { + rcu_read_lock(); + WRITE_ONCE(current->rcu_read_unlock_special.b.need_qs, true); + rcu_read_unlock(); + [neeraj: Apply Frederic's suggested fix for PREEMPT_RT]
CVE-2025-38395	HIGH	7.1	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: regulator: gpio: Fix the out-of-bounds access to drvdata::gpiods drvdata::gpiods is supposed to hold an array of 'gpio_desc' pointers. But the memory is allocated for only one pointer. This will lead to out-of-bounds access later in the code if 'config::ngpios' is > 1. So fix the code to allocate enough memory to hold 'config::ngpios' of GPIO descriptors. While at it, also move the check for memory allocation failure to be below the allocation to make it more readable.
CVE-2025-38636	HIGH	7.1	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: rv: Use strings in da monitors tracepoints Using DA monitors tracepoints with KASAN enabled triggers the following warning: BUG: KASAN: global-out-of-bounds in do_trace_event_raw_event_event_da_monitor+0xd6/0x1a0 Read of size 32 at addr ffffffffaada8980 by task ... Call Trace: [...] do_trace_event_raw_event_event_da_monitor+0xd6/0x1a0 ? __pfx_do_trace_event_raw_event_event_da_monitor+0x10/0x10 ? trace_event_sncid+0x83/0x200 trace_event_sncid+0x163/0x200 [...] The buggy address belongs to the variable: automaton_snep+0x4e0/0x5e0 This is caused by the tracepoints reading 32 bytes __array instead of __string from the automata definition. Such strings are literals and reading 32 bytes ends up in out of bound memory accesses (e.g. the next automaton's data in this case). The error is harmless as, while printing the string, we stop at the null terminator, but it should still be fixed. Use the __string facilities while defining the tracepoints to avoid reading out of bound memory.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38728	HIGH	7.1	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: smb3: fix for slab out of bounds on mount to ksmbd With KASAN enabled, it is possible to get a slab out of bounds during mount to ksmbd due to missing check in parse_server_interfaces() (see below): BUG: KASAN: slab-out-of-bounds in parse_server_interfaces+0x14ee/0x1880 [cifs] Read of size 4 at addr ffff8881433dba98 by task mount/9827 CPU: 5 UID: 0 PID: 9827 Comm: mount Tainted: G OE 6.16.0-rc2-kasan #2 PREEMPT(voluntary) Tainted: [O]=OOT_MODULE, [E]=UNSIGNED_MODULE Hardware name: Dell Inc. Precision Tower 3620/0MWYPT, BIOS 2.13.1 06/14/2019 Call Trace: dump_stack_lvl+0x9f/0xf0 print_report+0xd1/0x670 __virt_addr_valid+0x22c/0x430 ? parse_server_interfaces+0x14ee/0x1880 [cifs] ? kasan_complete_mode_report_info+0x2a/0x1f0 ? parse_server_interfaces+0x14ee/0x1880 [cifs] kasan_report+0xd6/0x110 parse_server_interfaces+0x14ee/0x1880 [cifs] __asan_report_load_n_noabort+0x13/0x20 parse_server_interfaces+0x14ee/0x1880 [cifs] ? __pfx_parse_server_interfaces+0x10/0x10 [cifs] ? trace_hardirqs_on+0x51/0x60 SMB3_request_interfaces+0x1ad/0x3f0 [cifs] ? __pfx_SMB3_request_interfaces+0x10/0x10 [cifs] ? SMB2_tcon+0x23c/0x15d0 [cifs] smb3_qfs_tcon+0x173/0x2b0 [cifs] ? __pfx_smb3_qfs_tcon+0x10/0x10 [cifs] ? cifs_get_tcon+0x105d/0x2120 [cifs] ? do_raw_spin_unlock+0x5d/0x200 ? cifs_get_tcon+0x105d/0x2120 [cifs] ? __pfx_smb3_qfs_tcon+0x10/0x10 [cifs] cifs_mount_get_tcon+0x369/0xb90 [cifs] ? dfs_cache_find+0xe7/0x150 [cifs] dfs_mount_share+0x985/0x2970 [cifs] ? check_path.constprop.0+0x28/0x50 ? save_trace+0x54/0x370 ? __pfx_dfs_mount_share+0x10/0x10 [cifs] ? __lock_acquire+0xb82/0x2ba0 ? __kasan_check_write+0x18/0x20 cifs_mount+0xbc/0x9e0 [cifs] ? __pfx_cifs_mount+0x10/0x10 [cifs] ? do_raw_spin_unlock+0x5d/0x200 ? cifs_setup_cifs_sb+0x29d/0x810 [cifs] cifs_smb3_do_mount+0x263/0x1990 [cifs]
CVE-2025-38616	HIGH	7.1	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: tls: handle data disappearing from under the TLS ULP TLS expects that it owns the receive queue of the TCP socket. This cannot be guaranteed in case the reader of the TCP socket entered before the TLS ULP was installed, or uses some non-standard read API (eg. zerocopy ones). Replace the WARN_ON() and a buggy early exit (which leaves anchor pointing to a freed skb) with real error handling. Wipe the parsing state and tell the reader to retry. We already reload the anchor every time we (re)acquire the socket lock, so the only condition we need to avoid is an out of bounds read (not having enough bytes in the socket for previously parsed record len). If some data was read from under TLS but there's enough in the queue we'll reload and decrypt what is most likely not a valid TLS record. Leading to some undefined behavior from TLS perspective (corrupting a stream? missing an alert? missing an attack?) but no kernel crash should take place.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-39683	HIGH	7.1	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: tracing: Limit access to parser->buffer when trace_get_user failed When the length of the string written to set_ftrace_filter exceeds FTRACE_BUFF_MAX, the following KASAN alarm will be triggered: BUG: KASAN: slab-out-of-bounds in strsep+0x18c/0x1b0 Read of size 1 at addr ffff0000d00bd5ba by task ash/165 CPU: 1 UID: 0 PID: 165 Comm: ash Not tainted 6.16.0-g6bcdbd62bd56-dirty Hardware name: linux,dummy-virt (DT) Call trace: show_stack+0x34/0x50 (C) dump_stack_lvl+0xa0/0x158 print_address_description.constprop.0+0x88/0x398 print_report+0xb0/0x280 kasan_report+0xa4/0xf0 __asan_report_load1_noabort+0x20/0x30 strsep+0x18c/0x1b0 ftrace_process_regex.isra.0+0x100/0x2d8 ftrace_regex_release+0x484/0x618 __fput+0x364/0xa58 ____fput+0x28/0x40 task_work_run+0x154/0x278 do_notify_resume+0x1f0/0x220 el0_svc+0xec/0xf0 el0t_64_sync_handler+0xa0/0xe8 el0t_64_sync+0x1ac/0x1b0 The reason is that trace_get_user will fail when processing a string longer than FTRACE_BUFF_MAX, but not set the end of parser->buffer to 0. Then an OOB access will be triggered in ftrace_regex_release-> ftrace_process_regex->strsep->strpbrk. We can solve this problem by limiting access to parser->buffer when trace_get_user failed.
CVE-2025-39760	HIGH	7.1	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: usb: core: config: Prevent OOB read in SS endpoint companion parsing usb_parse_ss_endpoint_companion() checks descriptor type before length, enabling a potentially odd read outside of the buffer size. Fix this up by checking the size first before looking at any of the fields in the descriptor.
CVE-2025-38497	HIGH	7.1	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: usb: gadget: configfs: Fix OOB read on empty string write When writing an empty string to either 'qw_sign' or 'landingPage' sysfs attributes, the store functions attempt to access page[l - 1] before validating that the length 'l' is greater than zero. This patch fixes the vulnerability by adding a check at the beginning of os_desc_qw_sign_store() and webusb_landingPage_store() to handle the zero-length input case gracefully by returning immediately.
CVE-2025-39750	HIGH	7.1	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: wifi: ath12k: Correct tid cleanup when tid setup fails Currently, if any error occurs during ath12k_dp_rx_peer_tid_setup(), the tid value is already incremented, even though the corresponding TID is not actually allocated. Proceed to ath12k_dp_rx_peer_tid_delete() starting from unallocated tid, which might leads to freeing unallocated TID and cause potential crash or out-of-bounds access. Hence, fix by correctly decrementing tid before cleanup to match only the successfully allocated TIDs. Also, remove tid-- from failure case of ath12k_dp_rx_peer_frag_setup(), as decrementing the tid before cleanup in loop will take care of this. Compile tested only.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-39761	HIGH	7.1	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: wifi: ath12k: Decrement TID on RX peer frag setup error handling Currently, TID is not decremented before peer cleanup, during error handling path of ath12k_dp_rx_peer_frag_setup(). This could lead to out-of-bounds access in peer->rx_tid[]. Hence, add a decrement operation for TID, before peer cleanup to ensures proper cleanup and prevents out-of-bounds access issues when the RX peer frag setup fails. Found during code review. Compile tested only.
CVE-2024-58015	HIGH	7.1	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: wifi: ath12k: Fix for out-of bound access error Selfgen stats are placed in a buffer using print_array_to_buf_index() function. Array length parameter passed to the function is too big, resulting in possible out-of bound memory error. Decreasing buffer size by one fixes faulty upper bound of passed array. Discovered in coverity scan, CID 1600742 and CID 1600758
CVE-2025-38292	HIGH	7.1	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: wifi: ath12k: fix invalid access to memory In ath12k_dp_rx_msdu_coalesce(), rxcb is fetched from skb and boolean is_continuation is part of rxcb. Currently, after freeing the skb, the rxcb->is_continuation accessed again which is wrong since the memory is already freed. This might lead use-after-free error. Hence, fix by locally defining bool is_continuation from rxcb, so that after freeing skb, is_continuation can be used. Compile tested only.
CVE-2024-57982	HIGH	7.1	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: xfrm: state: fix out-of-bounds read during lookup lookup and resize can run in parallel. The xfrm_state_hash_generation seqlock ensures a retry, but the hash functions can observe a hmask value that is too large for the new hlist array. rehash does: rcu_assign_pointer(net->xfrm.state_bydst, ndst) [..] net->xfrm.state_hmask = nhashmask; While state lookup does: h = xfrm_dst_hash(net, daddr, saddr, tmpl->reqid, encap_family); hlist_for_each_entry_rcu(x, net->xfrm.state_bydst + h, bydst) { This is only safe in case the update to state_bydst is larger than net->xfrm.xfrm_state_hmask (or if the lookup function gets serialized via state spinlock again). Fix this by prefetching state_hmask and the associated pointers. The xfrm_state_hash_generation seqlock retry will ensure that the pointer and the hmask will be consistent. The existing helpers, like xfrm_dst_hash(), are now unsafe for RCU side, add lockdep assertions to document that they are only safe for insert side. xfrm_state_lookup_byaddr() uses the spinlock rather than RCU. AFAICS this is an oversight from back when state lookup was converted to RCU, this lock should be replaced with RCU in a future patch.
CVE-2025-22039	HIGH	7.1	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ksmbd: fix overflow in dacloffset bounds check The dacloffset field was originally typed as int and used in an unchecked addition, which could overflow and bypass the existing bounds check in both smb_check_perm_dacl() and smb_inherit_dacl(). This could result in out-of-bounds memory access and a kernel crash when dereferencing the DACL pointer. This patch converts dacloffset to unsigned int and uses check_add_overflow() to validate access to the DACL.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38670	HIGH	7.1	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: arm64/entry: Mask DAIF in cpu_switch_to(), call_on_irq_stack() `cpu_switch_to()` and `call_on_irq_stack()` manipulate SP to change to different stacks along with the Shadow Call Stack if it is enabled. Those two stack changes cannot be done atomically and both functions can be interrupted by SErrors or Debug Exceptions which, though unlikely, is very much broken : if interrupted, we can end up with mismatched stacks and Shadow Call Stack leading to clobbered stacks. In `cpu_switch_to()`, it can happen when SP_EL0 points to the new task, but x18 stills points to the old task's SCS. When the interrupt handler tries to save the task's SCS pointer, it will save the old task SCS pointer (x18) into the new task struct (pointed to by SP_EL0), clobbering it. In `call_on_irq_stack()`, it can happen when switching from the task stack to the IRQ stack and when switching back. In both cases, we can be interrupted when the SCS pointer points to the IRQ SCS, but SP points to the task stack. The nested interrupt handler pushes its return addresses on the IRQ SCS. It then detects that SP points to the task stack, calls `call_on_irq_stack()` and clobbers the task SCS pointer with the IRQ SCS pointer, which it will also use ! This leads to tasks returning to addresses on the wrong SCS, or even on the IRQ SCS, triggering kernel panics via CONFIG_VMAP_STACK or FPAC if enabled. This is possible on a default config, but unlikely. However, when enabling CONFIG_ARM64_PSEUDO_NMI, DAIF is unmasked and instead the GIC is responsible for filtering what interrupts the CPU should receive based on priority. Given the goal of emulating NMIs, pseudo-NMIs can be received by the CPU even in `cpu_switch_to()` and `call_on_irq_stack()`, possibly *very* frequently depending on the system configuration and workload, leading to unpredictable kernel panics. Completely mask DAIF in `cpu_switch_to()` and restore it when returning. Do the same in `call_on_irq_stack()`, but restore and mask around the branch. Mask DAIF even if CONFIG_SHADOW_CALL_STACK is not enabled for consistency of behaviour between all configurations. Introduce and use an assembly macro for saving and masking DAIF, as the existing one saves but only masks IF.
CVE-2024-53147	HIGH	7.1	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: exfat: fix out-of-bounds access of directory entries In the case of the directory size is greater than or equal to the cluster size, if start_clu becomes an EOF cluster(an invalid cluster) due to file system corruption, then the directory entry where ei->hint_femp.eidx hint is outside the directory, resulting in an out-of-bounds access, which may cause further file system corruption. This commit adds a check for start_clu, if it is an invalid cluster, the file or directory will be treated as empty.
CVE-2025-38501	HIGH	7.5	CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ksmbd: limit repeated connections from clients with the same IP Repeated connections from clients with the same IP address may exhaust the max connections and prevent other normal client connections. This patch limit repeated connections from clients with the same IP.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2026-22998	HIGH	7.5	CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: nvme-tcp: fix NULL pointer dereferences in nvmet_tcp_build_pdu_iovec Commit efa56305908b ("nvmet-tcp: Fix a kernel panic when host sends an invalid H2C PDU length") added ttag bounds checking and data_offset validation in nvmet_tcp_handle_h2c_data_pdu(), but it did not validate whether the command's data structures (cmd->req.sg and cmd->iov) have been properly initialized before processing H2C_DATA PDUs. The nvmet_tcp_build_pdu_iovec() function dereferences these pointers without NULL checks. This can be triggered by sending H2C_DATA PDU immediately after the ICREQ/ICRESP handshake, before sending a CONNECT command or NVMe write command. Attack vectors that trigger NULL pointer dereferences: 1. H2C_DATA PDU sent before CONNECT → both pointers NULL 2. H2C_DATA PDU for READ command → cmd->req.sg allocated, cmd->iov NULL 3. H2C_DATA PDU for uninitialized command slot → both pointers NULL The fix validates both cmd->req.sg and cmd->iov before calling nvmet_tcp_build_pdu_iovec(). Both checks are required because: - Uninitialized commands: both NULL - READ commands: cmd->req.sg allocated, cmd->iov NULL - WRITE commands: both allocated
CVE-2025-38566	HIGH	7.5	CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: sunrpc: fix handling of server side tls alerts Scott Mayhew discovered a security exploit in NFS over TLS in tls_alert_recv() due to its assumption it can read data from the msg iterator's kvec.. kTLS implementation splits TLS non-data record payload between the control message buffer (which includes the type such as TLS aler or TLS cipher change) and the rest of the payload (say TLS alert's level/description) which goes into the msg payload buffer. This patch proposes to rework how control messages are setup and used by sock_recvmsg(). If no control message structure is setup, kTLS layer will read and process TLS data record types. As soon as it encounters a TLS control message, it would return an error. At that point, NFS can setup a kvec backed msg buffer and read in the control message such as a TLS alert. Msg iterator can advance the kvec pointer as a part of the copy process thus we need to revert the iterator before calling into the tls_alert_recv.
CVE-2025-39701	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ACPI: pfr_update: Fix the driver update version check The security-version-number check should be used rather than the runtime version check for driver updates. Otherwise, the firmware update would fail when the update binary had a lower runtime version number than the current one. [rjw: Changelog edits]
CVE-2025-71082	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: Bluetooth: btusb: revert use of devm_kzalloc in btusb This reverts commit 98921dbd00c4e ("Bluetooth: Use devm_kzalloc in btusb.c file"). In btusb_probe(), we use devm_kzalloc() to allocate the btusb data. This ties the lifetime of all the btusb data to the binding of a driver to one interface, INTF. In a driver that binds to other interfaces, ISOC and DIAG, this is an accident waiting to happen. The issue is revealed in btusb_disconnect(), where calling usb_driver_release_interface(&btusb_driver, data->intf) will have devm free the data that is also being used by the other interfaces of the driver that may not be released yet. To fix this, revert the use of devm and go back to freeing memory explicitly.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38494	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: HID: core: do not bypass hid_hw_raw_request hid_hw_raw_request() is actually useful to ensure the provided buffer and length are valid. Directly calling in the low level transport driver function bypassed those checks and allowed invalid paramto be used.
CVE-2025-39730	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: NFS: Fix filehandle bounds checking in nfs_fh_to_dentry() The function needs to check the minimal filehandle length before it can access the embedded filehandle.
CVE-2025-39828	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: atm: atmtcp: Prevent arbitrary write in atmtcp_recv_control(). syzbot reported the splat below. [0] When atmtcp_v_open() or atmtcp_v_close() is called via connect() or close(), atmtcp_send_control() is called to send an in-kernel special message. The message has ATMTCP_HDR_MAGIC in atmtcp_control.hdr.length. Also, a pointer of struct atm_vcc is set to atmtcp_control.vcc. The notable thing is struct atmtcp_control is uAPI but has a space for an in-kernel pointer. struct atmtcp_control { struct atmtcp_hdr hdr; /* must be first */ ... atm_kptr_t vcc; /* both directions */ ... } __ATM_API_ALIGN; typedef struct { unsigned char _[8]; } __ATM_API_ALIGN atm_kptr_t; The special message is processed in atmtcp_recv_control() called from atmtcp_c_send(). atmtcp_c_send() is vcc->dev->ops->send() and called from 2 paths: 1. .ndo_start_xmit() (vcc->send() == atm_send_aal0()) 2. vcc_sendmsg() The problem is sendmsg() does not validate the message length and userspace can abuse atmtcp_recv_control() to overwrite any kptr by atmtcp_control. Let's add a new ->pre_send() hook to validate messages from sendmsg(). [0]: Oops: general protection fault, probably for non-canonical address 0xdfffffc00200000ab: 0000 [#1] SMP KASAN PTI KASAN: probably user-memory-access in range [0x0000000100000558-0x000000010000055f] CPU: 0 UID: 0 PID: 5865 Comm: syz-executor331 Not tainted 6.17.0-rc1-syzkaller-00215-gbab3ce404553 #0 PREEMPT(full) Hardware name: Google Google Compute Engine/Google Compute Engine, BIOS Google 07/12/2025 RIP: 0010:atmtcp_recv_control drivers/atm/atmtcp.c:93 [inline] RIP: 0010:atmtcp_c_send+0x1da/0x950 drivers/atm/atmtcp.c:297 Code: 4d 8d 75 1a 4c 89 f0 48 c1 e8 03 42 0f b6 04 20 84 c0 0f 85 15 06 00 00 41 0f b7 1e 4d 8d b7 60 05 00 00 4c 89 f0 48 c1 e8 03 <42> 0f b6 04 20 84 c0 0f 85 13 06 00 00 66 41 89 1e 4d 8d 75 1c 4c RSP: 0018:ffffc90003f5f810 EFLAGS: 00010203 RAX: 00000000200000ab RBX: 0000000000000000 RCX: 0000000000000000 RDX: ffff88802a510000 RSI: 00000000ffffffff RDI: ffff888030a6068c RBP: ffff88802699fb40 R08: ffff888030a606eb R09: 1ffff1100614c0dd R10: dffffc0000000000 R11: ffffffff8718fc40 R12: dffffc0000000000 R13: ffff888030a60680 R14: 000000010000055f R15: 00000000ffffffff FS: 00007f8d7e9236c0(0000) GS:ffff888125c1c000(0000) knlGS:0000000000000000 CS: 0010 DS: 0000 ES: 0000 CR0: 0000000080050033 CR2: 000000000045ad50 CR3: 0000000075bde000 CR4: 00000000003526f0 Call Trace: vcc_sendmsg+0xa10/0xc60 net/atm/common.c:645 sock_sendmsg_nosec net/socket.c:714 [inline] __sock_sendmsg+0x219/0x270 net/socket.c:729 ____sys_sendmsg+0x505/0x830 net/socket.c:2614 ___sys_sendmsg+0x21f/0x2a0 net/socket.c:2668 __sys_sendmsg net/socket.c:2700 [inline] __do_sys_sendmsg net/socket.c:2705 [inline] __se_sys_sendmsg net/socket.c:2703 [inline]

CVE	Severity	CVSS	CVSS Vector	Component	Description
					__x64_sys_sendmsg+0x19b/0x260 net/socket.c:2703 do_syscall_x64 arch/x86/entry/syscall_64.c:63 [inline] do_syscall_64+0xfa/0x3b0 arch/x86/entry/syscall_64.c:94 entry_SYSCALL_64_after_hwframe+0x77/0x7f RIP: 0033:0x7f8d7e96a4a9 Code: 28 00 00 00 75 05 48 83 c4 28 c3 e8 51 18 00 00 90 48 89 f8 48 89 f7 48 89 d6 48 89 ca 4d 89 c2 4d 89 c8 4c 8b 4c 24 08 0f 05 <48> 3d 01 f0 ff ff 73 01 c3 48 c7 c1 b0 ff ff ff f7 d8 64 89 01 48 RSP: 002b:00007f8d7e923198 EFLAGS: 00000246 ORIG_RAX: 000000000000002e RAX: ffffffffffffffda RBX: 00007f8d7e9f4308 RCX: 00007f8d7e96a4a9 RDX: 0000000000000000 RSI: 0000200000000240 RDI: 0000000000000005 RBP: 00007f8d7e9f4300 R08: 65732f636f72702f R09: 65732f636f72702f R10: 65732f636f72702f R11: 0000000000000246 R12: 00007f8d7e9c10ac R13: 00007f8d7e9231a0 R14: 0000200000000200 R15: 0000200000000250 Modules linked in:

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38279	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: bpf: Do not include stack ptr register in precision backtracking bookkeeping Yi Lai reported an issue ([1]) where the following warning appears in kernel dmesg: [60.643604] verifier backtracking bug [60.643635] WARNING: CPU: 10 PID: 2315 at kernel/bpf/verifier.c:4302 __mark_chain_precision+0x3a6c/0x3e10 [60.648428] Modules linked in: bpf_testmod(OE) [60.650471] CPU: 10 UID: 0 PID: 2315 Comm: test_progs Tainted: G OE 6.15.0-rc4-gef11287f8289-dirty #327 PREEMPT(full) [60.654385] Tainted: [O]=OOT_MODULE, [E]=UNSIGNED_MODULE [60.656682] Hardware name: QEMU Standard PC (i440FX + PIIX, 1996), BIOS rel-1.14.0-0-g155821a1990b-prebuilt.qemu.org 04/01/2014 [60.660475] RIP: 0010:__mark_chain_precision+0x3a6c/0x3e10 [60.662814] Code: 5a 30 84 89 ea e8 c4 d9 01 00 80 3d 3e 7d d8 04 00 0f 85 60 fa ff ff c6 05 31 7d d8 04 01 48 c7 c7 00 58 30 84 e8 c4 06 a5 ff <0f> 0b e9 46 fa ff ff 48 ... [60.668720] RSP: 0018:ffff888116cc7298 EFLAGS: 00010246 [60.671075] RAX: 54d70e82dfd31900 RBX: ffff888115b65e20 RCX: 0000000000000000 [60.673659] RDX: 0000000000000001 RSI: 0000000000000004 RDI: 00000000ffffffff [60.676241] RBP: 0000000000000400 R08: ffff8881f6f23bd3 R09: 1ffff1103ede477a [60.678787] R10: dfffffc0000000000 R11: ffffed103ede477b R12: ffff888115b60ae8 [60.681420] R13: 1ffff11022b6cbc4 R14: 00000000ffffffff2 R15: 0000000000000001 [60.684030] FS: 00007fc2aedd80c0(0000) GS:ffff88826fa8a000(0000) knlGS:0000000000000000 [60.686837] CS: 0010 DS: 0000 ES: 0000 CR0: 0000000080050033 [60.689027] CR2: 000056325369e000 CR3: 000000011088b002 CR4: 0000000000370ef0 [60.691623] Call Trace: [60.692821] [60.693960] ? __pfx_verbose+0x10/0x10 [60.695656] ? __pfx_disasm_kfunc_name+0x10/0x10 [60.697495] check_cond_jmp_op+0x16f7/0x39b0 [60.699237] do_check+0x58fa/0xab10 ... Further analysis shows the warning is at line 4302 as below: 4294 /* static subprog call instruction, which 4295 * means that we are exiting current subprog, 4296 * so only r1-r5 could be still requested as 4297 * precise, r0 and r6-r10 or any stack slot in 4298 * the current frame should be zero by now 4299 */ 4300 if (bt_reg_mask(bt) & ~BPF_REGMASK_ARGS) { 4301 verbose(env, "BUG regs %x\n", bt_reg_mask(bt)); 4302 WARN_ONCE(1, "verifier backtracking bug"); 4303 return -EFAULT; 4304 } With the below test (also in the next patch): __used __naked static void __bpf_jmp_r10(void) { asm volatile ("r2 = 2314885393468386424 ll;" "goto +0;" "if r2 <= r10 goto +3;" "if r1 >= -1835016 goto +0;" "if r2 <= 8 goto +0;" "if r3 <= 0 goto +0;" "exit;" ::: __clobber_all); } SEC("?raw_tp") __naked void bpf_jmp_r10(void) { asm volatile ("r3 = 0 ll;" "call __bpf_jmp_r10;" "r0 = 0;" "exit;" ::: __clobber_all); } The following is the verifier failure log: 0: (18) r3 = 0x0 ; R3_w=0 2: (85) call pc+2 caller: R10=fp0 callee: frame1: R1=ctx() R3_w=0 R10=fp0 5: frame1: R1=ctx() R3_w=0 R10=fp0 ; asm volatile (" \ @ verifier_precision.c:184 5: (18) r2 = 0x20202000256c6c78 ; frame1: R2_w=0x20202000256c6c78 7: (05) goto pc+0 8: (bd) if r2 <= r10 goto pc+3 ; frame1: R2_w=0x20202000256c6c78 R10=fp0 9: (35) if r1 >= 0xffe3fff8 goto pc+0 ; frame1: R1=ctx() 10: (b5) if r2 <= 0x8 goto pc+0 mark_precise: frame1: last_idx 10 first_idx 0 subseq_idx -1 mark_precise: frame1: regs=r2 stack= before 9: (35) if r1 >= 0xffe3fff8 goto pc+0 mark_precise: frame1: regs=r2 stack= before 8: (bd) if r2 <= r10 goto pc+3 mark_preci ---truncated---

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-39738	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: btrfs: do not allow relocation of partially dropped subvolumes [BUG] There is an internal report that balance triggered transaction abort, with the following call trace: item 85 key (594509824 169 0) itemoff 12599 itemsize 33 extent refs 1 gen 197740 flags 2 ref#0: tree block backref root 7 item 86 key (594558976 169 0) itemoff 12566 itemsize 33 extent refs 1 gen 197522 flags 2 ref#0: tree block backref root 7 ... BTRFS error (device loop0): extent item not found for insert, bytenr 594526208 num_bytes 16384 parent 449921024 root_objectid 934 owner 1 offset 0 BTRFS error (device loop0): failed to run delayed ref for logical 594526208 num_bytes 16384 type 182 action 1 ref_mod 1: -117 ----------[cut here]---------- BTRFS: Transaction aborted (error -117) WARNING: CPU: 1 PID: 6963 at ../fs/btrfs/extent-tree.c:2168 btrfs_run_delayed_refs+0xfa/0x110 [btrfs] And btrfs check doesn't report anything wrong related to the extent tree. [CAUSE] The cause is a little complex, firstly the extent tree indeed doesn't have the backref for 594526208. The extent tree only have the following two backrefs around that bytenr on-disk: item 65 key (594509824 METADATA_ITEM 0) itemoff 13880 itemsize 33 refs 1 gen 197740 flags TREE_BLOCK tree block skinny level 0 (176 0x7) tree block backref root CSUM_TREE item 66 key (594558976 METADATA_ITEM 0) itemoff 13847 itemsize 33 refs 1 gen 197522 flags TREE_BLOCK tree block skinny level 0 (176 0x7) tree block backref root CSUM_TREE But the such missing backref item is not an corruption on disk, as the offending delayed ref belongs to subvolume 934, and that subvolume is being dropped: item 0 key (934 ROOT_ITEM 198229) itemoff 15844 itemsize 439 generation 198229 root_dirid 256 bytenr 10741039104 byte_limit 0 bytes_used 345571328 last_snapshot 198229 flags 0x1000000000001(RDONLY) refs 0 drop_progress key (206324 EXTENT_DATA 2711650304) drop_level 2 level 2 generation_v2 198229 And that offending tree block 594526208 is inside the dropped range of that subvolume. That explains why there is no backref item for that bytenr and why btrfs check is not reporting anything wrong. But this also shows another problem, as btrfs will do all the orphan subvolume cleanup at a read-write mount. So half-dropped subvolume should not exist after an RW mount, and balance itself is also exclusive to subvolume cleanup, meaning we shouldn't hit a subvolume half-dropped during relocation. The root cause is, there is no orphan item for this subvolume. In fact there are 5 subvolumes from around 2021 that have the same problem. It looks like the original report has some older kernels running, and caused those zombie subvolumes. Thankfully upstream commit 8d488a8c7ba2 ("btrfs: fix subvolume/snapshot deletion not triggered on mount") has long fixed the bug. [ENHANCEMENT] For repairing such old fs, btrfs-progs will be enhanced. Considering how delayed the problem will show up (at run delayed ref time) and at that time we have to abort transaction already, it is too late. Instead here we reject any half-dropped subvolume for reloc tree at the earliest time, preventing confusion and extra time wasted on debugging similar bugs.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-39686	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: comedi: Make insn_rw_emulate_bits() do insn->n samples The `insn_rw_emulate_bits()` function is used as a default handler for `INSN_READ` instructions for subdevices that have a handler for `INSN_BITS` but not for `INSN_READ`. Similarly, it is used as a default handler for `INSN_WRITE` instructions for subdevices that have a handler for `INSN_BITS` but not for `INSN_WRITE`. It works by emulating the `INSN_READ` or `INSN_WRITE` instruction handling with a constructed `INSN_BITS` instruction. However, `INSN_READ` and `INSN_WRITE` instructions are supposed to be able read or write multiple samples, indicated by the `insn->n` value, but `insn_rw_emulate_bits()` currently only handles a single sample. For `INSN_READ`, the comedi core will copy `insn->n` samples back to user-space. (That triggered KASAN kernel-infoleak errors when `insn->n` was greater than 1, but that is being fixed more generally elsewhere in the comedi core.) Make `insn_rw_emulate_bits()` either handle `insn->n` samples, or return an error, to conform to the general expectation for `INSN_READ` and `INSN_WRITE` handlers.
CVE-2025-38361	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: drm/amd/display: Check dce_hwseq before dereferencing it [WHAT] hws was checked for null earlier in dce110_blank_stream, indicating hws can be null, and should be checked whenever it is used. (cherry picked from commit 79db43611ff61280b6de58ce1305e0b2ecf675ad)
CVE-2025-38389	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: drm/i915/gt: Fix timeline left held on VMA alloc error The following error has been reported sporadically by CI when a test unbinds the i915 driver on a ring submission platform: <4> [239.330153] ------------[cut here]------------ <4> [239.330166] i915 0000:00:02.0: [drm] drm_WARN_ON(dev_priv->mm.shrink_count) <4> [239.330196] WARNING: CPU: 1 PID: 18570 at drivers/gpu/drm/i915/i915_gem.c:1309 i915_gem_cleanup_early+0x13e/0x150 [i915] ... <4> [239.330640] RIP: 0010:i915_gem_cleanup_early+0x13e/0x150 [i915] ... <4> [239.330942] Call Trace: <4> [239.330944] <4> [239.330949] i915_driver_late_release+0x2b/0xa0 [i915] <4> [239.331202] i915_driver_release+0x86/0xa0 [i915] <4> [239.331482] devm_drm_dev_init_release+0x61/0x90 <4> [239.331494] devm_action_release+0x15/0x30 <4> [239.331504] release_nodes+0x3d/0x120 <4> [239.331517] devres_release_all+0x96/0xd0 <4> [239.331533] device_unbind_cleanup+0x12/0x80 <4> [239.331543] device_release_driver_internal+0x23a/0x280 <4> [239.331550] ? bus_find_device+0xa5/0xe0 <4> [239.331563] device_driver_detach+0x14/0x20 ... <4> [357.719679] ---[end trace 0000000000000000]--- If the test also unloads the i915 module then that's followed with: <3> [357.787478] ========== = <3> [357.788006] BUG i915_vma (Tainted: G U W N): Objects remaining on __kmem_cache_shutdown() <3> [357.788031] -------- <3> [357.788204] Object 0xffff888109e7f480 @offset=29824 <3> [357.788670] Allocated in i915_vma_instance+0xee/0xc10 [i915] age=292729 cpu=4 pid=2244 <4> [357.788994] i915_vma_instance+0xee/0xc10 [i915] <4> [357.789290] init_status_page+0x7b/0x420 [i915] <4> [357.789532] intel_engines_init+0x1d8/0x980 [i915] <4> [357.789772] intel_gt_init+0x175/0x450 [i915]

CVE	Severity	CVSS	CVSS Vector	Component	Description
					<4> [357.790014] i915_gem_init+0x113/0x340 [i915] <4> [357.790281] i915_driver_probe+0x847/0xed0 [i915] <4> [357.790504] i915_pci_probe+0xe6/0x220 [i915] ... Closer analysis of CI results history has revealed a dependency of the error on a few IGT tests, namely: - igt@api_intel_allocator@fork-simple-stress-signal, - igt@api_intel_allocator@two-level-inception-interruptible, - igt@gem_linear_blits@interruptible, - igt@prime_mmap_coherency@ioctl-errors, which invisibly trigger the issue, then exhibited with first driver unbind attempt. All of the above tests perform actions which are actively interrupted with signals. Further debugging has allowed to narrow that scope down to DRM_IOCTL_I915_GEM_EXECBUFFER2, and ring_context_alloc(), specific to ring submission, in particular. If successful then that function, or its execlists or GuC submission equivalent, is supposed to be called only once per GEM context engine, followed by raise of a flag that prevents the function from being called again. The function is expected to unwind its internal errors itself, so it may be safely called once more after it returns an error. In case of ring submission, the function first gets a reference to the engine's legacy timeline and then allocates a VMA. If the VMA allocation fails, e.g. when i915_vma_instance() called from inside is interrupted with a signal, then ring_context_alloc() fails, leaving the timeline held referenced. On next I915_GEM_EXECBUFFER2 IOCTL, another reference to the timeline is got, and only that last one is put on successful completion. As a consequence, the legacy timeline, with its underlying engine status page's VMA object, is still held and not released on driver unbind. Get the legacy timeline only after successful allocation of the context engine's VMA. v2: Add a note on other submission methods (Krzysztof Karas): Both execlists and GuC submission use lrc_alloc() which seems free from a similar issue. (cherry picked from commit cc43422b3cc79eacff4c5a8ba0d224688ca9dd4f)
CVE-2026-23156	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: efivarfs: fix error propagation in efivar_entry_get() efivar_entry_get() always returns success even if the underlying __efivar_entry_get() fails, masking errors. This may result in uninitialized heap memory being copied to userspace in the efivarfs_file_read() path. Fix it by returning the error from __efivar_entry_get().
CVE-2026-23083	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: fou: Don't allow 0 for FOU_ATTR_IPPROTO. fou_udp_recv() has the same problem mentioned in the previous patch. If FOU_ATTR_IPPROTO is set to 0, skb is not freed by fou_udp_recv() nor "resubmit"-ted in ip_protocol_deliver_rcu(). Let's forbid 0 for FOU_ATTR_IPPROTO.
CVE-2025-38707	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: fs/ntfs3: Add sanity check for file name The length of the file name should be smaller than the directory entry size.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38396	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: fs: export anon_inode_make_secure_inode() and fix secretmem LSM bypass Export anon_inode_make_secure_inode() to allow KVM guest_memfd to create anonymous inodes with proper security context. This replaces the current pattern of calling alloc_anon_inode() followed by inode_init_security_anon() for creating security context manually. This change also fixes a security regression in secretmem where the S_PRIVATE flag was not cleared after alloc_anon_inode(), causing LSM/SELinux checks to be bypassed for secretmem file descriptors. As guest_memfd currently resides in the KVM module, we need to export this symbol for use outside the core kernel. In the future, guest_memfd might be moved to core-mm, at which point the symbols no longer would have to be exported. When/if that happens is still unclear.
CVE-2025-38548	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: hwmon: (corsair-cpro) Validate the size of the received input buffer Add buffer_recv_size to store the size of the received bytes. Validate buffer_recv_size in send_usb_cmd().
CVE-2025-39911	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: i40e: fix IRQ freeing in i40e_vsi_request_irq_msix error path If request_irq() in i40e_vsi_request_irq_msix() fails in an iteration later than the first, the error path wants to free the IRQs requested so far. However, it uses the wrong dev_id argument for free_irq(), so it does not free the IRQs correctly and instead triggers the warning: Trying to free already-free IRQ 173 WARNING: CPU: 25 PID: 1091 at kernel/irq/manage.c:1829 __free_irq+0x192/0x2c0 Modules linked in: i40e(+) [...] CPU: 25 UID: 0 PID: 1091 Comm: NetworkManager Not tainted 6.17.0-rc1+ #1 PREEMPT(lazy) Hardware name: [...] RIP: 0010:__free_irq+0x192/0x2c0 [...] Call Trace: free_irq+0x32/0x70 i40e_vsi_request_irq_msix.cold+0x63/0x8b [i40e] i40e_vsi_request_irq+0x79/0x80 [i40e] i40e_vsi_open+0x21f/0x2f0 [i40e] i40e_open+0x63/0x130 [i40e] __dev_open+0xfc/0x210 __dev_change_flags+0x1fc/0x240 netif_change_flags+0x27/0x70 do_setlink.isra.0+0x341/0xc70 rtnl_newlink+0x468/0x860 rtnetlink_rcv_msg+0x375/0x450 netlink_rcv_skb+0x5c/0x110 netlink_unicast+0x288/0x3c0 netlink_sendmsg+0x20d/0x430 ____sys_sendmsg+0x3a2/0x3d0 ___sys_sendmsg+0x99/0xe0 __sys_sendmsg+0x8a/0xf0 do_syscall_64+0x82/0x2c0 entry_SYSCALL_64_after_hwframe+0x76/0x7e [...] ---[end trace 0000000000000000]--- Use the same dev_id for free_irq() as for request_irq(). I tested this with inserting code to fail intentionally.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-71089	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: iommu: disable SVA when CONFIG_X86 is set Patch series "Fix stale IOTLB entries for kernel address space", v7. This proposes a fix for a security vulnerability related to IOMMU Shared Virtual Addressing (SVA). In an SVA context, an IOMMU can cache kernel page table entries. When a kernel page table page is freed and reallocated for another purpose, the IOMMU might still hold stale, incorrect entries. This can be exploited to cause a use-after-free or write-after-free condition, potentially leading to privilege escalation or data corruption. This solution introduces a deferred freeing mechanism for kernel page table pages, which provides a safe window to notify the IOMMU to invalidate its caches before the page is reused. This patch (of 8): In the IOMMU Shared Virtual Addressing (SVA) context, the IOMMU hardware shares and walks the CPU's page tables. The x86 architecture maps the kernel's virtual address space into the upper portion of every process's page table. Consequently, in an SVA context, the IOMMU hardware can walk and cache kernel page table entries. The Linux kernel currently lacks a notification mechanism for kernel page table changes, specifically when page table pages are freed and reused. The IOMMU driver is only notified of changes to user virtual address mappings. This can cause the IOMMU's internal caches to retain stale entries for kernel VA. Use-After-Free (UAF) and Write-After-Free (WAF) conditions arise when kernel page table pages are freed and later reallocated. The IOMMU could misinterpret the new data as valid page table entries. The IOMMU might then walk into attacker-controlled memory, leading to arbitrary physical memory DMA access or privilege escalation. This is also a Write-After-Free issue, as the IOMMU will potentially continue to write Accessed and Dirty bits to the freed memory while attempting to walk the stale page tables. Currently, SVA contexts are unprivileged and cannot access kernel mappings. However, the IOMMU will still walk kernel-only page tables all the way down to the leaf entries, where it realizes the mapping is for the kernel and errors out. This means the IOMMU still caches these intermediate page table entries, making the described vulnerability a real concern. Disable SVA on x86 architecture until the IOMMU can receive notification to flush the paging cache before freeing the CPU kernel page table pages.
CVE-2025-71122	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: iommufd/selftest: Check for overflow in IOMMU_TEST_OP_ADD_RESERVED syzkaller found it could overflow math in the test infrastructure and cause a WARN_ON by corrupting the reserved interval tree. This only effects test kernels with CONFIG_IOMMUFD_TEST. Validate the user input length in the test ioctl.
CVE-2025-38688	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: iommufd: Prevent ALIGN() overflow When allocating IOVA the candidate range gets aligned to the target alignment. If the range is close to ULONG_MAX then the ALIGN() can wrap resulting in a corrupted iova. Open code the ALIGN() using get_add_overflow() to prevent this. This simplifies the checks as we don't need to check for length earlier either. Consolidate the two copies of this code under a single helper. This bug would allow userspace to create a mapping that overlaps with some other mapping or a reserved range.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38550	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ipv6: mcast: Delay put pmc->idev in mld_del_delrec() pmc->idev is still used in ip6_mc_clear_src(), so as mld_clear_delrec() does, the reference should be put after ip6_mc_clear_src() return.
CVE-2025-38572	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ipv6: reject malicious packets in ipv6_gso_segment() syzbot was able to craft a packet with very long IPv6 extension headers leading to an overflow of skb->transport_header. This 16bit field has a limited range. Add skb_reset_transport_header_careful() helper and use it from ipv6_gso_segment() WARNING: CPU: 0 PID: 5871 at ./include/linux/skbuff.h:3032 skb_reset_transport_header include/linux/skbuff.h:3032 [inline] WARNING: CPU: 0 PID: 5871 at ./include/linux/skbuff.h:3032 ipv6_gso_segment+0x15e2/0x21e0 net/ipv6/ip6_offload.c:151 Modules linked in: CPU: 0 UID: 0 PID: 5871 Comm: syz-executor211 Not tainted 6.16.0-rc6-syzkaller-g7abc678e3084 #0 PREEMPT(full) Hardware name: Google Google Compute Engine/Google Compute Engine, BIOS Google 07/12/2025 RIP: 0010:skb_reset_transport_header include/linux/skbuff.h:3032 [inline] RIP: 0010:ipv6_gso_segment+0x15e2/0x21e0 net/ipv6/ip6_offload.c:151 Call Trace: skb_mac_gso_segment+0x31c/0x640 net/core/gso.c:53 nsh_gso_segment+0x54a/0xe10 net/nsh/nsh.c:110 skb_mac_gso_segment+0x31c/0x640 net/core/gso.c:53 __skb_gso_segment+0x342/0x510 net/core/gso.c:124 skb_gso_segment include/net/gso.h:83 [inline] validate_xmit_skb+0x857/0x11b0 net/core/dev.c:3950 validate_xmit_skb_list+0x84/0x120 net/core/dev.c:4000 sch_direct_xmit+0xd3/0x4b0 net/sched/sch_generic.c:329 __dev_xmit_skb net/core/dev.c:4102 [inline] __dev_queue_xmit+0x17b6/0x3a70 net/core/dev.c:4679
CVE-2025-39743	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: jfs: truncate good inode pages when hard link is 0 The fileset value of the inode copy from the disk by the reproducer is AGGR_RESERVED_I. When executing evict, its hard link number is 0, so its inode pages are not truncated. This causes the bugon to be triggered when executing clear_inode() because nrpages is greater than 0.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2026-23025	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: mm/page_alloc: prevent pcp corruption with SMP=n The kernel test robot has reported: BUG: spinlock trylock failure on UP on CPU#0, kcompactd0/28 lock: 0xffff888807e35ef0, .magic: dead4ead, .owner: kcompactd0/28, .owner_cpu: 0 CPU: 0 UID: 0 PID: 28 Comm: kcompactd0 Not tainted 6.18.0-rc5-00127-ga06157804399 #1 PREEMPT 8cc09ef94dcec767faa911515ce9e609c45db470 Call Trace: __dump_stack (lib/dump_stack.c:95) dump_stack_lvl (lib/dump_stack.c:123) dump_stack (lib/dump_stack.c:130) spin_dump (kernel/locking/spinlock_debug.c:71) do_raw_spin_trylock (kernel/locking/spinlock_debug.c:?) _raw_spin_trylock (include/linux/spinlock_api_smp.h:89 kernel/locking/spinlock.c:138) __free_frozen_pages (mm/page_alloc.c:2973) ___free_pages (mm/page_alloc.c:5295) __free_pages (mm/page_alloc.c:5334) tlb_remove_table_rcu (include/linux/mm.h:? include/linux/mm.h:3122 include/asm-generic/tlb.h:220 mm/mmu_gather.c:227 mm/mmu_gather.c:290) ? __cfi_tlb_remove_table_rcu (mm/mmu_gather.c:289) ? rcu_core (kernel/rcu/tree.c:?) rcu_core (include/linux/rcupdate.h:341 kernel/rcu/tree.c:2607 kernel/rcu/tree.c:2861) rcu_core_si (kernel/rcu/tree.c:2879) handle_softirqs (arch/x86/include/asm/jump_label.h:36 include/trace/events/irq.h:142 kernel/softirq.c:623) __irq_exit_rcu (arch/x86/include/asm/jump_label.h:36 kernel/softirq.c:725) irq_exit_rcu (kernel/softirq.c:741) sysvec_apic_timer_interrupt (arch/x86/kernel/apic/apic.c:1052) RIP: 0010:_raw_spin_unlock_irqrestore (arch/x86/include/asm/preempt.h:95 include/linux/spinlock_api_smp.h:152 kernel/locking/spinlock.c:194) free_pcppages_bulk (mm/page_alloc.c:1494) drain_pages_zone (include/linux/spinlock.h:391 mm/page_alloc.c:2632) __drain_all_pages (mm/page_alloc.c:2731) drain_all_pages (mm/page_alloc.c:2747) kcompactd (mm/compaction.c:3115) kthread (kernel/kthread.c:465) ? __cfi_kcompactd (mm/compaction.c:3166) ? __cfi_kthread (kernel/kthread.c:412) ret_from_fork (arch/x86/kernel/process.c:164) ? __cfi_kthread (kernel/kthread.c:412) ret_from_fork_asm (arch/x86/entry/entry_64.S:255) Matthew has analyzed the report and identified that in drain_page_zone() we are in a section protected by spin_lock(&pcp->lock) and then get an interrupt that attempts spin_trylock() on the same lock. The code is designed to work this way without disabling IRQs and occasionally fail the trylock with a fallback. However, the SMP=n spinlock implementation assumes spin_trylock() will always succeed, and thus it's normally a no-op. Here the enabled lock debugging catches the problem, but otherwise it could cause a corruption of the pcp structure. The problem has been introduced by commit 574907741599 ("mm/page_alloc: leave IRQs enabled for per-cpu page allocations"). The pcp locking scheme recognizes the need for disabling IRQs to prevent nesting spin_trylock() sections on SMP=n, but the need to prevent the nesting in spin_lock() has not been recognized. Fix it by introducing local wrappers that change the spin_lock() to spin_lock_iqsave() with SMP=n and use them in all places that do spin_lock(&pcp->lock). [vbabka@suse.cz: add pcp_ prefix to the spin_lock_irqsave wrappers, per Steven]

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38552	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: mptcp: plug races between subflow fail and subflow creation We have races similar to the one addressed by the previous patch between subflow failing and additional subflow creation. They are just harder to trigger. The solution is similar. Use a separate flag to track the condition 'socket state prevent any additional subflow creation' protected by the fallback lock. The socket fallback makes such flag true, and also receiving or sending an MP_FAIL option. The field 'allow_infinite_fallback' is now always touched under the relevant lock, we can drop the ONCE annotation on write.
CVE-2025-39766	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net/sched: Make cake_enqueue return NET_XMIT_CN when past buffer_limit The following setup can trigger a WARNING in htb_activate due to the condition: !cl->leaf.q->q.qlen tc qdisc del dev lo root tc qdisc add dev lo root handle 1: htb default 1 tc class add dev lo parent 1: classid 1:1 \ htb rate 64bit tc qdisc add dev lo parent 1:1 handle f: \ cake memlimit 1b ping -I lo -f -c1 -s64 -W0.001 127.0.0.1 This is because the low memlimit leads to a low buffer_limit, which causes packet dropping. However, cake_enqueue still returns NET_XMIT_SUCCESS, causing htb_enqueue to call htb_activate with an empty child qdisc. We should return NET_XMIT_CN when packets are dropped from the same tin and flow. I do not believe return value of NET_XMIT_CN is necessary for packet drops in the case of ack filtering, as that is meant to optimize performance, not to signal congestion.
CVE-2025-38201	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: netfilter: nft_set_pipapo: clamp maximum map bucket size to INT_MAX Otherwise, it is possible to hit WARN_ON_ONCE in __kvmalloc_node_noprof() when resizing hashtable because __GFP_NOWARN is unset. Similar to: b541ba7d1f5a ("netfilter: conntrack: clamp maximum hashtable size to INT_MAX")
CVE-2025-38565	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: perf/core: Exit early on perf_mmap() fail When perf_mmap() fails to allocate a buffer, it still invokes the event_mapped() callback of the related event. On X86 this might increase the perf_rdpmc_allowed reference counter. But nothing undoes this as perf_mmap_close() is never called in this case, which causes another reference count leak. Return early on failure to prevent that.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38563	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: perf/core: Prevent VMA split of buffer mappings The perf mmap code is careful about mmap()'ing the user page with the ringbuffer and additionally the auxiliary buffer, when the event supports it. Once the first mapping is established, subsequent mapping have to use the same offset and the same size in both cases. The reference counting for the ringbuffer and the auxiliary buffer depends on this being correct. Though perf does not prevent that a related mapping is split via mmap(2), munmap(2) or mremap(2). A split of a VMA results in perf_mmap_open() calls, which take reference counts, but then the subsequent perf_mmap_close() calls are not longer fulfilling the offset and size checks. This leads to reference count leaks. As perf already has the requirement for subsequent mappings to match the initial mapping, the obvious consequence is that VMA splits, caused by resizing of a mapping or partial unmapping, have to be prevented. Implement the vm_operations_struct::may_split() callback and return unconditionally -EINVAL. That ensures that the mapping offsets and sizes cannot be changed after the fact. Remapping to a different fixed address with the same size is still possible as it takes the references for the new mapping and drops those of the old mapping.
CVE-2025-38535	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: phy: tegra: xusb: Fix unbalanced regulator disable in UTMI PHY mode When transitioning from USB_ROLE_DEVICE to USB_ROLE_NONE, the code assumed that the regulator should be disabled. However, if the regulator is marked as always-on, regulator_is_enabled() continues to return true, leading to an incorrect attempt to disable a regulator which is not enabled. This can result in warnings such as: [250.155624] WARNING: CPU: 1 PID: 7326 at drivers/regulator/core.c:3004 _regulator_disable+0xe4/0x1a0 [250.155652] unbalanced disables for VIN_SYS_5V0 To fix this, we move the regulator control logic into tegra186_xusb_padctl_id_override() function since it's directly related to the ID override state. The regulator is now only disabled when the role transitions from USB_ROLE_HOST to USB_ROLE_NONE, by checking the VBUS_ID register. This ensures that regulator enable/disable operations are properly balanced and only occur when actually transitioning to/from host mode.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-71078	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: powerpc/64s/slb: Fix SLB multihit issue during SLB preload On systems using the hash MMU, there is a software SLB preload cache that mirrors the entries loaded into the hardware SLB buffer. This preload cache is subject to periodic eviction — typically after every 256 context switches — to remove old entry. To optimize performance, the kernel skips switch_mmu_context() in switch_mm_irqs_off() when the prev and next mm_struct are the same. However, on hash MMU systems, this can lead to inconsistencies between the hardware SLB and the software preload cache. If an SLB entry for a process is evicted from the software cache on one CPU, and the same process later runs on another CPU without executing switch_mmu_context(), the hardware SLB may retain stale entries. If the kernel then attempts to reload that entry, it can trigger an SLB multi-hit error. The following timeline shows how stale SLB entries are created and can cause a multi-hit error when a process moves between CPUs without a MMU context switch. CPU 0 CPU 1 ------ ------ Process P exec swapper/1 load_elf_binary begin_new_exc activate_mm switch_mm_irqs_off switch_mmu_context switch_slb /* * This invalidates all * the entries in the HW * and setup the new HW * SLB entries as per the * preload cache. */ context_switch sched_migrate_task migrates process P to cpu-1 Process swapper/0 context switch (to process P) (uses mm_struct of Process P) switch_mm_irqs_off() switch_slb load_slb++ /* * load_slb becomes 0 here * and we evict an entry from * the preload cache with * preload_age(). We still * keep HW SLB and preload * cache in sync, that is * because all HW SLB entries * anyways gets evicted in * switch_slb during SLBIA. * We then only add those * entries back in HW SLB, * which are currently * present in preload_cache * (after eviction). */ load_elf_binary continues... setup_new_exec() slb_setup_new_exec() sched_switch event sched_migrate_task migrates process P to cpu-0 context_switch from swapper/0 to Process P switch_mm_irqs_off() /* * Since both prev and next mm struct are same we don't call * switch_mmu_context(). This will cause the HW SLB and SW preload * cache to go out of sync in preload_new_slb_context. Because there * was an SLB entry which was evicted from both HW and preload cache * on cpu-1. Now later in preload_new_slb_context(), when we will try * to add the same preload entry again, we will add this to the SW * preload cache and then will add it to the HW SLB. Since on cpu-0 * this entry was never invalidated, hence adding this entry to the HW * SLB will cause a SLB multi-hit error. */ load_elf_binary cont ---truncated---
CVE-2025-38704	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: rcu/nocb: Fix possible invalid rdp's->nocb_cb_kthread pointer access In the preparation stage of CPU online, if the corresponding the rdp's->nocb_cb_kthread does not exist, will be created, there is a situation where the rdp's rcuop kthreads creation fails, and then de-offload this CPU's rdp, does not assign this CPU's rdp->nocb_cb_kthread pointer, but this rdp's->nocb_gp_rdp and rdp's->rdp_gp->nocb_gp_kthread is still valid. This will cause the subsequent re-offload operation of this offline CPU, which will pass the conditional check and the kthread_unpark() will access invalid rdp's->nocb_cb_kthread pointer. This commit therefore use rdp's->nocb_gp_kthread instead of rdp_gp's->nocb_gp_kthread for safety check.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-37822	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: riscv: uprobes: Add missing fence.i after building the XOL buffer The XOL (execute out-of-line) buffer is used to single-step the replaced instruction(s) for uprobes. The RISC-V port was missing a proper fence.i (i$ flushing) after constructing the XOL buffer, which can result in incorrect execution of stale/broken instructions. This was found running the BPF selftests "test_progs: uprobe_autoattach, attach_probe" on the Spacemit K1/X60, where the uprobes tests randomly blew up.
CVE-2025-71220	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: smb/server: call ksmbd_session_rpc_close() on error path in create_smb2_pipe() When ksmbd_iov_pin_rsp() fails, we should call ksmbd_session_rpc_close().
CVE-2025-71068	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: svcrdma: bound check rq_pages index in inline path svc_rdma_copy_inline_range indexed rqstp->rq_pages[rc_curpage] without verifying rc_curpage stays within the allocated page array. Add guards before the first use and after advancing to a new page.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-39955	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: tcp: Clear tcp_sk(sk)->fastopen_rsk in tcp_disconnect(). syzbot reported the splat below where a socket had tcp_sk(sk)->fastopen_rsk in the TCP_ESTABLISHED state. [0] syzbot reused the server-side TCP Fast Open socket as a new client before the TFO socket completes 3WHS: 1. accept() 2. connect(AF_UNSPEC) 3. connect() to another destination As of accept(), sk->sk_state is TCP_SYN_RECV, and tcp_disconnect() changes it to TCP_CLOSE and makes connect() possible, which restarts timers. Since tcp_disconnect() forgot to clear tcp_sk(sk)->fastopen_rsk, the retransmit timer triggered the warning and the intended packet was not retransmitted. Let's call reqsk_fastopen_remove() in tcp_disconnect(). [0]: WARNING: CPU: 2 PID: 0 at net/ipv4/tcp_timer.c:542 tcp_retransmit_timer (net/ipv4/tcp_timer.c:542 (discriminator 7)) Modules linked in: CPU: 2 UID: 0 PID: 0 Comm: swapper/2 Not tainted 6.17.0-rc5-g201825fb4278 #62 PREEMPT(voluntary) Hardware name: QEMU Standard PC (i440FX + PIIX, 1996), BIOS 1.16.3-debian-1.16.3-2 04/01/2014 RIP: 0010:tcp_retransmit_timer (net/ipv4/tcp_timer.c:542 (discriminator 7)) Code: 41 55 41 54 55 53 48 8b af b8 08 00 00 48 89 fb 48 85 ed 0f 84 55 01 00 00 0f b6 47 12 3c 03 74 0c 0f b6 47 12 3c 04 74 04 90 <0f> 0b 90 48 8b 85 c0 00 00 00 48 89 ef 48 8b 40 30 e8 6a 4f 06 3e RSP: 0018:ffffc900002f8d40 EFLAGS: 00010293 RAX: 0000000000000002 RBX: ffff888106911400 RCX: 0000000000000017 RDX: 0000000002517619 RSI: ffffffff83764080 RDI: ffff888106911400 RBP: ffff888106d5c000 R08: 0000000000000001 R09: ffffc900002f8de8 R10: 00000000000000c2 R11: ffffc900002f8ff8 R12: ffff888106911540 R13: ffff888106911480 R14: ffff888106911840 R15: ffffc900002f8de0 FS: 0000000000000000(0000) GS:ffff88907b768000(0000) knlGS:0000000000000000 CS: 0010 DS: 0000 ES: 0000 CR0: 0000000080050033 CR2: 00007f8044d69d90 CR3: 0000000002c30003 CR4: 0000000000370ef0 Call Trace: tcp_write_timer (net/ipv4/tcp_timer.c:738) call_timer_fn (kernel/time/timer.c:1747) __run_timers (kernel/time/timer.c:1799 kernel/time/timer.c:2372) timer_expire_remote (kernel/time/timer.c:2385 kernel/time/timer.c:2376 kernel/time/timer.c:2135) tmigr_handle_remote_up (kernel/time/timer_migration.c:944 kernel/time/timer_migration.c:1035) __walk_groups.isra.0 (kernel/time/timer_migration.c:533 (discriminator 1)) tmigr_handle_remote (kernel/time/timer_migration.c:1096) handle_softirqs (./arch/x86/include/asm/jump_label.h:36 ./include/trace/events/irq.h:142 kernel/softirq.c:580) irq_exit_rcu (kernel/softirq.c:614 kernel/softirq.c:453 kernel/softirq.c:680 kernel/softirq.c:696) sysvec_apic_timer_interrupt (arch/x86/kernel/apic/apic.c:1050 (discriminator 35) arch/x86/kernel/apic/apic.c:1050 (discriminator 35))

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-39913	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: tcp_bpf: Call sk_msg_free() when tcp_bpf_send_verdict() fails to allocate psock->cork. syzbot reported the splat below. [0] The repro does the following: 1. Load a sk_msg prog that calls bpf_msg_cork_bytes(msg, cork_bytes) 2. Attach the prog to a SOCKMAP 3. Add a socket to the SOCKMAP 4. Activate fault injection 5. Send data less than cork_bytes At 5., the data is carried over to the next sendmsg() as it is smaller than the cork_bytes specified by bpf_msg_cork_bytes(). Then, tcp_bpf_send_verdict() tries to allocate psock->cork to hold the data, but this fails silently due to fault injection + __GFP_NOWARN. If the allocation fails, we need to revert the sk->sk_forward_alloc change done by sk_msg_alloc(). Let's call sk_msg_free() when tcp_bpf_send_verdict fails to allocate psock->cork. The "*copied" also needs to be updated such that a proper error can be returned to the caller, sendmsg. It fails to allocate psock->cork. Nothing has been corked so far, so this patch simply sets "*copied" to 0. [0]: WARNING: net/ipv4/af_inet.c:156 at inet_sock_destruct+0x623/0x730 net/ipv4/af_inet.c:156, CPU#1: syz-executor/5983 Modules linked in: CPU: 1 UID: 0 PID: 5983 Comm: syz-executor Not tainted syzkaller #0 PREEMPT(full) Hardware name: Google Google Compute Engine/Google Compute Engine, BIOS Google 07/12/2025 RIP: 0010:inet_sock_destruct+0x623/0x730 net/ipv4/af_inet.c:156 Code: 0f 0b 90 e9 62 fe ff ff e8 7a db b5 f7 90 0f 0b 90 e9 95 fe ff ff e8 6c db b5 f7 90 0f 0b 90 e9 bb fe ff ff e8 5e db b5 f7 90 <0f> 0b 90 e9 e1 fe ff ff 89 f9 80 e1 07 80 c1 03 38 c1 0f 8c 9f fc RSP: 0018:ffffc90000a08b48 EFLAGS: 00010246 RAX: ffffffff8a09d0b2 RBX: dfffffc0000000000 RCX: ffff888024a23c80 RDX: 0000000000000100 RSI: 0000000000000fff RDI: 0000000000000000 RBP: 0000000000000fff R08: ffff88807e07c627 R09: 1ffff1100fc0f8c4 R10: dfffffc0000000000 R11: ffffed100fc0f8c5 R12: ffff88807e07c380 R13: dfffffc0000000000 R14: ffff88807e07c60c R15: 1ffff1100fc0f872 FS: 00005555604c4500(0000) GS:ffff888125af1000(0000) knlGS:0000000000000000 CS: 0010 DS: 0000 ES: 0000 CR0: 0000000080050033 CR2: 00005555604df5c8 CR3: 0000000032b06000 CR4: 00000000003526f0 Call Trace: __sk_destruct+0x86/0x660 net/core/sock.c:2339 rcu_do_batch kernel/rcu/tree.c:2605 [inline] rcu_core+0xca8/0x1770 kernel/rcu/tree.c:2861 handle_softirqs+0x286/0x870 kernel/softirq.c:579 __do_softirq kernel/softirq.c:613 [inline] invoke_softirq kernel/softirq.c:453 [inline] __irq_exit_rcu+0xca/0x1f0 kernel/softirq.c:680 irq_exit_rcu+0x9/0x30 kernel/softirq.c:696 instr_sysvec_apic_timer_interrupt arch/x86/kernel/apic/apic.c:1052 [inline] sysvec_apic_timer_interrupt+0xa6/0xc0 arch/x86/kernel/apic/apic.c:1052
CVE-2025-71091	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: team: fix check for port enabled in team_queue_override_port_prio_changed() There has been a syzkaller bug reported recently with the following trace: list_del corruption, ffff888058bea080->prev is LIST_POISON2 (dead000000000122) ------------[cut here]------------ kernel BUG at lib/list_debug.c:59! Oops: invalid opcode: 0000 [#1] SMP KASAN NOPTI CPU: 3 UID: 0 PID: 21246 Comm: syz.0.2928 Not tainted syzkaller #0 PREEMPT(full) Hardware name: QEMU Standard PC (Q35 + ICH9, 2009), BIOS 1.16.3-debian-1.16.3-2~bpo12+1 04/01/2014 RIP: 0010:__list_del_entry_valid_or_report+0x13e/0x200 lib/list_debug.c:59 Code: 48 c7 c7 e0 71 f0 8b e8 30 08 ef fc 90 0f 0b 48 89 ef e8 a5 02 55 fd 48 89 ea 48 89 de 48 c7 c7 40 72 f0 8b e8 13 08 ef fc 90 <0f> 0b 48 89 ef e8 88 02 55 fd 48 89 ea 48 b8 00 00 00 00 00 fc ff RSP:

CVE	Severity	CVSS	CVSS Vector	Component	Description
					0018:ffffc9000d49f370 EFLAGS: 00010286 RAX: 000000000000004e RBX: ffff888058bea080 RCX: ffffc9002817d000 RDX: 0000000000000000 RSI: ffffffff819becc6 RDI: 0000000000000005 RBP: dead000000000122 R08: 0000000000000005 R09: 0000000000000000 R10: 0000000080000000 R11: 0000000000000001 R12: ffff888039e9c230 R13: ffff888058bea088 R14: ffff888058bea080 R15: ffff888055461480 FS: 00007fbbcfe6f6c0(0000) GS:ffff8880d6d0a000(0000) knlGS:0000000000000000 CS: 0010 DS: 0000 ES: 0000 CR0: 0000000080050033 CR2: 000000110c3afcb0 CR3: 00000000382c7000 CR4: 0000000000352ef0 Call Trace: __list_del_entry_valid include/linux/list.h:132 [inline] __list_del_entry include/linux/list.h:223 [inline] list_del_rcu include/linux/rculist.h:178 [inline] __team_queue_override_port_del drivers/net/team/team_core.c:826 [inline] __team_queue_override_port_del drivers/net/team/team_core.c:821 [inline] team_queue_override_port_prio_changed drivers/net/team/team_core.c:883 [inline] team_priority_option_set+0x171/0x2f0 drivers/net/team/team_core.c:1534 team_option_set drivers/net/team/team_core.c:376 [inline] team_nl_options_set_doit+0x8ae/0xe60 drivers/net/team/team_core.c:2653 genl_family_rcv_msg_doit+0x209/0x2f0 net/netlink/genetlink.c:1115 genl_family_rcv_msg net/netlink/genetlink.c:1195 [inline] genl_rcv_msg+0x55c/0x800 net/netlink/genetlink.c:1210 netlink_rcv_skb+0x158/0x420 net/netlink/af_netlink.c:2552 genl_rcv+0x28/0x40 net/netlink/genetlink.c:1219 netlink_unicast_kernel net/netlink/af_netlink.c:1320 [inline] netlink_unicast+0x5aa/0x870 net/netlink/af_netlink.c:1346 netlink_sendmsg+0x8c8/0xdd0 net/netlink/af_netlink.c:1896 sock_sendmsg_nosec net/socket.c:727 [inline] __sock_sendmsg net/socket.c:742 [inline] ____sys_sendmsg+0xa98/0xc70 net/socket.c:2630 ___sys_sendmsg+0x134/0x1d0 net/socket.c:2684 __sys_sendmsg+0x16d/0x220 net/socket.c:2716 do_syscall_x64 arch/x86/entry/syscall_64.c:63 [inline] do_syscall_64+0xcd/0xfa0 arch/x86/entry/syscall_64.c:94 entry_SYSCALL_64_after_hwframe+0x77/0x7f The problem is in this flow: 1) Port is enabled, queue_id != 0, in qom_list 2) Port gets disabled -> team_port_disable() -> team_queue_override_port_del() -> del (removed from list) 3) Port is disabled, queue_id != 0, not in any list 4) Priority changes -> team_queue_override_port_prio_changed() -> checks: port disabled && queue_id != 0 -> calls del - hits the BUG as it is removed already To fix this, change the check in team_queue_override_port_prio_changed() so it returns early if port is not enabled.
CVE-2025-38403	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: vsock/vmci: Clear the vmci transport packet properly when initializing it In vmci_transport_packet_init memset the vmci_transport_packet before populating the fields to avoid any uninitialised data being left in the structure.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-39957	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: wifi: mac80211: increase scan_ies_len for S1G Currently the S1G capability element is not taken into account for the scan_ies_len, which leads to a buffer length validation failure in ieee80211_prep_hw_scan() and subsequent WARN in __ieee80211_start_scan(). This prevents hw scanning from functioning. To fix ensure we accommodate for the S1G capability length.
CVE-2025-38512	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: wifi: prevent A-MSDU attacks in mesh networks This patch is a mitigation to prevent the A-MSDU spoofing vulnerability for mesh networks. The initial update to the IEEE 802.11 standard, in response to the FragAttacks, missed this case (CVE-2025-27558). It can be considered a variant of CVE-2020-24588 but for mesh networks. This patch tries to detect if a standard MSDU was turned into an A-MSDU by an adversary. This is done by parsing a received A-MSDU as a standard MSDU, calculating the length of the Mesh Control header, and seeing if the 6 bytes after this header equal the start of an rfc1042 header. If equal, this is a strong indication of an ongoing attack attempt. This defense was tested with mac80211_hwsim against a mesh network that uses an empty Mesh Address Extension field, i.e., when four addresses are used, and when using a 12-byte Mesh Address Extension field, i.e., when six addresses are used. Functionality of normal MSDUs and A-MSDUs was also tested, and confirmed working, when using both an empty and 12-byte Mesh Address Extension field. It was also tested with mac80211_hwsim that A-MSDU attacks in non-mesh networks keep being detected and prevented. Note that the vulnerability being patched, and the defense being implemented, was also discussed in the following paper and in the following IEEE 802.11 presentation: https://papers.mathyvanhoef.com/wisec2025.pdf https://mentor.ieee.org/802.11/dcn/25/11-25-0949-00-000m-a-msdu-mesh-spoof-protection.docx

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-39797	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: xfrm: Duplicate SPI Handling The issue originates when Strongswan initiates an XFRM_MSG_ALLOCSPI Netlink message, which triggers the kernel function xfrm_alloc_spi(). This function is expected to ensure uniqueness of the Security Parameter Index (SPI) for inbound Security Associations (SAs). However, it can return success even when the requested SPI is already in use, leading to duplicate SPIs assigned to multiple inbound SAs, differentiated only by their destination addresses. This behavior causes inconsistencies during SPI lookups for inbound packets. Since the lookup may return an arbitrary SA among those with the same SPI, packet processing can fail, resulting in packet drops. According to RFC 4301 section 4.4.2 , for inbound processing a unicast SA is uniquely identified by the SPI and optionally protocol. Reproducing the Issue Reliably: To consistently reproduce the problem, restrict the available SPI range in charon.conf : spi_min = 0x10000000 spi_max = 0x10000002 This limits the system to only 2 usable SPI values. Next, create more than 2 Child SA. each using unique pair of src/dst address. As soon as the 3rd Child SA is initiated, it will be assigned a duplicate SPI, since the SPI pool is already exhausted. With a narrow SPI range, the issue is consistently reproducible. With a broader/default range, it becomes rare and unpredictable. Current implementation: xfrm_spi_hash() lookup function computes hash using daddr, proto, and family. So if two SAs have the same SPI but different destination addresses, then they will: a. Hash into different buckets b. Be stored in different linked lists (by spi + h) c. Not be seen in the same hlist_for_each_entry_rcu() iteration. As a result, the lookup will result in NULL and kernel allows that Duplicate SPI Proposed Change: xfrm_state_lookup_spi_proto() does a truly global search - across all states, regardless of hash bucket and matches SPI and proto.
CVE-2025-39835	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: xfs: do not propagate ENODATA disk errors into xattr code ENODATA (aka ENOATTR) has a very specific meaning in the xfs xattr code; namely, that the requested attribute name could not be found. However, a medium error from disk may also return ENODATA. At best, this medium error may escape to userspace as "attribute not found" when in fact it's an IO (disk) error. At worst, we may oops in xfs_attr_leaf_get() when we do: error = xfs_attr_leaf_hasname(args, &bp); if (error == -ENOATTR) { xfs_trans_brelse(args->trans, bp); return error; } because an ENODATA/ENOATTR error from disk leaves us with a null bp, and the xfs_trans_brelse will then null-deref it. As discussed on the list, we really need to modify the lower level IO functions to trap all disk errors and ensure that we don't let unique errors like this leak up into higher xfs functions - many like this should be remapped to EIO. However, this patch directly addresses a reported bug in the xattr code, and should be safe to backport to stable kernels. A larger-scope patch to handle more unique errors at lower levels can follow later. (Note, prior to 07120f1abdff we did not oops, but we did return the wrong error code to userspace.)
CVE-2024-42118	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: drm/amd/display: Do not return negative stream id for array [WHY] resource_stream_to_stream_idx returns an array index and it return -1 when not found; however, -1 is not a valid array index number. [HOW] When this happens, call ASSERT(), and return a zero instead. This fixes an OVERRUN and an NEGATIVE_RETURNS issues reported by Coverity.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38375	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: virtio-net: ensure the received length does not exceed allocated size In xdp_linearize_page, when reading the following buffers from the ring, we forget to check the received length with the true allocate size. This can lead to an out-of-bound read. This commit adds that missing check.
CVE-2025-39823	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: KVM: x86: use array_index_nospec with indices that come from guest min and dest_id are guest-controlled indices. Using array_index_nospec() after the bounds checks clamps these values to mitigate speculative execution side-channels.
CVE-2025-71143	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: clk: samsung: exynos-clkout: Assign .num before accessing .hws Commit f316cdff8d67 ("clk: Annotate struct clk_hw_onecell_data with __counted_by") annotated the hws member of 'struct clk_hw_onecell_data' with __counted_by, which informs the bounds sanitizer (UBSAN_BOUNDS) about the number of elements in .hws[], so that it can warn when .hws[] is accessed out of bounds. As noted in that change, the __counted_by member must be initialized with the number of elements before the first array access happens, otherwise there will be a warning from each access prior to the initialization because the number of elements is zero. This occurs in exynos_clkout_probe() due to .num being assigned after .hws[] has been accessed: UBSAN: array-index-out-of-bounds in drivers/clk/samsung/clk-exynos-clkout.c:178:18 index 0 is out of range for type 'clk_hw *[*]' Move the .num initialization to before the first access of .hws[], clearing up the warning.
CVE-2024-26914	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: drm/amd/display: fix incorrect mpc_combine array size [why] MAX_SURFACES is per stream, while MAX_PLANES is per asic. The mpc_combine is an array that records all the planes per asic. Therefore MAX_PLANES should be used as the array size. Using MAX_SURFACES causes array overflow when there are more than 3 planes. [how] Use the MAX_PLANES for the mpc_combine array size.
CVE-2025-38697	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: jfs: upper bound check of tree index in dbAllocAG When computing the tree index in dbAllocAG, we never check if we are out of bounds realative to the size of the stree. This could happen in a scenario where the filesystem metadata are corrupted.
CVE-2025-71086	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net: rose: fix invalid array index in rose_kill_by_device() rose_kill_by_device() collects sockets into a local array[] and then iterates over them to disconnect sockets bound to a device being brought down. The loop mistakenly indexes array[cnt] instead of array[i]. For cnt < ARRAY_SIZE(array), this reads an uninitialized entry; for cnt == ARRAY_SIZE(array), it is an out-of-bounds read. Either case can lead to an invalid socket pointer dereference and also leaks references taken via sock_hold(). Fix the index to use i.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-40014	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: objtool, spi: amd: Fix out-of-bounds stack access in amd_set_spi_freq() If speed_hz < AMD_SPI_MIN_HZ, amd_set_spi_freq() iterates over the entire amd_spi_freq array without breaking out early, causing 'i' to go beyond the array bounds. Fix that by stopping the loop when it gets to the last entry, so the low speed_hz value gets clamped up to AMD_SPI_MIN_HZ. Fixes the following warning with an UBSAN kernel: drivers/spi/spi-amd.o: error: objtool: amd_set_spi_freq() falls through to next function amd_spi_set_opcode()
CVE-2024-53203	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: usb: typec: fix potential array underflow in ucsi_ccg_sync_control() The "command" variable can be controlled by the user via debugfs. The worry is that if con_index is zero then "&uc->ucsi->connector[con_index - 1]" would be an array underflow.
CVE-2024-46729	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: drm/amd/display: Fix incorrect size calculation for loop [WHY] fe_clk_en has size of 5 but sizeof(fe_clk_en) has byte size 20 which is lager than the array size. [HOW] Divide byte size 20 by its element size. This fixes 2 OVERRUN issues reported by Coverity.
CVE-2025-39967	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: fbcon: fix integer overflow in fbcon_do_set_font Fix integer overflow vulnerabilities in fbcon_do_set_font() where font size calculations could overflow when handling user-controlled font parameters. The vulnerabilities occur when: 1. CALC_FONTSZ(h, pitch, charcount) performs h * pith * charcount multiplication with user-controlled values that can overflow. 2. FONT_EXTRA_WORDS * sizeof(int) + size addition can also overflow 3. This results in smaller allocations than expected, leading to buffer overflows during font data copying. Add explicit overflow checking using check_mul_overflow() and check_add_overflow() kernel helpers to safety validate all size calculations before allocation.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38385	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net: usb: lan78xx: fix WARN in __netif_napi_del_locked on disconnect Remove redundant netif_napi_del() call from disconnect path. A WARN may be triggered in __netif_napi_del_locked() during USB device disconnect: WARNING: CPU: 0 PID: 11 at net/core/dev.c:7417 __netif_napi_del_locked+0x2b4/0x350 This happens because netif_napi_del() is called in the disconnect path while NAPI is still enabled. However, it is not necessary to call netif_napi_del() explicitly, since unregister_netdev() will handle NAPI teardown automatically and safely. Removing the redundant call avoids triggering the warning. Full trace: lan78xx 1-1:1.0 enu1: Failed to read register index 0x000000c4. ret = -ENODEV lan78xx 1-1:1.0 enu1: Failed to set MAC down with error -ENODEV lan78xx 1-1:1.0 enu1: Link is Down lan78xx 1-1:1.0 enu1: Failed to read register index 0x00000120. ret = -ENODEV ------------[cut here]------------ WARNING: CPU: 0 PID: 11 at net/core/dev.c:7417 __netif_napi_del_locked+0x2b4/0x350 Modules linked in: flexcan can_dev fuse CPU: 0 UID: 0 PID: 11 Comm: kworker/0:1 Not tainted 6.16.0-rc2-00624-ge926949dab03 #9 PREEMPT Hardware name: SKOV IMX8MP CPU revC - bd500 (DT) Workqueue: usb_hub_wq hub_event pstate: 60000005 (nZCv daif -PAN -UAO -TCO -DIT -SSBS BTYPE=--) pc : __netif_napi_del_locked+0x2b4/0x350 lr : __netif_napi_del_locked+0x7c/0x350 sp : ffffffc085b673c0 x29: ffffffc085b673c0 x28: ffffff800b7f2000 x27: ffffff800b7f20d8 x26: ffffff80110bcf58 x25: ffffff80110bd978 x24: 1ffffff0022179eb x23: ffffff80110bc000 x22: ffffff800b7f5000 x21: ffffff80110bc000 x20: ffffff80110bcf38 x19: ffffff80110bcf28 x18: dfffffc000000000 x17: ffffffc081578940 x16: ffffffc08284cee0 x15: 0000000000000028 x14: 0000000000000006 x13: 0000000000040000 x12: ffffffb0022179e8 x11: 1ffffff0022179e7 x10: ffffffb0022179e7 x9 : dfffffc000000000 x8 : 0000004ffdde8619 x7 : ffffff80110bcf3f x6 : 0000000000000001 x5 : ffffff80110bcf38 x4 : ffffff80110bcf38 x3 : 0000000000000000 x2 : 0000000000000000 x1 : 1ffffff0022179e7 x0 : 0000000000000000 Call trace: __netif_napi_del_locked+0x2b4/0x350 (P) lan78xx_disconnect+0xf4/0x360 usb_unbind_interface+0x158/0x718 device_remove+0x100/0x150 device_release_driver_internal+0x308/0x478 device_release_driver+0x1c/0x30 bus_remove_device+0x1a8/0x368 device_del+0x2e0/0x7b0 usb_disable_device+0x244/0x540 usb_disconnect+0x220/0x758 hub_event+0x105c/0x35e0 process_one_work+0x760/0x17b0 worker_thread+0x768/0xce8 kthread+0x3bc/0x690 ret_from_fork+0x10/0x20 irq event stamp: 211604 hardirqs last enabled at (211603): [] _raw_spin_unlock_irqrestore+0x84/0x98 hardirqs last disabled at (211604): [] el1_dbg+0x24/0x80 softirqs last enabled at (211296): [] handle_softirqs+0x820/0xbc8 softirqs last disabled at (210993): [] __do_softirq+0x18/0x20 ---[end trace 0000000000000000]--- lan78xx 1-1:1.0 enu1: failed to kill vid 0081/0

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38593	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: Bluetooth: hci_sync: fix double free in 'hci_discovery_filter_clear()' Function 'hci_discovery_filter_clear()' frees 'uuids' array and then sets it to NULL. There is a tiny chance of the following race: 'hci_cmd_sync_work()' 'update_passive_scan_sync()' 'hci_update_passive_scan_sync()' 'hci_discovery_filter_clear()' kfree(uuids); <----------preempted----------> 'start_service_discovery()' 'hci_discovery_filter_clear()' kfree(uuids); // DOUBLE FREE <----------preempted----------> uuids = NULL; To fix it let's add locking around 'kfree()' call and NULL pointer assignment. Otherwise the following backtrace fires: [ ] ------------[cut here]------------ [ ] kernel BUG at mm/slub.c:547! [ ] Internal error: Oops - BUG: 00000000f2000800 [#1] PREEMPT SMP [ ] CPU: 3 UID: 0 PID: 246 Comm: bluetoothd Tainted: G O 6.12.19-kernel #1 [ ] Tainted: [O]=OOT_MODULE [ ] pstate: 60400005 (nZCv daif +PAN -UAO -TCO -DIT -SSBS BTYPE=--) [ ] pc : __slab_free+0xf8/0x348 [ ] lr : __slab_free+0x48/0x348 ... [ ] Call trace: [ ] __slab_free+0xf8/0x348 [ ] kfree+0x164/0x27c [ ] start_service_discovery+0x1d0/0x2c0 [ ] hci_sock_sendmsg+0x518/0x924 [ ] __sock_sendmsg+0x54/0x60 [ ] sock_write_iter+0x98/0xf8 [ ] do_iter_readv_writev+0xe4/0x1c8 [ ] vfs_writev+0x128/0x2b0 [ ] do_writev+0xfc/0x118 [ ] __arm64_sys_writev+0x20/0x2c [ ] invoke_syscall+0x68/0xf0 [ ] el0_svc_common.constprop.0+0x40/0xe0 [ ] do_el0_svc+0x1c/0x28 [ ] el0_svc+0x30/0xd0 [ ] el0t_64_sync_handler+0x100/0x12c [ ] el0t_64_sync+0x194/0x198 [ ] Code: 8b0002e6 eb17031f 54fffbe1 d503201f (d4210000) [ ] ---[end trace 0000000000000000]---
CVE-2025-38069	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: PCI: endpoint: pci-epf-test: Fix double free that causes kernel to oops Fix a kernel oops found while testing the stm32_pcie Endpoint driver with handling of PERST# deassertion: During EP initialization, pci_epf_test_alloc_space() allocates all BARs, which are further freed if epc_set_bar() fails (for instance, due to no free inbound window). However, when pci_epc_set_bar() fails, the error path: pci_epc_set_bar() -> pci_epf_free_space() does not clear the previous assignment to epf_test->reg[bar]. Then, if the host reboots, the PERST# deassertion restarts the BAR allocation sequence with the same allocation failure (no free inbound window), creating a double free situation since epf_test->reg[bar] was deallocated and is still non-NULL. Thus, make sure that pci_epf_alloc_space() and pci_epf_free_space() invocations are symmetric, and as such, set epf_test->reg[bar] to NULL when memory is freed. [kwilczynski: commit log]

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38582	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: RDMA/hns: Fix double destruction of rsv_qp rsv_qp may be double destroyed in error flow, first in free_mr_init(), and then in hns_roce_exit(). Fix it by moving the free_mr_init() call into hns_roce_v2_init(). list_del corruption, ffff589732eb9b50->next is LIST_POISON1 (dead000000000100) WARNING: CPU: 8 PID: 1047115 at lib/list_debug.c:53 __list_del_entry_valid+0x148/0x240 ... Call trace: __list_del_entry_valid+0x148/0x240 hns_roce_qp_remove+0x4c/0x3f0 [hns_roce_hw_v2] hns_roce_v2_destroy_qp_common+0x1dc/0x5f4 [hns_roce_hw_v2] hns_roce_v2_destroy_qp+0x22c/0x46c [hns_roce_hw_v2] free_mr_exit+0x6c/0x120 [hns_roce_hw_v2] hns_roce_v2_exit+0x170/0x200 [hns_roce_hw_v2] hns_roce_exit+0x118/0x350 [hns_roce_hw_v2] __hns_roce_hw_v2_init_instance+0x1c8/0x304 [hns_roce_hw_v2] hns_roce_hw_v2_reset_notify_init+0x170/0x21c [hns_roce_hw_v2] hns_roce_hw_v2_reset_notify+0x6c/0x190 [hns_roce_hw_v2] hclge_notify_roce_client+0x6c/0x160 [hclge] hclge_reset_rebuild+0x150/0x5c0 [hclge] hclge_reset+0x10c/0x140 [hclge] hclge_reset_subtask+0x80/0x104 [hclge] hclge_reset_service_task+0x168/0x3ac [hclge] hclge_service_task+0x50/0x100 [hclge] process_one_work+0x250/0x9a0 worker_thread+0x324/0x990 kthread+0x190/0x210 ret_from_fork+0x10/0x18
CVE-2025-39790	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: bus: mhi: host: Detect events pointing to unexpected TREs When a remote device sends a completion event to the host, it contains a pointer to the consumed TRE. The host uses this pointer to process all of the TREs between it and the host's local copy of the ring's read pointer. This works when processing completion for chained transactions, but can lead to nasty results if the device sends an event for a single-element transaction with a read pointer that is multiple elements ahead of the host's read pointer. For instance, if the host accesses an event ring while the device is updating it, the pointer inside of the event might still point to an old TRE. If the host uses the channel's xfer_cb() to directly free the buffer pointed to by the TRE, the buffer will be double-freed. This behavior was observed on an ep that used upstream EP stack without 'commit 6f18d174b73d ("bus: mhi: ep: Update read pointer only after buffer is written")'. Where the device updated the events ring pointer before updating the event contents, so it left a window where the host was able to access the stale data the event pointed to, before the device had the chance to update them. The usual pattern was that the host received an event pointing to a TRE that is not immediately after the last processed one, so it got treated as if it was a chained transaction, processing all of the TREs in between the two read pointers. This commit aims to harden the host by ensuring transactions where the event points to a TRE that isn't local_rp + 1 are chained. [mani: added stable tag and reworded commit message]

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-39870	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: dmaengine: idxd: Fix double free in idxd_setup_wqs() The clean up in idxd_setup_wqs() has had a couple bugs because the error handling is a bit subtle. It's simpler to just re-write it in a cleaner way. The issues here are: 1) If "idxd->max_wqs" is <= 0 then we call put_device(conf_dev) when "conf_dev" hasn't been initialized. 2) If kzalloc_node() fails then again "conf_dev" is invalid. It's either uninitialized or it points to the "conf_dev" from the previous iteration so it leads to a double free. It's better to free partial loop iterations within the loop and then the unwinding at the end can handle whole loop iterations. I also renamed the labels to describe what the goto does and not where the goto was located.
CVE-2025-38206	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: exfat: fix double free in delayed_free The double free could happen in the following path. exfat_create_upcase_table() exfat_create_upcase_table() : return error exfat_free_upcase_table() : free ->vol_utbl exfat_load_default_upcase_table : return error exfat_kill_sb() delayed_free() exfat_free_upcase_table() <--------- double free This patch set ->vol_util as NULL after freeing it.
CVE-2025-38490	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net: libwx: remove duplicate page_pool_put_full_page() page_pool_put_full_page() should only be invoked when freeing Rx buffers or building a skb if the size is too short. At other times, the pages need to be reused. So remove the redundant page put. In the original code, double free pages cause kernel panic: [876.949834] __irq_exit_rcu+0xc7/0x130 [876.949836] common_interrupt+0xb8/0xd0 [876.949838] [876.949838] [876.949840] asm_common_interrupt+0x22/0x40 [876.949841] RIP: 0010:cpuidle_enter_state+0xc2/0x420 [876.949843] Code: 00 00 e8 d1 1d 5e ff e8 ac f0 ff ff 49 89 c5 0f 1f 44 00 00 31 ff e8 cd fc 5c ff 45 84 ff 0f 85 40 02 00 00 fb 0f 1f 44 00 00 <45> 85 f6 0f 88 84 01 00 00 49 63 d6 48 8d 04 52 48 8d 04 82 49 8d [876.949844] RSP: 0018:ffffaa7340267e78 EFLAGS: 00000246 [876.949845] RAX: ffff9e3f135be000 RBX: 0000000000000002 RCX: 0000000000000000 [876.949846] RDX: 000000cc2dc4cb7c RSI: ffffffff89ee49ae RDI: ffffffff89ef9f9e [876.949847] RBP: ffff9e378f940800 R08: 0000000000000002 R09: 00000000000000ed [876.949848] R10: 000000000000afc8 R11: ffff9e3e9e5a9b6c R12: ffffffff8a6d8580 [876.949849] R13: 000000cc2dc4cb7c R14: 0000000000000002 R15: 0000000000000000 [876.949852] ? cpuidle_enter_state+0xb3/0x420 [876.949855] cpuidle_enter+0x29/0x40 [876.949857] cpuidle_idle_call+0xfd/0x170 [876.949859] do_idle+0x7a/0xc0 [876.949861] cpu_startup_entry+0x25/0x30 [876.949862] start_secondary+0x117/0x140 [876.949864] common_startup_64+0x13e/0x148 [876.949867] [876.949868] ---[end trace 0000000000000000]--- [876.949869] ------------[cut here]------------ [876.949870] list_del corruption, ffffead40445a348->next is NULL [876.949873] WARNING: CPU: 14 PID: 0 at lib/list_debug.c:52 __list_del_entry_valid_or_report+0x67/0x120 [876.949875] Modules linked in: snd_hrtimer(E) bnep(E) binfmt_misc(E) amdgpu(E) squashfs(E) vfat(E) loop(E) fat(E) amd_atl(E) snd_hda_codec_realtek(E) intel_rapl_msr(E) snd_hda_codec_generic(E) intel_rapl_common(E) snd_hda_scodec_component(E) snd_hda_codec_hdmi(E) snd_hda_intel(E) edac_mce_amd(E) snd_intel_dspcfg(E) snd_hda_codec(E) snd_hda_core(E) amdxcp(E) kvm_amd(E) snd_hwdep(E) gpu_sched(E)

CVE	Severity	CVSS	CVSS Vector	Component	Description
					drm_panel_backlight_quirks(E) cec(E) snd_pcm(E) drm_buddy(E) snd_seq_dummy(E) drm_ttm_helper(E) btusb(E) kvm(E) snd_seq_oss(E) btrtl(E) ttm(E) btintel(E) snd_seq_midi(E) btbcm(E) drm_exec(E) snd_seq_midi_event(E) i2c_algo_bit(E) snd_rawmidi(E) bluetooth(E) drm_suballoc_helper(E) irqbypass(E) snd_seq(E) ghash_clmulni_intel(E) sha512_ssse3(E) drm_display_helper(E) aesni_intel(E) snd_seq_device(E) rfkill(E) snd_timer(E) gf128mul(E) drm_client_lib(E) drm_kms_helper(E) snd(E) i2c_piix4(E) joydev(E) soundcore(E) wmi_bmof(E) ccp(E) k10temp(E) i2c_smbus(E) gpio_amdpt(E) i2c_designware_platform(E) gpio_generic(E) sg(E) [876.949914] i2c_designware_core(E) sch_fq_codel(E) parport_pc(E) drm(E) ppdev(E) lp(E) parport(E) fuse(E) nfnetlink(E) ip_tables(E) ext4 crc16 mbcache jbd2 sd_mod sfp mdio_i2c i2c_core txgbe ahci ngbe pcs_xpcs libahci libwx r8169 phylink libata realtek ptp pps_core video wmi [876.949933] CPU: 14 UID: 0 PID: 0 Comm: swapper/14 Kdump: loaded Tainted: G W E 6.16.0-rc2+ #20 PREEMPT(voluntary) [876.949935] Tainted: [W]=WARN, [E]=UNSIGNED_MODULE [876.949936] Hardware name: Micro-Star International Co., Ltd. MS-7E16/X670E GAMING PLUS WIFI (MS-7E16), BIOS 1.90 12/31/2024 [876.949936] RIP: 0010:__list_del_entry_valid_or_report+0x67/0x120 [876.949938] Code: 00 00 00 48 39 7d 08 0f 85 a6 00 00 00 5b b8 01 00 00 00 5d 41 5c e9 73 0d 93 ff 48 89 fe 48 c7 c7 a0 31 e8 89 e8 59 7c b3 ff <0f> 0b 31 c0 5b 5d 41 5c e9 57 0d 93 ff 48 89 fe 48 c7 c7 c8 31 e8 [876.949940] RSP: 0018:ffffaa73405d0c60 EFLAGS: 00010282 [876.949941] RAX: 0000000000000000 RBX: ffffead40445a348 RCX: 0000000000000000 [876.949942] RDX: 0000000000000105 RSI: 00000 ---truncated---
CVE-2026-23098	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: netrom: fix double-free in nr_route_frame() In nr_route_frame(), old_skb is immediately freed without checking if nr_neigh->ax25 pointer is NULL. Therefore, if nr_neigh->ax25 is NULL, the caller function will free old_skb again, causing a double-free bug. Therefore, to prevent this, we need to modify it to check whether nr_neigh->ax25 is NULL before freeing old_skb.
CVE-2025-38699	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: scsi: bfa: Double-free fix When the bfad_im_probe() function fails during initialization, the memory pointed to by bfad->im is freed without setting bfad->im to NULL. Subsequently, during driver uninstallation, when the state machine enters the bfad_sm_stopping state and calls the bfad_im_probe_undo() function, it attempts to free the memory pointed to by bfad->im again, thereby triggering a double-free vulnerability. Set bfad->im to NULL if probing fails.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-71238	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: scsi: qla2xxx: Fix bsg_done() causing double free Kernel panic observed on system, [5353358.825191] BUG: unable to handle page fault for address: ff5f5e897b024000 [5353358.825194] #PF: supervisor write access in kernel mode [5353358.825195] #PF: error_code(0x0002) - not-present page [5353358.825196] PGD 100006067 P4D 0 [5353358.825198] Oops: 0002 [#1] PREEMPT SMP NOPTI [5353358.825200] CPU: 5 PID: 2132085 Comm: qlafwupdate.sub Kdump: loaded Tainted: G W L ------- --- 5.14.0-503.34.1.el9_5.x86_64 #1 [5353358.825203] Hardware name: HPE ProLiant DL360 Gen11/ProLiant DL360 Gen11, BIOS 2.44 01/17/2025 [5353358.825204] RIP: 0010:memcpy_erms+0x6/0x10 [5353358.825211] RSP: 0018:ff591da8f4f6b710 EFLAGS: 00010246 [5353358.825212] RAX: ff5f5e897b024000 RBX: 0000000000007090 RCX: 0000000000001000 [5353358.825213] RDX: 0000000000001000 RSI: ff591da8f4fed090 RDI: ff5f5e897b024000 [5353358.825214] RBP: 0000000000010000 R08: ff5f5e897b024000 R09: 0000000000000000 [5353358.825215] R10: ff46cf8c40517000 R11: 0000000000000001 R12: 0000000000008090 [5353358.825216] R13: ff591da8f4f6b720 R14: 0000000000001000 R15: 0000000000000000 [5353358.825218] FS: 00007f1e88d47740(0000) GS:ff46cf935f940000(0000) knlGS:0000000000000000 [5353358.825219] CS: 0010 DS: 0000 ES: 0000 CR0: 0000000080050033 [5353358.825220] CR2: ff5f5e897b024000 CR3: 0000000231532004 CR4: 0000000000771ef0 [5353358.825221] PKRU: 55555554 [5353358.825222] Call Trace: [5353358.825223] [5353358.825224] ? show_trace_log_lvl+0x1c4/0x2df [5353358.825229] ? show_trace_log_lvl+0x1c4/0x2df [5353358.825232] ? sg_copy_buffer+0xc8/0x110 [5353358.825236] ? __die_body.cold+0x8/0xd [5353358.825238] ? page_fault_oops+0x134/0x170 [5353358.825242] ? kernelmode_fixup_or_oops+0x84/0x110 [5353358.825244] ? exc_page_fault+0xa8/0x150 [5353358.825247] ? asm_exc_page_fault+0x22/0x30 [5353358.825252] ? memcpy_erms+0x6/0x10 [5353358.825253] sg_copy_buffer+0xc8/0x110 [5353358.825259] qla2x00_process_vendor_specific+0x652/0x1320 [qla2xxx] [5353358.825317] qla24xx_bsg_request+0x1b2/0x2d0 [qla2xxx] Most routines in qla_bsg.c call bsg_done() only for success cases. However a few invoke it for failure case as well leading to a double free. Validate before calling bsg_done().
CVE-2026-23068	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: spi: spi-sprd-adi: Fix double free in probe error path The driver currently uses spi_alloc_host() to allocate the controller but registers it using devm_spi_register_controller(). If devm_register_restart_handler() fails, the code jumps to the put_ctlr label and calls spi_controller_put(). However, since the controller was registered via a devm function, the device core will automatically call spi_controller_put() again when the probe fails. This results in a double-free of the spi_controller structure. Fix this by switching to devm_spi_alloc_host() and removing the manual spi_controller_put() call.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2026-23089	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ALSA: usb-audio: Fix use-after-free in snd_usb_mixer_free() When snd_usb_create_mixer() fails, snd_usb_mixer_free() frees mixer->id_elems but the controls already added to the card still reference the freed memory. Later when snd_card_register() runs, the OSS mixer layer calls their callbacks and hits a use-after-free read. Call trace: get_ctl_value+0x63f/0x820 sound/usb/mixer.c:411 get_min_max_with_quirks.isra.0+0x240/0x1f40 sound/usb/mixer.c:1241 mixer_ctl_feature_info+0x26b/0x490 sound/usb/mixer.c:1381 snd_mixer_oss_build_test+0x174/0x3a0 sound/core/oss/mixer_oss.c:887 ... snd_card_register+0x4ed/0x6d0 sound/core/init.c:923 usb_audio_probe+0x5ef/0x2a90 sound/usb/card.c:1025 Fix by calling snd_ctl_remove() for all mixer controls before freeing id_elems. We save the next pointer first because snd_ctl_remove() frees the current element.
CVE-2025-39860	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: Bluetooth: Fix use-after-free in l2cap_sock_cleanup_listen() syzbot reported the splat below without a repro. In the splat, a single thread calling bt_accept_dequeue() freed sk and touched it after that. The root cause would be the racy l2cap_sock_cleanup_listen() call added by the cited commit. bt_accept_dequeue() is called under lock_sock() except for l2cap_sock_release(). Two threads could see the same socket during the list iteration in bt_accept_dequeue(): CPU1 CPU2 (close()) ---- ---- sock_hold(sk) sock_hold(sk); lock_sock(sk) <-- block close() sock_put(sk) bt_accept_unlink(sk) sock_put(sk) <-- refcnt by bt_accept_enqueue() release_sock(sk) lock_sock(sk) sock_put(sk) bt_accept_unlink(sk) sock_put(sk) <-- last refcnt bt_accept_unlink(sk) <-- UAF Depending on the timing, the other thread could show up in the "Freed by task" part. Let's call l2cap_sock_cleanup_listen() under lock_sock() in l2cap_sock_release(). [0]: BUG: KASAN: slab-use-after-free in debug_spin_lock_before kernel/locking/spinlock_debug.c:86 [inline] BUG: KASAN: slab-use-after-free in do_raw_spin_lock+0x26f/0x2b0 kernel/locking/spinlock_debug.c:115 Read of size 4 at addr ffff88803b7eb1c4 by task syz.5.3276/16995 CPU: 3 UID: 0 PID: 16995 Comm: syz.5.3276 Not tainted syzkaller #0 PREEMPT(full) Hardware name: QEMU Standard PC (Q35 + ICH9, 2009), BIOS 1.16.3-debian-1.16.3-2~bpo12+1 04/01/2014 Call Trace: __dump_stack lib/dump_stack.c:94 [inline] dump_stack_lvl+0x116/0x1f0 lib/dump_stack.c:120 print_address_description mm/kasan/report.c:378 [inline] print_report+0xcd/0x630 mm/kasan/report.c:482 kasan_report+0xe0/0x110 mm/kasan/report.c:595 debug_spin_lock_before kernel/locking/spinlock_debug.c:86 [inline] do_raw_spin_lock+0x26f/0x2b0 kernel/locking/spinlock_debug.c:115 spin_lock_bh include/linux/spinlock.h:356 [inline] release_sock+0x21/0x220 net/core/sock.c:3746 bt_accept_dequeue+0x505/0x600 net/bluetooth/af_bluetooth.c:312 l2cap_sock_cleanup_listen+0x5c/0x2a0 net/bluetooth/l2cap_sock.c:1451 l2cap_sock_release+0x5c/0x210 net/bluetooth/l2cap_sock.c:1425 __sock_release+0xb3/0x270 net/socket.c:649 sock_close+0x1c/0x30 net/socket.c:1439 __fput+0x3ff/0xb70 fs/file_table.c:468 task_work_run+0x14d/0x240 kernel/task_work.c:227 resume_user_mode_work

CVE	Severity	CVSS	CVSS Vector	Component	Description
					include/linux/resume_user_mode.h:50 [inline] exit_to_user_mode_loop+0xeb/0x110 kernel/entry/common.c:43 exit_to_user_mode_prepare include/linux/irq-entry-common.h:225 [inline] syscall_exit_to_user_mode_work include/linux/entry-common.h:175 [inline] syscall_exit_to_user_mode include/linux/entry-common.h:210 [inline] do_syscall_64+0x3f6/0x4c0 arch/x86/entry/syscall_64.c:100 entry_SYSCALL_64_after_hwframe+0x77/0x7f RIP: 0033:0x7f2accf8ebe9 Code: ff ff c3 66 2e 0f 1f 84 00 00 00 00 00 0f 1f 40 00 48 89 f8 48 89 f7 48 89 d6 48 89 ca 4d 89 c2 4d 89 c8 4c 8b 4c 24 08 0f 05 <48> 3d 01 f0 ff ff 73 01 c3 48 c7 c1 a8 ff ff ff f7 d8 64 89 01 48 RSP: 002b:00007ffdb6cb1378 EFLAGS: 00000246 ORIG_RAX: 00000000000001b4 RAX: 0000000000000000 RBX: 00000000000426fb RCX: 00007f2accf8ebe9 RDX: 0000000000000000 RSI: 000000000000001e RDI: 0000000000000003 RBP: 00007f2acd1b7da0 R08: 0000000000000001 R09: 00000012b6cb166f R10: 0000001b30e20000 R11: 0000000000000246 R12: 00007f2acd1b609c R13: 00007f2acd1b6090 R14: ffffffffffffffff R15: 00007ffdb6cb1490 Allocated by task 5326: kasan_save_stack+0x33/0x60 mm/kasan/common.c:47 kasan_save_track+0x14/0x30 mm/kasan/common.c:68 poison_kmalloc_redzone mm/kasan/common.c:388 [inline] __kasan_kmalloc+0xaa/0xb0 mm/kasan/common.c:405 kasan_kmalloc include/linux/kasan.h:260 [inline] __do_kmalloc_node mm/slub.c:4365 [inline] __kmalloc_nopro ---truncated---
CVE-2025-38250	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: Bluetooth: hci_core: Fix use-after-free in vhci_flush() syzbot reported use-after-free in vhci_flush() without repro. [0] From the splat, a thread close()d a vhci file descriptor while its device was being used by iotcl() on another thread. Once the last fd refcnt is released, vhci_release() calls hci_unregister_dev(), hci_free_dev(), and kfree() for struct vhci_data, which is set to hci_dev->dev->driver_data. The problem is that there is no synchronisation after unlinking hdev from hci_dev_list in hci_unregister_dev(). There might be another thread still accessing the hdev which was fetched before the unlink operation. We can use SRCU for such synchronisation. Let's run hci_dev_reset() under SRCU and wait for its completion in hci_unregister_dev(). Another option would be to restore hci_dev->destruct(), which was removed in commit 587ae086f6e4 ("Bluetooth: Remove unused hci-destruct cb"). However, this would not be a good solution, as we should not run hci_unregister_dev() while there are in-flight ioctl() requests, which could lead to another data-race KCSAN splat. Note that other drivers seem to have the same problem, for exmaple, virtbt_remove(). [0]: BUG: KASAN: slab-use-after-free in skb_queue_empty_lockless include/linux/skbuff.h:1891 [inline] BUG: KASAN: slab-use-after-free in skb_queue_purge_reason+0x99/0x360 net/core/skbuff.c:3937 Read of size 8 at addr ffff88807cb8d858 by task syz.1.219/6718 CPU: 1 UID: 0 PID: 6718 Comm: syz.1.219 Not tainted 6.16.0-rc1-syzkaller-00196-g08207f42d3ff #0 PREEMPT(full) Hardware name: Google Google Compute Engine/Google Compute Engine, BIOS Google 05/07/2025 Call Trace: dump_stack_lvl+0x189/0x250 lib/dump_stack.c:120 print_address_description mm/kasan/report.c:408 [inline] print_report+0xd2/0x2b0 mm/kasan/report.c:521 kasan_report+0x118/0x150 mm/kasan/report.c:634 skb_queue_empty_lockless include/linux/skbuff.h:1891 [inline] skb_queue_purge_reason+0x99/0x360

CVE	Severity	CVSS	CVSS Vector	Component	Description
					net/core/skbuff.c:3937 skb_queue_purge include/linux/skbuff.h:3368 [inline] vhci_flush+0x44/0x50 drivers/bluetooth/hci_vhci.c:69 hci_dev_do_reset net/bluetooth/hci_core.c:552 [inline] hci_dev_reset+0x420/0x5c0 net/bluetooth/hci_core.c:592 sock_do_ioctl+0xd9/0x300 net/socket.c:1190 sock_ioctl+0x576/0x790 net/socket.c:1311 vfs_ioctl fs/ioctl.c:51 [inline] __do_sys_ioctl fs/ioctl.c:907 [inline] __se_sys_ioctl+0xf9/0x170 fs/ioctl.c:893 do_syscall_x64 arch/x86/entry/syscall_64.c:63 [inline] do_syscall_64+0xfa/0x3b0 arch/x86/entry/syscall_64.c:94 entry_SYSCALL_64_after_hwframe+0x77/0x7f RIP: 0033:0x7fcf5b98e929 Code: ff ff c3 66 2e 0f 1f 84 00 00 00 00 00 0f 1f 40 00 48 89 f8 48 89 f7 48 89 d6 48 89 ca 4d 89 c2 4d 89 c8 4c 8b 4c 24 08 0f 05 <48> 3d 01 f0 ff ff 73 01 c3 48 c7 c1 a8 ff ff ff f7 d8 64 89 01 48 RSP: 002b:00007fcf5c7b9038 EFLAGS: 00000246 ORIG_RAX: 0000000000000010 RAX: ffffffffffffffda RBX: 00007fcf5bbb6160 RCX: 00007fcf5b98e929 RDX: 0000000000000000 RSI: 00000000400448cb RDI: 0000000000000009 RBP: 00007fcf5ba10b39 R08: 0000000000000000 R09: 0000000000000000 R10: 0000000000000000 R11: 0000000000000246 R12: 0000000000000000 R13: 0000000000000000 R14: 00007fcf5bbb6160 R15: 00007ffd6353d528 Allocated by task 6535: kasan_save_stack mm/kasan/common.c:47 [inline] kasan_save_track+0x3e/0x80 mm/kasan/common.c:68 poison_kmalloc_redzone mm/kasan/common.c:377 [inline] __kasan_kmalloc+0x93/0xb0 mm/kasan/common.c:394 kasan_kmalloc include/linux/kasan.h:260 [inline] __kmalloc_cache_noprof+0x230/0x3d0 mm/slub.c:4359 kmalloc_noprof include/linux/slab.h:905 [inline] kzalloc_noprof include/linux/slab.h:1039 [inline] vhci_open+0x57/0x360 drivers/bluetooth/hci_vhci.c:635 misc_open+0x2bc/0x330 drivers/char/misc.c:161 chrdev_open+0x4c9/0x5e0 fs/char_dev.c:414 do_dentry_open+0xdf0/0x1970 fs/open.c:964 vfs_open+0x3b/0x340 fs/open.c:1094 do_open fs/namei.c:3887 [inline] path_openat+0x2ee5/0x3830 fs/name ---truncated---
CVE-2025-39861	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: Bluetooth: vhci: Prevent use-after-free by removing debugfs files early Move the creation of debugfs files into a dedicated function, and ensure they are explicitly removed during vhci_release(), before associated data structures are freed. Previously, debugfs files such as "force_suspend", "force_wakeup", and others were created under hdev->debugfs but not removed in vhci_release(). Since vhci_release() frees the backing vhci_data structure, any access to these files after release would result in use-after-free errors. Although hdev->debugfs is later freed in hci_release_dev(), user can access files after vhci_data is freed but before hdev->debugfs is released.
CVE-2025-39824	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: HID: asus: fix UAF via HID_CLAIMED_INPUT validation After hid_hw_start() is called hidinput_connect() will eventually be called to set up the device with the input layer since the HID_CONNECT_DEFAULT connect mask is used. During hidinput_connect() all input and output reports are processed and corresponding hid_inputs are allocated and configured via hidinput_configure_usages(). This process involves slot tagging report fields and configuring usages by setting relevant bits in the capability bitmaps. However it is possible that the capability bitmaps are not set at all leading to the

CVE	Severity	CVSS	CVSS Vector	Component	Description
					subsequent hidinput_has_been_populated() check to fail leading to the freeing of the hid_input and the underlying input device. This becomes problematic because a malicious HID device like a ASUS ROG N-Key keyboard can trigger the above scenario via a specially crafted descriptor which then leads to a user-after-free when the name of the freed input device is written to later on after hid_hw_start(). Below, report 93 intentionally utilises the HID_UP_UNDEFINED Usage Page which is skipped during usage configuration, leading to the frees. 0x05, 0x0D, // Usage Page (Digitizer) 0x09, 0x05, // Usage (Touch Pad) 0xA1, 0x01, // Collection (Application) 0x85, 0x0D, // Report ID (13) 0x06, 0x00, 0xFF, // Usage Page (Vendor Defined 0xFF00) 0x09, 0xC5, // Usage (0xC5) 0x15, 0x00, // Logical Minimum (0) 0x26, 0xFF, 0x00, // Logical Maximum (255) 0x75, 0x08, // Report Size (8) 0x95, 0x04, // Report Count (4) 0xB1, 0x02, // Feature (Data,Var,Abs) 0x85, 0x5D, // Report ID (93) 0x06, 0x00, 0x00, // Usage Page (Undefined) 0x09, 0x01, // Usage (0x01) 0x15, 0x00, // Logical Minimum (0) 0x26, 0xFF, 0x00, // Logical Maximum (255) 0x75, 0x08, // Report Size (8) 0x95, 0x1B, // Report Count (27) 0x81, 0x02, // Input (Data,Var,Abs) 0xC0, // End Collection Below is the KASAN splat after triggering the UAF: [21.672709] == [21.673700] BUG: KASAN: slab-use-after-free in asus_probe+0xeeb/0xf80 [21.673700] Write of size 8 at addr ffff88810a0ac000 by task kworker/1:2/54 [21.673700] [21.673700] CPU: 1 UID: 0 PID: 54 Comm: kworker/1:2 Not tainted 6.16.0-rc4-g9773391cf4dd-dirty #36 PREEMPT(voluntary) [21.673700] Hardware name: QEMU Standard PC (i440FX + PIIX, 1996), BIOS 1.16.2-debian-1.16.2-1 04/01/2014 [21.673700] Call Trace: [21.673700] [21.673700] dump_stack_lvl+0x5f/0x80 [21.673700] print_report+0xd1/0x660 [21.673700] kasan_report+0xe5/0x120 [21.673700] __asan_report_store8_noabort+0x1b/0x30 [21.673700] asus_probe+0xeeb/0xf80 [21.673700] hid_device_probe+0x2ee/0x700 [21.673700] really_probe+0x1c6/0x6b0 [21.673700] __driver_probe_device+0x24f/0x310 [21.673700] driver_probe_device+0x4e/0x220 [...] [21.673700] [21.673700] Allocated by task 54: [21.673700] kasan_save_stack+0x3d/0x60 [21.673700] kasan_save_track+0x18/0x40 [21.673700] kasan_save_alloc_info+0x3b/0x50 [21.673700] __kasan_kmalloc+0x9c/0xa0 [21.673700] __kmalloc_cache_noprof+0x139/0x340 [21.673700] input_allocate_device+0x44/0x370 [21.673700] hidinput_connect+0xcb6/0x2630 [21.673700] hid_connect+0xf74/0x1d60 [21.673700] hid_hw_start+0x8c/0x110 [21.673700] asus_probe+0x5a3/0xf80 [21.673700] hid_device_probe+0x2ee/0x700 [21.673700] really_probe+0x1c6/0x6b0 [21.673700] __driver_probe_device+0x24f/0x310 [21.673700] driver_probe_device+0x4e/0x220 [...] [21.673700] [21.673700] Freed by task 54: [21.673700] kasan_save_stack+0x3d/0x60 [21.673700] kasan_save_track+0x18/0x40 [21.673700] kasan_save_free_info+0x3f/0x60 [21.673700] __kasan_slab_free+0x3c/0x50 [21.673700] kfre ---truncated---

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-71073	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: Input: lkkbd - disable pending work before freeing device lkkbd_interrupt() schedules lk->tq via schedule_work(), and the work handler lkkbd_reinit() dereferences the lkkbd structure and its serio/input_dev fields. lkkbd_disconnect() and error paths in lkkbd_connect() free the lkkbd structure without preventing the reinit work from being queued again until serio_close() returns. This can allow the work handler to run after the structure has been freed, leading to a potential use-after-free. Use disable_work_sync() instead of cancel_work_sync() to ensure the reinit work cannot be re-queued, and call it both in lkkbd_disconnect() and in lkkbd_connect() error paths after serio_open().
CVE-2024-58093	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: PCI/ASPM: Fix link state exit during switch upstream function removal Before 456d8aa37d0f ("PCI/ASPM: Disable ASPM on MFD function removal to avoid use-after-free"), we would free the ASPM link only after the last function on the bus pertaining to the given link was removed. That was too late. If function 0 is removed before sibling function, link->downstream would point to free'd memory after. After above change, we freed the ASPM parent link state upon any function removal on the bus pertaining to a given link. That is too early. If the link is to a PCIe switch with MFD on the upstream port, then removing functions other than 0 first would free a link which still remains parent_link to the remaining downstream ports. The resulting GPFs are especially frequent during hot-unplug, because pciehp removes devices on the link bus in reverse order. On that switch, function 0 is the virtual P2P bridge to the internal bus. Free exactly when function 0 is removed -- before the parent link is obsolete, but after all subordinate links are gone. [kwilczynski: commit log]

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38022	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: RDMA/core: Fix "KASAN: slab-use-after-free Read in ib_register_device" problem Call Trace: __dump_stack lib/dump_stack.c:94 [inline] dump_stack_lvl+0x116/0x1f0 lib/dump_stack.c:120 print_address_description mm/kasan/report.c:408 [inline] print_report+0xc3/0x670 mm/kasan/report.c:521 kasan_report+0xe0/0x110 mm/kasan/report.c:634 strlen+0x93/0xa0 lib/string.c:420 __fortify_strlen include/linux/fortify-string.h:268 [inline] get_kobj_path_length lib/kobject.c:118 [inline] kobject_get_path+0x3f/0x2a0 lib/kobject.c:158 kobject_uevent_env+0x289/0x1870 lib/kobject_uevent.c:545 ib_register_device drivers/infiniband/core/device.c:1472 [inline] ib_register_device+0x8cf/0xe00 drivers/infiniband/core/device.c:1393 rxe_register_device+0x275/0x320 drivers/infiniband/sw/rxe/rxe_verbs.c:1552 rxe_net_add+0x8e/0xe0 drivers/infiniband/sw/rxe/rxe_net.c:550 rxe_newlink+0x70/0x190 drivers/infiniband/sw/rxe/rxe.c:225 nldev_newlink+0x3a3/0x680 drivers/infiniband/core/nldev.c:1796 rdma_nl_rcv_msg+0x387/0x6e0 drivers/infiniband/core/netlink.c:195 rdma_nl_rcv_skb.constprop.0.isra.0+0x2e5/0x450 netlink_unicast_kernel net/netlink/af_netlink.c:1313 [inline] netlink_unicast+0x53a/0x7f0 net/netlink/af_netlink.c:1339 netlink_sendmsg+0x8d1/0xdd0 net/netlink/af_netlink.c:1883 sock_sendmsg_nosec net/socket.c:712 [inline] __sock_sendmsg net/socket.c:727 [inline] ____sys_sendmsg+0xa95/0xc70 net/socket.c:2566 ___sys_sendmsg+0x134/0x1d0 net/socket.c:2620 __sys_sendmsg+0x16d/0x220 net/socket.c:2652 do_syscall_x64 arch/x86/entry/syscall_64.c:63 [inline] do_syscall_64+0xcd/0x260 arch/x86/entry/syscall_64.c:94 entry_SYSCALL_64_after_hwframe+0x77/0x7f This problem is similar to the problem that the commit 1d6a9e7449e2 ("RDMA/core: Fix use-after-free when rename device name") fixes. The root cause is: the function ib_device_rename() renames the name with lock. But in the function kobject_uevent(), this name is accessed without lock protection at the same time. The solution is to add the lock protection when this name is accessed in the function kobject_uevent().
CVE-2024-53170	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: block: fix uaf for flush rq while iterating tags blk_mq_clear_flush_rq_mapping() is not called during scsi probe, by checking blk_queue_init_done(). However, QUEUE_FLAG_INIT_DONE is cleared in del_gendisk by commit aec89dc5d421 ("block: keep q_usage_counter in atomic mode after del_gendisk"), hence for disk like scsi, following blk_mq_destroy_queue() will not clear flush rq from tags->rqs[] as well, cause following uaf that is found by our syzkaller for v6.6: == BUG: KASAN: slab-use-after-free in blk_mq_find_and_get_req+0x16e/0x1a0 block/blk-mq-tag.c:261 Read of size 4 at addr ffff88811c969c20 by task kworker/1:2H/224909 CPU: 1 PID: 224909 Comm: kworker/1:2H Not tainted 6.6.0-ga836a5060850 #32 Workqueue: kblockd blk_mq_timeout_work Call Trace: __dump_stack lib/dump_stack.c:88 [inline] dump_stack_lvl+0x91/0xf0 lib/dump_stack.c:106 print_address_description.constprop.0+0x66/0x300 mm/kasan/report.c:364 print_report+0x3e/0x70 mm/kasan/report.c:475

CVE	Severity	CVSS	CVSS Vector	Component	Description
					kasan_report+0xb8/0xf0 mm/kasan/report.c:588 blk_mq_find_and_get_req+0x16e/0x1a0 block/blk-mq-tag.c:261 bt_iter block/blk-mq-tag.c:288 [inline] __sbitmap_for_each_set include/linux/sbitmap.h:295 [inline] sbitmap_for_each_set include/linux/sbitmap.h:316 [inline] bt_for_each+0x455/0x790 block/blk-mq-tag.c:325 blk_mq_queue_tag_busy_iter+0x320/0x740 block/blk-mq-tag.c:534 blk_mq_timeout_work+0x1a3/0x7b0 block/blk-mq.c:1673 process_one_work+0x7c4/0x1450 kernel/workqueue.c:2631 process_scheduled_works kernel/workqueue.c:2704 [inline] worker_thread+0x804/0xe40 kernel/workqueue.c:2785 kthread+0x346/0x450 kernel/kthread.c:388 ret_from_fork+0x4d/0x80 arch/x86/kernel/process.c:147 ret_from_fork_asm+0x1b/0x30 arch/x86/entry/entry_64.S:293 Allocated by task 942: kasan_save_stack+0x22/0x50 mm/kasan/common.c:45 kasan_set_track+0x25/0x30 mm/kasan/common.c:52 ____kasan_kmalloc mm/kasan/common.c:374 [inline] __kasan_kmalloc mm/kasan/common.c:383 [inline] __kasan_kmalloc+0xaa/0xb0 mm/kasan/common.c:380 kasan_kmalloc include/linux/kasan.h:198 [inline] __do_kmalloc_node mm/slab_common.c:1007 [inline] __kmalloc_node+0x69/0x170 mm/slab_common.c:1014 kmalloc_node include/linux/slab.h:620 [inline] kzalloc_node include/linux/slab.h:732 [inline] blk_alloc_flush_queue+0x144/0x2f0 block/blk-flush.c:499 blk_mq_alloc_hctx+0x601/0x940 block/blk-mq.c:3788 blk_mq_alloc_and_init_hctx+0x27f/0x330 block/blk-mq.c:4261 blk_mq_realloc_hw_ctxs+0x488/0x5e0 block/blk-mq.c:4294 blk_mq_init_allocated_queue+0x188/0x860 block/blk-mq.c:4350 blk_mq_init_queue_data block/blk-mq.c:4166 [inline] blk_mq_init_queue+0x8d/0x100 block/blk-mq.c:4176 scsi_alloc_sdev+0x843/0xd50 drivers/scsi/scsi_scan.c:335 scsi_probe_and_add_lun+0x77c/0xde0 drivers/scsi/scsi_scan.c:1189 __scsi_scan_target+0x1fc/0x5a0 drivers/scsi/scsi_scan.c:1727 scsi_scan_channel drivers/scsi/scsi_scan.c:1815 [inline] scsi_scan_channel+0x14b/0x1e0 drivers/scsi/scsi_scan.c:1791 scsi_scan_host_selected+0x2fe/0x400 drivers/scsi/scsi_scan.c:1844 scsi_scan+0x3a0/0x3f0 drivers/scsi/scsi_sysfs.c:151 store_scan+0x2a/0x60 drivers/scsi/scsi_sysfs.c:191 dev_attr_store+0x5c/0x90 drivers/base/core.c:2388 sysfs_kf_write+0x11c/0x170 fs/sysfs/file.c:136 kernfs_fop_write_iter+0x3fc/0x610 fs/kernfs/file.c:338 call_write_iter include/linux/fs.h:2083 [inline] new_sync_write+0x1b4/0x2d0 fs/read_write.c:493 vfs_write+0x76c/0xb00 fs/read_write.c:586 ksys_write+0x127/0x250 fs/read_write.c:639 do_syscall_x64 arch/x86/entry/common.c:51 [inline] do_syscall_64+0x70/0x120 arch/x86/entry/common.c:81 entry_SYSCALL_64_after_hwframe+0x78/0xe2 Freed by task 244687: kasan_save_stack+0x22/0x50 mm/kasan/common.c:45 kasan_set_track+0x25/0x30 mm/kasan/common.c:52 kasan_save_free_info+0x2b/0x50 mm/kasan/generic.c:522 ____kasan_slab_free mm/kasan/common.c:236 [inline] __kasan_slab_free+0x12a/0x1b0 mm/kasan/common.c:244 kasan_slab_free include/linux/kasan.h:164 [in ---truncated---

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2024-41045	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: bpf: Defer work in bpf_timer_cancel_and_free Currently, the same case as previous patch (two timer callbacks trying to cancel each other) can be invoked through bpf_map_update_elem as well, or more precisely, freeing map elements containing timers. Since this relies on hrtimer_cancel as well, it is prone to the same deadlock situation as the previous patch. It would be sufficient to use hrtimer_try_to_cancel to fix this problem, as the timer cannot be enqueued after async_cancel_and_free. Once async_cancel_and_free has been done, the timer must be reinitialized before it can be armed again. The callback running in parallel trying to arm the timer will fail, and freeing bpf_hrtimer without waiting is sufficient (given kfree_rcu), and bpf_timer_cb will return HRTIMER_NORESTART, preventing the timer from being rearmed again. However, there exists a UAF scenario where the callback arms the timer before entering this function, such that if cancellation fails (due to timer callback invoking this routine, or the target timer callback running concurrently). In such a case, if the timer expiration is significantly far in the future, the RCU grace period expiration happening before it will free the bpf_hrtimer state and along with it the struct hrtimer, that is enqueued. Hence, it is clear cancellation needs to occur after async_cancel_and_free, and yet it cannot be done inline due to deadlock issues. We thus modify bpf_timer_cancel_and_free to defer work to the global workqueue, adding a work_struct alongside rcu_head (both used at _different_ points of time, so can share space). Update existing code comments to reflect the new state of affairs.
CVE-2026-23221	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: bus: fsl-mc: fix use-after-free in driver_override_show() The driver_override_show() function reads the driver_override string without holding the device_lock. However, driver_override_store() uses driver_set_override(), which modifies and frees the string while holding the device_lock. This can result in a concurrent use-after-free if the string is freed by the store function while being read by the show function. Fix this by holding the device_lock around the read operation.
CVE-2025-39873	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: can: xilinx_can: xcan_write_frame(): fix use-after-free of transmitted SKB can_put_echo_skb() takes ownership of the SKB and it may be freed during or after the call. However, xilinx_can xcan_write_frame() keeps using SKB after the call. Fix that by only calling can_put_echo_skb() after the code is done touching the SKB. The tx_lock is held for the entire xcan_write_frame() execution and also on the can_get_echo_skb() side so the order of operations does not matter. An earlier fix commit 3d3c817c3a40 ("can: xilinx_can: Fix usage of skb memory") did not move the can_put_echo_skb() call far enough. [mkl: add "commit" in front of sha1 in patch description] [mkl: fix indention]

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-39945	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: cnic: Fix use-after-free bugs in cnic_delete_task The original code uses cancel_delayed_work() in cnic_cm_stop_bnx2x_hw(), which does not guarantee that the delayed work item 'delete_task' has fully completed if it was already running. Additionally, the delayed work item is cyclic, the flush_workqueue() in cnic_cm_stop_bnx2x_hw() only blocks and waits for work items that were already queued to the workqueue prior to its invocation. Any work items submitted after flush_workqueue() is called are not included in the set of tasks that the flush operation awaits. This means that after the cyclic work items have finished executing, a delayed work item may still exist in the workqueue. This leads to use-after-free scenarios where the cnic_dev is deallocated by cnic_free_dev(), while delete_task remains active and attempt to dereference cnic_dev in cnic_delete_task(). A typical race condition is illustrated below: CPU 0 (cleanup) | CPU 1 (delayed work callback) cnic_netdev_event() | cnic_stop_hw() | cnic_delete_task() cnic_cm_stop_bnx2x_hw() | ... cancel_delayed_work() | /* the queue_delayed_work() flush_workqueue() | executes after flush_workqueue()*/ | queue_delayed_work() cnic_free_dev(dev)//free | cnic_delete_task() //new instance | dev = cp->dev; //use Replace cancel_delayed_work() with cancel_delayed_work_sync() to ensure that the cyclic delayed work item is properly canceled and that any ongoing execution of the work item completes before the cnic_dev is deallocated. Furthermore, since cancel_delayed_work_sync() uses __flush_work(work, true) to synchronously wait for any currently executing instance of the work item to finish, the flush_workqueue() becomes redundant and should be removed. This bug was identified through static analysis. To reproduce the issue and validate the fix, I simulated the cnic PCI device in QEMU and introduced intentional delays — such as inserting calls to ssleep() within the cnic_delete_task() function — to increase the likelihood of triggering the bug.
CVE-2025-39871	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: dmaengine: idxd: Remove improper idxd_free The call to idxd_free() introduces a duplicate put_device() leading to a reference count underflow: refcount_t: underflow; use-after-free. WARNING: CPU: 15 PID: 4428 at lib/refcount.c:28 refcount_warn_saturate+0xbe/0x110 ... Call Trace: idxd_remove+0xe4/0x120 [idxd] pci_device_remove+0x3f/0xb0 device_release_driver_internal+0x197/0x200 driver_detach+0x48/0x90 bus_remove_driver+0x74/0xf0 pci_unregister_driver+0x2e/0xb0 idxd_exit_module+0x34/0x7a0 [idxd] __do_sys_delete_module.constprop.0+0x183/0x280 do_syscall_64+0x54/0xd70 entry_SYSCALL_64_after_hwframe+0x76/0x7e The idxd_unregister_devices() which is invoked at the very beginning of idxd_remove(), already takes care of the necessary put_device() through the following call path: idxd_unregister_devices() -> device_unregister() -> put_device() In addition, when CONFIG_DEBUG_KOBJECT_RELEASE is enabled, put_device() may trigger asynchronous cleanup via schedule_delayed_work(). If idxd_free() is called immediately after, it can result in a use-after-free. Remove the improper idxd_free() to avoid both the refcount underflow and potential memory corruption during module unload.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-71162	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: dmaengine: tegra-adma: Fix use-after-free A use-after-free bug exists in the Tegra ADMA driver when audio streams are terminated, particularly during XRUN conditions. The issue occurs when the DMA buffer is freed by tegra_adma_terminate_all() before the vchan completion tasklet finishes accessing it. The race condition follows this sequence: 1. DMA transfer completes, triggering an interrupt that schedules the completion tasklet (tasklet has not executed yet) 2. Audio playback stops, calling tegra_adma_terminate_all() which frees the DMA buffer memory via kfree() 3. The scheduled tasklet finally executes, calling vchan_complete() which attempts to access the already-freed memory Since tasklets can execute at any time after being scheduled, there is no guarantee that the buffer will remain valid when vchan_complete() runs. Fix this by properly synchronizing the virtual channel completion: - Calling vchan_terminate_vdesc() in tegra_adma_stop() to mark the descriptors as terminated instead of freeing the descriptor. - Add the callback tegra_adma_synchronize() that calls vchan_synchronize() which kills any pending tasklets and frees any terminated descriptors. Crash logs: [337.427523] BUG: KASAN: use-after-free in vchan_complete+0x124/0x3b0 [337.427544] Read of size 8 at addr ffff000132055428 by task swapper/0/0 [337.427562] Call trace: [337.427564] dump_backtrace+0x0/0x320 [337.427571] show_stack+0x20/0x30 [337.427575] dump_stack_lvl+0x68/0x84 [337.427584] print_address_description.constprop.0+0x74/0x2b8 [337.427590] kasan_report+0x1f4/0x210 [337.427598] __asan_load8+0xa0/0xd0 [337.427603] vchan_complete+0x124/0x3b0 [337.427609] tasklet_action_common.constprop.0+0x190/0x1d0 [337.427617] tasklet_action+0x30/0x40 [337.427623] __do_softirq+0x1a0/0x5c4 [337.427628] irq_exit+0x110/0x140 [337.427633] handle_domain_irq+0xa4/0xe0 [337.427640] gic_handle_irq+0x64/0x160 [337.427644] call_on_irq_stack+0x20/0x4c [337.427649] do_interrupt_handler+0x7c/0x90 [337.427654] el1_interrupt+0x30/0x80 [337.427659] el1h_64_irq_handler+0x18/0x30 [337.427663] el1h_64_irq+0x7c/0x80 [337.427667] cpuidle_enter_state+0xe4/0x540 [337.427674] cpuidle_enter+0x54/0x80 [337.427679] do_idle+0x2e0/0x380 [337.427685] cpu_startup_entry+0x2c/0x70 [337.427690] rest_init+0x114/0x130 [337.427695] arch_call_rest_init+0x18/0x24 [337.427702] start_kernel+0x380/0x3b4 [337.427706] __primary_switched+0xc0/0xc8

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38708	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: drbd: add missing kref_get in handle_write_conflicts With `two-primaries` enabled, DRBD tries to detect "concurrent" writes and handle write conflicts, so that even if you write to the same sector simultaneously on both nodes, they end up with the identical data once the writes are completed. In handling "superseded" writes, we forgot a kref_get, resulting in a premature drbd_destroy_device and use after free, and further to kernel crashes with symptoms. Relevance: No one should use DRBD as a random data generator, and apparently all users of "two-primaries" handle concurrent writes correctly on layer up. That is cluster file systems use some distributed lock manager, and live migration in virtualization environments stops writes on one node before starting writes on the other node. Which means that other than for "test cases", this code path is never taken in real life. FYI, in DRBD 9, things are handled differently nowadays. We still detect "write conflicts", but no longer try to be smart about them. We decided to disconnect hard instead: upper layers must not submit concurrent writes. If they do, that's their fault.
CVE-2025-38349	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: eventpoll: don't decrement ep refcount while still holding the ep mutex Jann Horn points out that epoll is decrementing the ep refcount and then doing a mutex_unlock(&ep->mtx); afterwards. That's very wrong, because it can lead to a use-after-free. That pattern is actually fine for the very last reference, because the code in question will delay the actual call to "ep_free(ep)" until after it has unlocked the mutex. But it's wrong for the much subtler "next to last" case when somebody *else* may also be dropping their reference and free the ep while we're still using the mutex. Note that this is true even if that other user is also using the same ep mutex: mutexes, unlike spinlocks, can not be used for object ownership, even if they guarantee mutual exclusion. A mutex "unlock" operation is not atomic, and as one user is still accessing the mutex as part of unlocking it, another user can come in and get the now released mutex and free the data structure while the first user is still cleaning up. See our mutex documentation in Documentation/locking/mutex-design.rst, in particular the section [1] about semantics: "mutex_unlock() may access the mutex structure even after it has internally released the lock already - so it's not safe for another context to acquire the mutex and assume that the mutex_unlock() context is not using the structure anymore" So if we drop our ep ref before the mutex unlock, but we weren't the last one, we may then unlock the mutex, another user comes in, drops _their_ reference and releases the 'ep' as it now has no users - all while the mutex_unlock() is still accessing it. Fix this by simply moving the ep refcount dropping to outside the mutex: the refcount itself is atomic, and doesn't need mutex protection (that's the whole _point_ of refcounts: unlike mutexes, they are inherently about object lifetimes).

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38627	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: f2fs: compress: fix UAF of f2fs_inode_info in f2fs_free_dic The decompress_io_ctx may be released asynchronously after I/O completion. If this file is deleted immediately after read, and the kworker of processing post_read_wq has not been executed yet due to high workloads, It is possible that the inode(f2fs_inode_info) is evicted and freed before it is used f2fs_free_dic. The UAF case as below: Thread A Thread B - f2fs_decompress_end_io - f2fs_put_dic - queue_work add free_dic work to post_read_wq - do_unlink - iput - evict - call_rcu This file is deleted after read. Thread C kworker to process post_read_wq - rcu_do_batch - f2fs_free_inode - kmem_cache_free inode is freed by rcu - process_scheduled_works - f2fs_late_free_dic - f2fs_free_dic - f2fs_release_decomp_mem read (dic->inode)->i_compress_algorithm This patch store compress_algorithm and sbi in dic to avoid inode UAF. In addition, the previous solution is deprecated in [1] may cause system hang. [1] https://lore.kernel.org/all/c36ab955-c8db-4a8b-a9d0-f07b5f426c3f@kernel.org
CVE-2026-23234	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: f2fs: fix to avoid UAF in f2fs_write_end_io() As syzbot reported an use-after-free issue in f2fs_write_end_io(). It is caused by below race condition: loop device umount - worker_thread - loop_process_work - do_req_filebacked - lo_rw_aio - lo_rw_aio_complete - blk_mq_end_request - blk_update_request - f2fs_write_end_io - dec_page_count - folio_end_writeback - kill_f2fs_super - kill_block_super - f2fs_put_super : free(sbi) : get_pages(, F2FS_WB_CP_DATA) accessed sbi which is freed In kill_f2fs_super(), we will drop all page caches of f2fs inodes before call free(sbi), it guarantee that all folios should end its writeback, so it should be safe to access sbi before last folio_end_writeback(). Let's relocate ckpt thread wakeup flow before folio_end_writeback() to resolve this issue.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-39691	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: fs/buffer: fix use-after-free when call bh_read() helper There's issue as follows: BUG: KASAN: stack-out-of-bounds in end_buffer_read_sync+0xe3/0x110 Read of size 8 at addr ffffc9000168f7f8 by task swapper/3/0 CPU: 3 UID: 0 PID: 0 Comm: swapper/3 Not tainted 6.16.0-862.14.0.6.x86_64 Hardware name: QEMU Standard PC (i440FX + PIIX, 1996) Call Trace: dump_stack_lvl+0x55/0x70 print_address_description.constprop.0+0x2c/0x390 print_report+0xb4/0x270 kasan_report+0xb8/0xf0 end_buffer_read_sync+0xe3/0x110 end_bio_bh_io_sync+0x56/0x80 blk_update_request+0x30a/0x720 scsi_end_request+0x51/0x2b0 scsi_io_completion+0xe3/0x480 ? scsi_device_unbusy+0x11e/0x160 blk_complete_reqs+0x7b/0x90 handle_softirqs+0xef/0x370 irq_exit_rcu+0xa5/0xd0 sysvec_apic_timer_interrupt+0x6e/0x90 Above issue happens when do ntfs3 filesystem mount, issue may happens as follows: mount IRQ ntfs_fill_super read_cache_page do_read_cache_folio filemap_read_folio mpage_read_folio do_mpage_readpage ntfs_get_block_vbo bh_read submit_bh wait_on_buffer(bh); blk_complete_reqs scsi_io_completion scsi_end_request blk_update_request end_bio_bh_io_sync end_buffer_read_sync __end_buffer_read_notouch unlock_buffer wait_on_buffer(bh);--> return will return to caller put_bh --> trigger stack-out-of-bounds In the mpage_read_folio() function, the stack variable 'map_bh' is passed to ntfs_get_block_vbo(). Once unlock_buffer() unlocks and wait_on_buffer() returns to continue processing, the stack variable is likely to be reclaimed. Consequently, during the end_buffer_read_sync() process, calling put_bh() may result in stack overrun. If the bh is not allocated on the stack, it belongs to a folio. Freeing a buffer head which belongs to a folio is done by drop_buffers() which will fail to free buffers which are still locked. So it is safe to call put_bh() before __end_buffer_read_notouch().

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-39866	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: fs: writeback: fix use-after-free in __mark_inode_dirty() An use-after-free issue occurred when __mark_inode_dirty() get the bdi_writeback that was in the progress of switching. CPU: 1 PID: 562 Comm: systemd-random- Not tainted 6.6.56-gb4403bd46a8e #1 ..... pstate: 60400005 (nZCv daif +PAN -UAO -TCO -DIT -SSBS BTYPE=--) pc : __mark_inode_dirty+0x124/0x418 lr : __mark_inode_dirty+0x118/0x418 sp : ffffffc08c9dbbc0 ..... Call trace: __mark_inode_dirty+0x124/0x418 generic_update_time+0x4c/0x60 file_modified+0xcc/0xd0 ext4_buffered_write_iter+0x58/0x124 ext4_file_write_iter+0x54/0x704 vfs_write+0x1c0/0x308 ksys_write+0x74/0x10c __arm64_sys_write+0x1c/0x28 invoke_syscall+0x48/0x114 el0_svc_common.constprop.0+0xc0/0xe0 do_el0_svc+0x1c/0x28 el0_svc+0x40/0xe4 el0t_64_sync_handler+0x120/0x12c el0t_64_sync+0x194/0x198 Root cause is: systemd-random-seed kworker ---------- ___mark_inode_dirty inode_switch_wbs_work_fn spin_lock(&inode->i_lock); inode_attach_wb locked_inode_to_wb_and_lock_list get inode->i_wb spin_unlock(&inode->i_lock); spin_lock(&wb->list_lock) spin_lock(&inode->i_lock) inode_io_list_move_locked spin_unlock(&wb->list_lock) spin_unlock(&inode->i_lock) spin_lock(&old_wb->list_lock) inode_do_switch_wbs spin_lock(&inode->i_lock) inode->i_wb = new_wb spin_unlock(&inode->i_lock) spin_unlock(&old_wb->list_lock) wb_put_many(old_wb, nr_switched) cgwb_release old wb released wb_wakeup_delayed() accesses wb, then trigger the use-after-free issue Fix this race condition by holding inode spinlock until wb_wakeup_delayed() finished.
CVE-2025-39689	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ftrace: Also allocate and copy hash for reading of filter files Currently the reader of set_ftrace_filter and set_ftrace_notrace just adds the pointer to the global tracer hash to its iterator. Unlike the writer that allocates a copy of the hash, the reader keeps the pointer to the filter hashes. This is problematic because this pointer is static across function calls that release the locks that can update the global tracer hashes. This can cause UAF and similar bugs. Allocate and copy the hash for reading the filter files like it is done for the writers. This not only fixes UAF bugs, but also makes the code a bit simpler as it doesn't have to differentiate when to free the iterator's hash between writers and readers.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38722	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: habanalabs: fix UAF in export_dmabuf() As soon as we'd inserted a file reference into descriptor table, another thread could close it. That's fine for the case when all we are doing is returning that descriptor to userland (it's a race, but it's a userland race and there's nothing the kernel can do about it). However, if we follow fd_install() with any kind of access to objects that would be destroyed on close (be it the struct file itself or anything destroyed by its ->release()), we have a UAF. dma_buf_fd() is a combination of reserving a descriptor and fd_install(). habanalabs export_dmabuf() calls it and then proceeds to access the objects destroyed on close. In particular, it grabs an extra reference to another struct file that will be dropped as part of ->release() for ours; that "will be" is actually "might have already been". Fix that by reserving descriptor before anything else and do fd_install() only when everything had been set up. As a side benefit, we no longer have the failure exit with file already created, but reference to underlying file (as well as ->dmabuf_export_cnt, etc.) not grabbed yet; unlike dma_buf_fd(), fd_install() can't fail.
CVE-2025-38485	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: iio: accel: fxls8962af: Fix use after free in fxls8962af_fifo_flush fxls8962af_fifo_flush() uses indio_dev->active_scan_mask (with iio_for_each_active_channel()) without making sure the indio_dev stays in buffer mode. There is a race if indio_dev exits buffer mode in the middle of the interrupt that flushes the fifo. Fix this by calling synchronize_irq() to ensure that no interrupt is currently running when disabling buffer mode. Unable to handle kernel NULL pointer dereference at virtual address 00000000 when read [...] _find_first_bit_le from fxls8962af_fifo_flush+0x17c/0x290 fxls8962af_fifo_flush from fxls8962af_interrupt+0x80/0x178 fxls8962af_interrupt from irq_thread_fn+0x1c/0x7c irq_thread_fn from irq_thread+0x110/0x1f4 irq_thread from kthread+0xe0/0xfc kthread from ret_from_fork+0x14/0x2c
CVE-2025-71071	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: iommu/mediatek: fix use-after-free on probe deferral The driver is dropping the references taken to the larb devices during probe after successful lookup as well as on errors. This can potentially lead to a use-after-free in case a larb device has not yet been bound to its driver so that the iommu driver probe defers. Fix this by keeping the references as expected while the iommu driver is bound.
CVE-2026-23010	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ipv6: Fix use-after-free in inet6_addr_del(). syzbot reported use-after-free of inet6_ifaddr in inet6_addr_del(). [0] The cited commit accidentally moved ipv6_del_addr() for mngtmpaddr before reading its ifp->flags for temporary addresses in inet6_addr_del(). Let's move ipv6_del_addr() down to fix the UAF. [0]: BUG: KASAN: slab-use-after-free in inet6_addr_del.constprop.0+0x67a/0x6b0 net/ipv6/addrconf.c:3117 Read of size 4 at addr ffff88807b89c86c by task syz.3.1618/9593 CPU: 0 UID: 0 PID: 9593 Comm: syz.3.1618 Not tainted syzkaller #0 PREEMPT(full) Hardware name: Google Google Compute Engine/Google Compute Engine, BIOS Google 10/25/2025 Call Trace: __dump_stack lib/dump_stack.c:94 [inline] dump_stack_lvl+0x116/0x1f0 lib/dump_stack.c:120 print_address_description mm/kasan/report.c:378 [inline] print_report+0xcd/0x630 mm/kasan/report.c:482 kasan_report+0xe0/0x110 mm/kasan/report.c:595 inet6_addr_del.constprop.0+0x67a/0x6b0

CVE	Severity	CVSS	CVSS Vector	Component	Description
					net/ipv6/addrconf.c:3117 addrconf_del_ifaddr+0x11e/0x190 net/ipv6/addrconf.c:3181 inet6_ioctl+0x1e5/0x2b0 net/ipv6/af_inet6.c:582 sock_do_ioctl+0x118/0x280 net/socket.c:1254 sock_ioctl+0x227/0x6b0 net/socket.c:1375 vfs_ioctl fs/ioctl.c:51 [inline] __do_sys_ioctl fs/ioctl.c:597 [inline] __se_sys_ioctl fs/ioctl.c:583 [inline] __x64_sys_ioctl+0x18e/0x210 fs/ioctl.c:583 do_syscall_x64 arch/x86/entry/syscall_64.c:63 [inline] do_syscall_64+0xcd/0xf80 arch/x86/entry/syscall_64.c:94 entry_SYSCALL_64_after_hwframe+0x77/0x7f RIP: 0033:0x7f164cf8f749 Code: ff ff c3 66 2e 0f 1f 84 00 00 00 00 00 0f 1f 40 00 48 89 f8 48 89 f7 48 89 d6 48 89 ca 4d 89 c2 4d 89 c8 4c 8b 4c 24 08 0f 05 <48> 3d 01 f0 ff ff 73 01 c3 48 c7 c1 a8 ff ff ff f7 d8 64 89 01 48 RSP: 002b:00007f164de64038 EFLAGS: 00000246 ORIG_RAX: 0000000000000010 RAX: ffffffffffffffda RBX: 00007f164d1e5fa0 RCX: 00007f164cf8f749 RDX: 0000200000000000 RSI: 0000000000008936 RDI: 0000000000000003 RBP: 00007f164d013f91 R08: 0000000000000000 R09: 0000000000000000 R10: 0000000000000000 R11: 0000000000000246 R12: 0000000000000000 R13: 00007f164d1e6038 R14: 00007f164d1e5fa0 R15: 00007ffde15c8288 Allocated by task 9593: kasan_save_stack+0x33/0x60 mm/kasan/common.c:56 kasan_save_track+0x14/0x30 mm/kasan/common.c:77 poison_kmalloc_redzone mm/kasan/common.c:397 [inline] __kasan_kmalloc+0xaa/0xb0 mm/kasan/common.c:414 kmalloc_noprof include/linux/slab.h:957 [inline] kzalloc_noprof include/linux/slab.h:1094 [inline] ipv6_add_addr+0x4e3/0x2010 net/ipv6/addrconf.c:1120 inet6_addr_add+0x256/0x9b0 net/ipv6/addrconf.c:3050 addrconf_add_ifaddr+0x1fc/0x450 net/ipv6/addrconf.c:3160 inet6_ioctl+0x103/0x2b0 net/ipv6/af_inet6.c:580 sock_do_ioctl+0x118/0x280 net/socket.c:1254 sock_ioctl+0x227/0x6b0 net/socket.c:1375 vfs_ioctl fs/ioctl.c:51 [inline] __do_sys_ioctl fs/ioctl.c:597 [inline] __se_sys_ioctl fs/ioctl.c:583 [inline] __x64_sys_ioctl+0x18e/0x210 fs/ioctl.c:583 do_syscall_x64 arch/x86/entry/syscall_64.c:63 [inline] do_syscall_64+0xcd/0xf80 arch/x86/entry/syscall_64.c:94 entry_SYSCALL_64_after_hwframe+0x77/0x7f Freed by task 6099: kasan_save_stack+0x33/0x60 mm/kasan/common.c:56 kasan_save_track+0x14/0x30 mm/kasan/common.c:77 kasan_save_free_info+0x3b/0x60 mm/kasan/generic.c:584 poison_slab_object mm/kasan/common.c:252 [inline] __kasan_slab_free+0x5f/0x80 mm/kasan/common.c:284 kasan_slab_free include/linux/kasan.h:234 [inline] slab_free_hook mm/slub.c:2540 [inline] slab_free_freelist_hook mm/slub.c:2569 [inline] slab_free_bulk mm/slub.c:6696 [inline] kmem_cache_free_bulk mm/slub.c:7383 [inline] kmem_cache_free_bulk+0x2bf/0x680 mm/slub.c:7362 kfree_bulk include/linux/slab.h:830 [inline] kvfree_rcu_bulk+0x1b7/0x1e0 mm/slab_common.c:1523 kvfree_rcu_drain_ready mm/slab_common.c:1728 [inline] kfree_rcu_monitor+0x1d0/0x2f0 mm/slab_common.c:1801 process_one_work+0x9ba/0x1b20 kernel/workqueue.c:3257 process_scheduled_works kernel/workqu ---truncated---

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-39881	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: kernfs: Fix UAF in polling when open file is released A use-after-free (UAF) vulnerability was identified in the PSI (Pressure Stall Information) monitoring mechanism: BUG: KASAN: slab-use-after-free in psi_trigger_poll+0x3c/0x140 Read of size 8 at addr ffff3de3d50bd308 by task systemd/1 psi_trigger_poll+0x3c/0x140 cgroup_pressure_poll+0x70/0xa0 cgroup_file_poll+0x8c/0x100 kernfs_fop_poll+0x11c/0x1c0 ep_item_poll.isra.0+0x188/0x2c0 Allocated by task 1: cgroup_file_open+0x88/0x388 kernfs_fop_open+0x73c/0xaf0 do_dentry_open+0x5fc/0x1200 vfs_open+0xa0/0x3f0 do_open+0x7e8/0xd08 path_openat+0x2fc/0x6b0 do_filp_open+0x174/0x368 Freed by task 8462: cgroup_file_release+0x130/0x1f8 kernfs_drain_open_files+0x17c/0x440 kernfs_drain+0x2dc/0x360 kernfs_show+0x1b8/0x288 cgroup_file_show+0x150/0x268 cgroup_pressure_write+0x1dc/0x340 cgroup_file_write+0x274/0x548 Reproduction Steps: 1. Open test/cpu.pressure and establish epoll monitoring 2. Disable monitoring: echo 0 > test/cgroup.pressure 3. Re-enable monitoring: echo 1 > test/cgroup.pressure The race condition occurs because: 1. When cgroup.pressure is disabled (echo 0 > cgroup.pressure), it: - Releases PSI triggers via cgroup_file_release() - Frees of->priv through kernfs_drain_open_files() 2. While epoll still holds reference to the file and continues polling 3. Re-enabling (echo 1 > cgroup.pressure) accesses freed of->priv epolling disable/enable cgroup.pressure fd=open(cpu.pressure) while(1) ... epoll_wait kernfs_fop_poll kernfs_get_active = true echo 0 > cgroup.pressure ... cgroup_file_show kernfs_show // inactive kn kernfs_drain_open_files cft->release(of); kfree(ctx); ... kernfs_get_active = false echo 1 > cgroup.pressure kernfs_show kernfs_activate_one(kn); kernfs_fop_poll kernfs_get_active = true cgroup_file_poll psi_trigger_poll // UAF ... end: close(fd) To address this issue, introduce kernfs_get_active_of() for kernfs open files to obtain active references. This function will fail if the open file has been released. Replace kernfs_get_active() with kernfs_get_active_of() to prevent further operations on released file descriptors.
CVE-2025-38437	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ksmbd: fix potential use-after-free in oplock/lease break ack If ksmbd_iov_pin_rsp return error, use-after-free can happen by accessing opinfo->state and opinfo_put and ksmbd_fd_put could called twice.
CVE-2025-37777	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ksmbd: fix use-after-free in __smb2_lease_break_noti() Move tcp_transport free to ksmbd_conn_free. If ksmbd connection is referenced when ksmbd server thread terminates, It will not be freed, but conn->tcp_transport is freed. __smb2_lease_break_noti can be performed asynchronously when the connection is disconnected. __smb2_lease_break_noti calls ksmbd_conn_write, which can cause use-after-free when conn->ksmbd_transport is already freed.
CVE-2025-37926	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ksmbd: fix use-after-free in ksmbd_session_rpc_open A UAF issue can occur due to a race condition between ksmbd_session_rpc_open() and __session_rpc_close(). Add rpc_lock to the session to protect it.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-68817	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ksmbd: fix use-after-free in ksmbd_tree_connect_put under concurrency Under high concurrency, A tree-connection object (tcon) is freed on a disconnect path while another path still holds a reference and later executes *_put()/write on it.
CVE-2025-37899	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ksmbd: fix use-after-free in session logoff The sess->user object can currently be in use by another thread, for example if another connection has sent a session setup request to bind to the session being free'd. The handler for that connection could be in the smb2_sess_setup function which makes use of sess->user.
CVE-2026-23209	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: macvlan: fix error recovery in macvlan_common_newlink() valis provided a nice repro to crash the kernel: ip link add p1 type veth peer p2 ip link set address 00:00:00:00:00:20 dev p1 ip link set up dev p1 ip link set up dev p2 ip link add mv0 link p2 type macvlan mode source ip link add invalid% link p2 type macvlan mode source macaddr add 00:00:00:00:00:20 ping -c1 -I p1 1.2.3.4 He also gave a very detailed analysis: The issue is triggered when a new macvlan link is created with MACVLAN_MODE_SOURCE mode and MACVLAN_MACADDR_ADD (or MACVLAN_MACADDR_SET) parameter, lower device already has a macvlan port and register_netdevice() called from macvlan_common_newlink() fails (e.g. because of the invalid link name). In this case macvlan_hash_add_source is called from macvlan_change_sources() / macvlan_common_newlink(): This adds a reference to vlan to the port's vlan_source_hash using macvlan_source_entry. vlan is a pointer to the priv data of the link that is being created. When register_netdevice() fails, the error is returned from macvlan_newlink() to rtnl_newlink_create(): if (ops->newlink) err = ops->newlink(dev, ¶ms, extack); else err = register_netdevice(dev); if (err < 0) { free_netdev(dev); goto out; } and free_netdev() is called, causing a kvfree() on the struct net_device that is still referenced in the source entry attached to the lower device's macvlan port. Now all packets sent on the macvlan port with a matching source mac address will trigger a use-after-free in macvlan_forward_source(). With all that, my fix is to make sure we call macvlan_flush_sources() regardless of @create value whenever "goto destroy_macvlan_port;" path is taken. Many thanks to valis for following up on this issue.
CVE-2026-23001	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: macvlan: fix possible UAF in macvlan_forward_source() Add RCU protection on (struct macvlan_source_entry)->vlan. Whenever macvlan_hash_del_source() is called, we must clear entry->vlan pointer before RCU grace period starts. This allows macvlan_forward_source() to skip over entries queued for freeing. Note that macvlan_dev are already RCU protected, as they are embedded in a standard netdev (netdev_priv(ndev)). https://lore.kernel.org/netdev/695fb1e8.050a0220.1c677c.039f.GAE@google.com/T/#u

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-39711	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: media: ivsc: Fix crash at shutdown due to missing mei_cldev_disable() calls Both the ACE and CSI driver are missing a mei_cldev_disable() call in their remove() function. This causes the mei_cl client to stay part of the mei_device->file_list list even though its memory is freed by mei_cl_bus_dev_release() calling kfree(cldev->cl). This leads to a use-after-free when mei_vsc_remove() runs mei_stop() which first removes all mei bus devices calling mei_ace_remove() and mei_csi_remove() followed by mei_cl_bus_dev_release() and then calls mei_cl_all_disconnect() which walks over mei_device->file_list dereferecing the just freed cldev->cl. And mei_vsc_remove() it self is run at shutdown because of the platform_device_unregister(tp->pdev) in vsc_tp_shutdown() When building a kernel with KASAN this leads to the following KASAN report: [106.634504] == [106.634623] BUG: KASAN: slab-use-after-free in mei_cl_set_disconnected (drivers/misc/mei/client.c:783) mei [106.634683] Read of size 4 at addr ffff88819cb62018 by task systemd-shutdow/1 [106.634729] [106.634767] Tainted: [E]=UNSIGNED_MODULE [106.634770] Hardware name: Dell Inc. XPS 16 9640/09CK4V, BIOS 1.12.0 02/10/2025 [106.634773] Call Trace: [106.634777] ... [106.634871] kasan_report (mm/kasan/report.c:221 mm/kasan/report.c:636) [106.634901] mei_cl_set_disconnected (drivers/misc/mei/client.c:783) mei [106.634921] mei_cl_all_disconnect (drivers/misc/mei/client.c:2165 (discriminator 4)) mei [106.634941] mei_reset (drivers/misc/mei/init.c:163) mei ... [106.635042] mei_stop (drivers/misc/mei/init.c:348) mei [106.635062] mei_vsc_remove (drivers/misc/mei/mei_dev.h:784 drivers/misc/mei/platform-vsc.c:393) mei_vsc [106.635066] platform_remove (drivers/base/platform.c:1424) Add the missing mei_cldev_disable() calls so that the mei_cl gets removed from mei_device->file_list before it is freed to fix this.
CVE-2025-39877	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: mm/damon/sysfs: fix use-after-free in state_show() state_show() reads kdamond->damon_ctx without holding damon_sysfs_lock. This allows a use-after-free race: CPU 0 CPU 1 ----- ----- state_show() damon_sysfs_turn_damon_on() ctx = kdamond->damon_ctx; mutex_lock(&damon_sysfs_lock); damon_destroy_ctx(kdamond->damon_ctx); kdamond->damon_ctx = NULL; mutex_unlock(&damon_sysfs_lock); damon_is_running(ctx); /* ctx is freed */ mutex_lock(&ctx->kdamond_lock); /* UAF */ (The race can also occur with damon_sysfs_kdamonds_rm_dirs() and damon_sysfs_kdamond_release(), which free or replace the context under damon_sysfs_lock.) Fix by taking damon_sysfs_lock before dereferencing the context, mirroring the locking used in pid_show(). The bug has existed since state_show() first accessed kdamond->damon_ctx.
CVE-2025-39776	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: mm/debug_vm_pgtable: clear page table entries at destroy_args() The mm/debug_vm_pagetable test allocates manually page table entries for the tests it runs, using also its manually allocated mm_struct. That in itself is ok, but when it exits, at destroy_args() it fails to clear those entries with the *_clear functions. The problem is that leaves stale entries. If another process allocates an mm_struct with a pgd at the same address, it

CVE	Severity	CVSS	CVSS Vector	Component	Description
					may end up running into the stale entry. This is happening in practice on a debug kernel with CONFIG_DEBUG_VM_PGTABLE=y, for example this is the output with some extra debugging I added (it prints a warning trace if pgtables_bytes goes negative, in addition to the warning at check_mm() function): [2.539353] debug_vm_pgtable: [get_random_vaddr]: random_vaddr is 0x7ea247140000 [2.539366] kmem_cache info [2.539374] kmem_cachep 0x000000002ce82385 - freelist 0x0000000000000000 - offset 0x508 [2.539447] debug_vm_pgtable: [init_args]: args->mm is 0x000000002267cc9e (...) [2.552800] WARNING: CPU: 5 PID: 116 at include/linux/mm.h:2841 free_pud_range+0x8bc/0x8d0 [2.552816] Modules linked in: [2.552843] CPU: 5 UID: 0 PID: 116 Comm: modprobe Not tainted 6.12.0-105.debug_vm2.el10.ppc64le+debug #1 VOLUNTARY [2.552859] Hardware name: IBM,9009-41A POWER9 (architected) 0x4e0202 0xf000005 of:IBM,FW910.00 (VL910_062) hv:phyp pSeries [2.552872] NIP: c0000000007eef3c LR: c0000000007eef30 CTR: c0000000003d8c90 [2.552885] REGS: c0000000622e73b0 TRAP: 0700 Not tainted (6.12.0-105.debug_vm2.el10.ppc64le+debug) [2.552899] MSR: 800000000282b033 CR: 24002822 XER: 0000000a [2.552954] CFAR: c0000000008f03f0 IRQMASK: 0 [2.552954] GPR00: c0000000007eef30 c0000000622e7650 c000000002b1ac00 0000000000000001 [2.552954] GPR04: 0000000000000008 0000000000000000 c0000000007eef30 ffffffffffffffff [2.552954] GPR08: 00000000ffff00f5 0000000000000001 0000000000000048 0000000000004000 [2.552954] GPR12: 00000003fa440000 c000000017ffa300 c0000000051d9f80 fffffffffffffdb [2.552954] GPR16: 0000000000000000 0000000000000008 000000000000000a 60000000000000e0 [2.552954] GPR20: 4080000000000000 c0000000113af038 00007fffcf130000 0000700000000000 [2.552954] GPR24: c000000062a6a000 0000000000000001 8000000062a68000 0000000000000001 [2.552954] GPR28: 000000000000000a c000000062ebc600 0000000000002000 c000000062ebc760 [2.553170] NIP [c0000000007eef3c] free_pud_range+0x8bc/0x8d0 [2.553185] LR [c0000000007eef30] free_pud_range+0x8b0/0x8d0 [2.553199] Call Trace: [2.553207] [c0000000622e7650] [c0000000007eef30] free_pud_range+0x8b0/0x8d0 (unreliable) [2.553229] [c0000000622e7750] [c0000000007f40b4] free_pgd_range+0x284/0x3b0 [2.553248] [c0000000622e7800] [c0000000007f4630] free_pgtables+0x450/0x570 [2.553274] [c0000000622e78e0] [c0000000008161c0] exit_mmap+0x250/0x650 [2.553292] [c0000000622e7a30] [c0000000001b95b8] __mmput+0x98/0x290 [2.558344] [c0000000622e7a80] [c0000000001d1018] exit_mm+0x118/0x1b0 [2.558361] [c0000000622e7ac0] [c0000000001d141c] do_exit+0x2ec/0x870 [2.558376] [c0000000622e7b60] [c0000000001d1ca8] do_group_exit+0x88/0x150 [2.558391] [c0000000622e7bb0] [c0000000001d1db8] sys_exit_group+0x48/0x50 [2.558407] [c0000000622e7be0] [c00000000003d810] system_call_exception+0x1e0/0x4c0 [2.558423] [c0000000622e7e50] [c00000000000d05c] system_call_vectored_common+0x15c/0x2ec (...) [2.558892] ---[end trace 0000000000000000]--- [2.559022] BUG: Bad rss-counter state mm:000000002267cc9e type:MM_ANONPAGES val:1 [2.559037] BUG: non-zero pgtables_bytes on freeing mm: -6144 Here the modprobe process ended up with an allocated mm_struct from the mm_struct slab that was used before by the debug_vm_pgtable test. That is not a problem, since the mm_stru ---truncated---

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38443	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: nbd: fix uaf in nbd_genl_connect() error path There is a use-after-free issue in nbd: block nbd6: Receive control failed (result -104) block nbd6: shutting down sockets == BUG: KASAN: slab-use-after-free in recv_work+0x694/0xa80 drivers/block/nbd.c:1022 Write of size 4 at addr ffff8880295de478 by task kworker/u33:0/67 CPU: 2 UID: 0 PID: 67 Comm: kworker/u33:0 Not tainted 6.15.0-rc5-syzkaller-00123-g2c89c1b655c0 #0 PREEMPT(full) Hardware name: QEMU Standard PC (Q35 + ICH9, 2009), BIOS 1.16.3-debian-1.16.3-2~bpo12+1 04/01/2014 Workqueue: nbd6-recv recv_work Call Trace: __dump_stack lib/dump_stack.c:94 [inline] dump_stack_lvl+0x116/0x1f0 lib/dump_stack.c:120 print_address_description mm/kasan/report.c:408 [inline] print_report+0xc3/0x670 mm/kasan/report.c:521 kasan_report+0xe0/0x110 mm/kasan/report.c:634 check_region_inline mm/kasan/generic.c:183 [inline] kasan_check_range+0xef/0x1a0 mm/kasan/generic.c:189 instrument_atomic_read_write include/linux/instrumented.h:96 [inline] atomic_dec include/linux/atomic/atomic-instrumented.h:592 [inline] recv_work+0x694/0xa80 drivers/block/nbd.c:1022 process_one_work+0x9cc/0x1b70 kernel/workqueue.c:3238 process_scheduled_works kernel/workqueue.c:3319 [inline] worker_thread+0x6c8/0xf10 kernel/workqueue.c:3400 kthread+0x3c2/0x780 kernel/kthread.c:464 ret_from_fork+0x45/0x80 arch/x86/kernel/process.c:153 ret_from_fork_asm+0x1a/0x30 arch/x86/entry/entry_64.S:245 nbd_genl_connect() does not properly stop the device on certain error paths after nbd_start_device() has been called. This causes the error path to put nbd->config while recv_work continue to use the config after putting it, leading to use-after-free in recv_work. This patch moves nbd_start_device() after the backend file creation.
CVE-2025-21751	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net/mlx5: HWS, change error flow on matcher disconnect Currently, when firmware failure occurs during matcher disconnect flow, the error flow of the function reconnects the matcher back and returns an error, which continues running the calling function and eventually frees the matcher that is being disconnected. This leads to a case where we have a freed matcher on the matchers list, which in turn leads to use-after-free and eventual crash. This patch fixes that by not trying to reconnect the matcher back when some FW command fails during disconnect. Note that we're dealing here with FW error. We can't overcome this problem. This might lead to bad steering state (e.g. wrong connection between matchers), and will also lead to resource leakage, as it is the case with any other error handling during resource destruction. However, the goal here is to allow the driver to continue and not crash the machine with use-after-free error.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38350	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net/sched: Always pass notifications when child class becomes empty Certain classful qdiscs may invoke their classes' dequeue handler on an enqueue operation. This may unexpectedly empty the child qdisc and thus make an in-flight class passive via qlen_notify(). Most qdiscs do not expect such behaviour at this point in time and may re-activate the class eventually anyways which will lead to a use-after-free. The referenced fix commit attempted to fix this behavior for the HFSC case by moving the backlog accounting around, though this turned out to be incomplete since the parent's parent may run into the issue too. The following reproducer demonstrates this use-after-free: tc qdisc add dev lo root handle 1: drr tc filter add dev lo parent 1: basic classid 1:1 tc class add dev lo parent 1: classid 1:1 drr tc qdisc add dev lo parent 1:1 handle 2: hfsc def 1 tc class add dev lo parent 2: classid 2:1 hfsc rt m1 8 d 1 m2 0 tc qdisc add dev lo parent 2:1 handle 3: netem tc qdisc add dev lo parent 3:1 handle 4: blackhole echo 1 | socat -u STDIN UDP4-DATAGRAM:127.0.0.1:8888 tc class delete dev lo classid 1:1 echo 1 | socat -u STDIN UDP4-DATAGRAM:127.0.0.1:8888 Since backlog accounting issues leading to a use-after-frees on stale class pointers is a recurring pattern at this point, this patch takes a different approach. Instead of trying to fix the accounting, the patch ensures that qdisc_tree_reduce_backlog always calls qlen_notify when the child qdisc is empty. This solves the problem because deletion of qdiscs always involves a call to qdisc_reset() and / or qdisc_purge_queue() which ultimately resets its qlen to 0 thus causing the following qdisc_tree_reduce_backlog() to report to the parent. Note that this may call qlen_notify on passive classes multiple times. This is not a problem after the recent patch series that made all the classful qdiscs qlen_notify() handlers idempotent.
CVE-2026-23074	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net/sched: Enforce that teql can only be used as root qdisc Design intent of teql is that it is only supposed to be used as root qdisc. We need to check for that constraint. Although not important, I will describe the scenario that unearthed this issue for the curious. GangMin Kim managed to concot a scenario as follows: ROOT qdisc 1:0 (QFQ) ├── class 1:1 (weight=15, lmax=16384) netem with delay 6.4s └── class 1:2 (weight=1, lmax=1514) teql GangMin sends a packet which is enqueued to 1:1 (netem). Any invocation of dequeue by QFQ from this class will not return a packet until after 6.4s. In the meantime, a second packet is sent and it lands on 1:2. teql's enqueue will return success and this will activate class 1:2. Main issue is that teql only updates the parent visible qlen (sch->q.qlen) at dequeue. Since QFQ will only call dequeue if peek succeeds (and teql's peek always returns NULL), dequeue will never be called and thus the qlen will remain as 0. With that in mind, when GangMin updates 1:2's lmax value, the qfq_change_class calls qfq_deact_rm_from_agg. Since the child qdisc's qlen was not incremented, qfq fails to deactivate the class, but still frees its pointers from the aggregate. So when the first packet is rescheduled after 6.4 seconds (netem's delay), a dangling pointer is accessed causing GangMin's causing a UAF.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38734	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net/smc: fix UAF on smcsk after smc_listen_out() BPF CI testing report a UAF issue: [16.446633] BUG: kernel NULL pointer dereference, address: 000000000000003 0 [16.447134] #PF: supervisor read access in kernel mod e [16.447516] #PF: error_code(0x0000) - not-present pag e [16.447878] PGD 0 P4D 0 [16.448063] Oops: Oops: 0000 [#1] PREEMPT SMP NOPT I [16.448409] CPU: 0 UID: 0 PID: 9 Comm: kworker/0:1 Tainted: G OE 6.13.0-rc3-g89e8a75fda73-dirty #4 2 [16.449124] Tainted: [O]=OOT_MODULE, [E]=UNSIGNED_MODUL E [16.449502] Hardware name: QEMU Ubuntu 24.04 PC (i440FX + PIIX, 1996), BIOS 1.16.3-debian-1.16.3-2 04/01/201 4 [16.450201] Workqueue: smc_hs_wq smc_listen_wor k [16.450531] RIP: 0010:smc_listen_work+0xc02/0x159 0 [16.452158] RSP: 0018:ffffb5ab40053d98 EFLAGS: 0001024 6 [16.452526] RAX: 0000000000000001 RBX: 0000000000000002 RCX: 000000000000030 0 [16.452994] RDX: 0000000000000280 RSI: 00003513840053f0 RDI: 000000000000000 0 [16.453492] RBP: ffffa097808e3800 R08: ffffa09782dba1e0 R09: 000000000000000 5 [16.453987] R10: 0000000000000000 R11: 0000000000000000 R12: ffffa0978274640 0 [16.454497] R13: 0000000000000000 R14: 0000000000000000 R15: ffffa09782d4092 0 [16.454996] FS: 0000000000000000(0000) GS:ffffa097bbc00000(0000) knlGS:000000000000000 0 [16.455557] CS: 0010 DS: 0000 ES: 0000 CR0: 000000008005003 3 [16.455961] CR2: 0000000000000030 CR3: 0000000102788004 CR4: 0000000000770ef 0 [16.456459] PKRU: 5555555 4 [16.456654] Call Trace : [16.456832] [16.456989] ? __die+0x23/0x7 0 [16.457215] ? page_fault_oops+0x180/0x4c 0 [16.457508] ? __lock_acquire+0x3e6/0x249 0 [16.457801] ? exc_page_fault+0x68/0x20 0 [16.458080] ? asm_exc_page_fault+0x26/0x3 0 [16.458389] ? smc_listen_work+0xc02/0x159 0 [16.458689] ? smc_listen_work+0xc02/0x159 0 [16.458987] ? lock_is_held_type+0x8f/0x10 0 [16.459284] process_one_work+0x1ea/0x6d 0 [16.459570] worker_thread+0x1c3/0x38 0 [16.459839] ? __pfx_worker_thread+0x10/0x1 0 [16.460144] kthread+0xe0/0x11 0 [16.460372] ? __pfx_kthread+0x10/0x1 0 [16.460640] ret_from_fork+0x31/0x5 0 [16.460896] ? __pfx_kthread+0x10/0x1 0 [16.461166] ret_from_fork_asm+0x1a/0x3 0 [16.461453] [16.461616] Modules linked in: bpf_testmod(OE) [last unloaded: bpf_testmod(OE)] [16.462134] CR2: 000000000000003 0 [16.462380] ---[end trace 0000000000000000]--- [16.462710] RIP: 0010:smc_listen_work+0xc02/0x1590 The direct cause of this issue is that after smc_listen_out_connected(), newclcsock->sk may be NULL since it will releases the smcsk. Therefore, if the application closes the socket immediately after accept, newclcsock->sk can be NULL. A possible execution order could be as follows: smc_listen_work | userspace ---------- ---------- lock_sock(sk) | smc_listen_out_connected() | |- smc_listen_out | | |- release_sock | |- sk->sk_data_ready() | | fd = accept(); | close(fd); | |- socket->sk = NULL; /* newclcsock->sk is NULL now */ SMC_STAT_SERV_SUCC_INC(sock_net(newclcsock->sk)) Since smc_listen_out_connected() will not fail, simply swapping the order of the code can easily fix this issue.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38666	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net: appletalk: Fix use-after-free in AARP proxy probe The AARP proxy-probe routine (aarp_proxy_probe_network) sends a probe, releases the aarp_lock, sleeps, then re-acquires the lock. During that window an expire timer thread (__aarp_expire_timer) can remove and kfree() the same entry, leading to a use-after-free. race condition: cpu 0 | cpu 1 atalk_sendmsg() | atif_proxy_probe_device() aarp_send_ddp() | aarp_proxy_probe_network() mod_timer() | lock(aarp_lock) // LOCK!! timeout around 200ms | alloc(aarp_entry) and then call | proxies[hash] = aarp_entry aarp_expire_timeout() | aarp_send_probe() | unlock(aarp_lock) // UNLOCK!! lock(aarp_lock) // LOCK!! | msleep(100); __aarp_expire_timer(&proxies[ct]) | free(aarp_entry) | unlock(aarp_lock) // UNLOCK!! | | lock(aarp_lock) // LOCK!! | UAF aarp_entry !! == BUG: KASAN: slab-use-after-free in aarp_proxy_probe_network+0x560/0x630 net/appletalk/aarp.c:493 Read of size 4 at addr ffff8880123aa360 by task repro/13278 CPU: 3 UID: 0 PID: 13278 Comm: repro Not tainted 6.15.2 #3 PREEMPT(full) Call Trace: __dump_stack lib/dump_stack.c:94 [inline] dump_stack_lvl+0x116/0x1b0 lib/dump_stack.c:120 print_address_description mm/kasan/report.c:408 [inline] print_report+0xc1/0x630 mm/kasan/report.c:521 kasan_report+0xca/0x100 mm/kasan/report.c:634 aarp_proxy_probe_network+0x560/0x630 net/appletalk/aarp.c:493 atif_proxy_probe_device net/appletalk/ddp.c:332 [inline] atif_ioctl+0xb58/0x16c0 net/appletalk/ddp.c:857 atalk_ioctl+0x198/0x2f0 net/appletalk/ddp.c:1818 sock_do_ioctl+0xdc/0x260 net/socket.c:1190 sock_ioctl+0x239/0x6a0 net/socket.c:1311 vfs_ioctl fs/ioctl.c:51 [inline] __do_sys_ioctl fs/ioctl.c:906 [inline] __se_sys_ioctl fs/ioctl.c:892 [inline] __x64_sys_ioctl+0x194/0x200 fs/ioctl.c:892 do_syscall_x64 arch/x86/entry/syscall_64.c:63 [inline] do_syscall_64+0xcb/0x250 arch/x86/entry/syscall_64.c:94 entry_SYSCALL_64_after_hwframe+0x77/0x7f Allocated: aarp_alloc net/appletalk/aarp.c:382 [inline] aarp_proxy_probe_network+0xd8/0x630 net/appletalk/aarp.c:468 atif_proxy_probe_device net/appletalk/ddp.c:332 [inline] atif_ioctl+0xb58/0x16c0 net/appletalk/ddp.c:857 atalk_ioctl+0x198/0x2f0 net/appletalk/ddp.c:1818 Freed: kfree+0x148/0x4d0 mm/slub.c:4841 __aarp_expire net/appletalk/aarp.c:90 [inline] __aarp_expire_timer net/appletalk/aarp.c:261 [inline] aarp_expire_timeout+0x480/0x6e0 net/appletalk/aarp.c:317 The buggy address belongs to the object at ffff8880123aa300 which belongs to the cache kmalloc-192 of size 192 The buggy address is located 96 bytes inside of freed 192-byte region [ffff8880123aa300, ffff8880123aa3c0) Memory state around the buggy address: ffff8880123aa200: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ffff8880123aa280: 00 00 00 00 fc fc fc fc fc fc fc fc fc fc fc fc >ffff8880123aa300: fa fb fb fb fb fb fb fb fb fb fb fb fb fb fb fb ^ ffff8880123aa380: fb fb fb fb fb fb fb fb fc fc fc fc fc fc fc fc ffff8880123aa400: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ==

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2026-23111	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: netfilter: nf_tables: fix inverted genmask check in nft_map_catchall_activate() nft_map_catchall_activate() has an inverted element activity check compared to its non-catchall counterpart nft_mapelem_activate() and compared to what is logically required. nft_map_catchall_activate() is called from the abort path to re-activate catchall map elements that were deactivated during a failed transaction. It should skip elements that are already active (they don't need re-activation) and process elements that are inactive (they need to be restored). Instead, the current code does the opposite: it skips inactive elements and processes active ones. Compare the non-catchall activate callback, which is correct: nft_mapelem_activate(): if (nft_set_elem_active(ext, iter->genmask)) return 0; /* skip active, process inactive */ With the buggy catchall version: nft_map_catchall_activate(): if (!nft_set_elem_active(ext, genmask)) continue; /* skip inactive, process active */ The consequence is that when a DELSET operation is aborted, nft_setelem_data_activate() is never called for the catchall element. For NFT_GOTO verdict elements, this means nft_data_hold() is never called to restore the chain->use reference count. Each abort cycle permanently decrements chain->use. Once chain->use reaches zero, DELCHAIN succeeds and frees the chain while catchall verdict elements still reference it, resulting in a use-after-free. This is exploitable for local privilege escalation from an unprivileged user via user namespaces + nftables on distributions that enable CONFIG_USER_NS and CONFIG_NF_TABLES. Fix by removing the negation so the check matches nft_mapelem_activate(): skip active elements, process inactive ones.
CVE-2025-38724	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: nfsd: handle get_client_locked() failure in nfsd4_setclientid_confirm() Lei Lu recently reported that nfsd4_setclientid_confirm() did not check the return value from get_client_locked(). a SETCLIENTID_CONFIRM could race with a confirmed client expiring and fail to get a reference. That could later lead to a UAF. Fix this by getting a reference early in the case where there is an extant confirmed client. If that fails then treat it as if there were no confirmed client found at all. In the case where the unconfirmed client is expiring, just fail and return the result from get_client_locked().

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2026-22980	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: nfsd: provide locking for v4_end_grace Writing to v4_end_grace can race with server shutdown and result in memory being accessed after it was freed - reclaim_str_hashtbl in particularly. We cannot hold nfsd_mutex across the nfsd4_end_grace() call as that is held while client_tracking_op->init() is called and that can wait for an upcall to nfsdcltrack which can write to v4_end_grace, resulting in a deadlock. nfsd4_end_grace() is also called by the landromat work queue and this doesn't require locking as server shutdown will stop the work and wait for it before freeing anything that nfsd4_end_grace() might access. However, we must be sure that writing to v4_end_grace doesn't restart the work item after shutdown has already waited for it. For this we add a new flag protected with nn->client_lock. It is set only while it is safe to make client tracking calls, and v4_end_grace only schedules work while the flag is set with the spinlock held. So this patch adds a nfsd_net field "client_tracking_active" which is set as described. Another field "grace_end_forced", is set when v4_end_grace is written. After this is set, and providing client_tracking_active is set, the laundromat is scheduled. This "grace_end_forced" field bypasses other checks for whether the grace period has finished. This resolves a race which can result in use-after-free.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-39944	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: octeontx2-pf: Fix use-after-free bugs in otx2_sync_tstamp() The original code relies on cancel_delayed_work() in otx2_ptp_destroy(), which does not ensure that the delayed work item synctstamp_work has fully completed if it was already running. This leads to use-after-free scenarios where otx2_ptp is deallocated by otx2_ptp_destroy(), while synctstamp_work remains active and attempts to dereference otx2_ptp in otx2_sync_tstamp(). Furthermore, the synctstamp_work is cyclic, the likelihood of triggering the bug is nonnegligible. A typical race condition is illustrated below: CPU 0 (cleanup) | CPU 1 (delayed work callback) otx2_remove() | otx2_ptp_destroy() | otx2_sync_tstamp() cancel_delayed_work() | kfree(ptp) | | ptp = container_of(...); //UAF | ptp-> //UAF This is confirmed by a KASAN report: BUG: KASAN: slab-use-after-free in __run_timer_base.part.0+0x7d7/0x8c0 Write of size 8 at addr ffff88800aa09a18 by task bash/136 ... Call Trace: dump_stack_lvl+0x55/0x70 print_report+0xcf/0x610 ? __run_timer_base.part.0+0x7d7/0x8c0 kasan_report+0xb8/0xf0 ? __run_timer_base.part.0+0x7d7/0x8c0 __run_timer_base.part.0+0x7d7/0x8c0 ? __pfx___run_timer_base.part.0+0x10/0x10 ? __pfx_read_tsc+0x10/0x10 ? ktime_get+0x60/0x140 ? lapic_next_event+0x11/0x20 ? clockevents_program_event+0x1d4/0x2a0 run_timer_softirq+0xd1/0x190 handle_softirqs+0x16a/0x550 irq_exit_rcu+0xaf/0xe0 sysvec_apic_timer_interrupt+0x70/0x80 ... Allocated by task 1: kasan_save_stack+0x24/0x50 kasan_save_track+0x14/0x30 __kasan_kmalloc+0x7f/0x90 otx2_ptp_init+0xb1/0x860 otx2_probe+0x4eb/0xc30 local_pci_probe+0xdc/0x190 pci_device_probe+0x2fe/0x470 really_probe+0x1ca/0x5c0 __driver_probe_device+0x248/0x310 driver_probe_device+0x44/0x120 __driver_attach+0xd2/0x310 bus_for_each_dev+0xed/0x170 bus_add_driver+0x208/0x500 driver_register+0x132/0x460 do_one_initcall+0x89/0x300 kernel_init_freeable+0x40d/0x720 kernel_init+0x1a/0x150 ret_from_fork+0x10c/0x1a0 ret_from_fork_asm+0x1a/0x30 Freed by task 136: kasan_save_stack+0x24/0x50 kasan_save_track+0x14/0x30 kasan_save_free_info+0x3a/0x60 __kasan_slab_free+0x3f/0x50 kfree+0x137/0x370 otx2_ptp_destroy+0x38/0x80 otx2_remove+0x10d/0x4c0 pci_device_remove+0xa6/0x1d0 device_release_driver_internal+0xf8/0x210 pci_stop_bus_device+0x105/0x150 pci_stop_and_remove_bus_device_locked+0x15/0x30 remove_store+0xcc/0xe0 kernfs_fop_write_iter+0x2c3/0x440 vfs_write+0x871/0xd70 ksys_write+0xee/0x1c0 do_syscall_64+0xac/0x280 entry_SYSCALL_64_after_hwframe+0x77/0x7f ... Replace cancel_delayed_work() with cancel_delayed_work_sync() to ensure that the delayed work item is properly canceled before the otx2_ptp is deallocated. This bug was initially identified through static analysis. To reproduce and test it, I simulated the OcteonTX2 PCI device in QEMU and introduced artificial delays within the otx2_sync_tstamp() function to increase the likelihood of triggering the bug.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38129	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: page_pool: Fix use-after-free in page_pool_recycle_in_ring syzbot reported a uaf in page_pool_recycle_in_ring: BUG: KASAN: slab-use-after-free in lock_release+0x151/0xa30 kernel/locking/lockdep.c:5862 Read of size 8 at addr ffff8880286045a0 by task syz.0.284/6943 CPU: 0 UID: 0 PID: 6943 Comm: syz.0.284 Not tainted 6.13.0-rc3-syzkaller-gdfa94ce54f41 #0 Hardware name: Google Google Compute Engine/Google Compute Engine, BIOS Google 09/13/2024 Call Trace: __dump_stack lib/dump_stack.c:94 [inline] dump_stack_lvl+0x241/0x360 lib/dump_stack.c:120 print_address_description mm/kasan/report.c:378 [inline] print_report+0x169/0x550 mm/kasan/report.c:489 kasan_report+0x143/0x180 mm/kasan/report.c:602 lock_release+0x151/0xa30 kernel/locking/lockdep.c:5862 __raw_spin_unlock_bh include/linux/spinlock_api_smp.h:165 [inline] _raw_spin_unlock_bh+0x1b/0x40 kernel/locking/spinlock.c:210 spin_unlock_bh include/linux/spinlock.h:396 [inline] ptr_ring_produce_bh include/linux/ptr_ring.h:164 [inline] page_pool_recycle_in_ring net/core/page_pool.c:707 [inline] page_pool_put_unrefed_netmem+0x748/0xb00 net/core/page_pool.c:826 page_pool_put_netmem include/net/page_pool/helpers.h:323 [inline] page_pool_put_full_netmem include/net/page_pool/helpers.h:353 [inline] napi_pp_put_page+0x149/0x2b0 net/core/skbuff.c:1036 skb_pp_recycle net/core/skbuff.c:1047 [inline] skb_free_head net/core/skbuff.c:1094 [inline] skb_release_data+0x6c4/0x8a0 net/core/skbuff.c:1125 skb_release_all net/core/skbuff.c:1190 [inline] __kfree_skb net/core/skbuff.c:1204 [inline] sk_skb_reason_drop+0x1c9/0x380 net/core/skbuff.c:1242 kfree_skb_reason include/linux/skbuff.h:1263 [inline] __skb_queue_purge_reason include/linux/skbuff.h:3343 [inline] root cause is: page_pool_recycle_in_ring ptr_ring_produce spin_lock(&r->producer_lock); WRITE_ONCE(r->queue[r->producer++], ptr) //recycle last page to pool page_pool_release page_pool_scrub page_pool_empty_ring ptr_ring_consume page_pool_return_page //release all page __page_pool_destroy free_percpu(pool->recycle_stats); free(pool) //free spin_unlock(&r->producer_lock); //pool->ring uaf read recycle_stat_inc(pool, ring); page_pool can be free while page pool recycle the last page in ring. Add producer-lock barrier to page_pool_release to prevent the page pool from being free before all pages have been recycled. recycle_stat_inc() is empty when CONFIG_PAGE_POOL_STATS is not enabled, which will trigger Wempty-body build warning. Add definition for pool stat macro to fix warning.
CVE-2025-38653	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: proc: use the same treatment to check proc_lseek as ones for proc_read_iter et.al Check pde->proc_ops->proc_lseek directly may cause UAF in rmmod scenario. It's a gap in proc_reg_open() after commit 654b33ada4ab("proc: fix UAF in proc_get_inode()"). Followed by Al Viro's suggestion, fix it in same manner.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-39859	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ptp: ocp: fix use-after-free bugs causing by ptp_ocp_watchdog The ptp_ocp_detach() only shuts down the watchdog timer if it is pending. However, if the timer handler is already running, the timer_delete_sync() is not called. This leads to race conditions where the devlink that contains the ptp_ocp is deallocated while the timer handler is still accessing it, resulting in use-after-free bugs. The following details one of the race scenarios. (thread 1) | (thread 2) ptp_ocp_remove() | ptp_ocp_detach() | ptp_ocp_watchdog() if (timer_pending(&bp->watchdog))| bp = timer_container_of() timer_delete_sync() | | devlink_free(devlink) //free | | bp-> //use Resolve this by unconditionally calling timer_delete_sync() to ensure the timer is reliably deactivated, preventing any access after free.
CVE-2025-38377	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: rose: fix dangling neighbour pointers in rose_rt_device_down() There are two bugs in rose_rt_device_down() that can cause use-after-free: 1. The loop bound `t->count` is modified within the loop, which can cause the loop to terminate early and miss some entries. 2. When removing an entry from the neighbour array, the subsequent entries are moved up to fill the gap, but the loop index `i` is still incremented, causing the next entry to be skipped. For example, if a node has three neighbours (A, A, B) with count=3 and A is being removed, the second A is not checked. i=0: (A, A, B) -> (A, B) with count=2 ^ checked i=1: (A, B) -> (A, B) with count=2 ^ checked (B, not A!) i=2: (doesn't occur because i < count is false) This leaves the second A in the array with count=2, but the rose_neigh structure has been freed. Code that accesses these entries assumes that the first `count` entries are valid pointers, causing a use-after-free when it accesses the dangling pointer. Fix both issues by iterating over the array in reverse order with a fixed loop bound. This ensures that all entries are examined and that the removal of an entry doesn't affect subsequent iterations.
CVE-2025-38476	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: rpl: Fix use-after-free in rpl_do_srh_inline(). Running lwt_dst_cache_ref_loop.sh in selftest with KASAN triggers the splat below [0]. rpl_do_srh_inline() fetches ipv6_hdr(skb) and accesses it after skb_cow_head(), which is illegal as the header could be freed then. Let's fix it by making oldhdr to a local struct instead of a pointer. [0]: [root@fedora net]# ./lwt_dst_cache_ref_loop.sh ... TEST: rpl (input) [57.631529] == BUG: KASAN: slab-use-after-free in rpl_do_srh_inline.isra.0 (net/ipv6/rpl_iptunnel.c:174) Read of size 40 at addr ffff888122bf96d8 by task ping6/1543 CPU: 50 UID: 0 PID: 1543 Comm: ping6 Not tainted 6.16.0-rc5-01302-gfadd1e6231b1 #23 PREEMPT(voluntary) Hardware name: QEMU Standard PC (i440FX + PIIX, 1996), BIOS 1.16.3-debian-1.16.3-2 04/01/2014 Call Trace: dump_stack_lvl (lib/dump_stack.c:122) print_report (mm/kasan/report.c:409 mm/kasan/report.c:521) kasan_report (mm/kasan/report.c:221 mm/kasan/report.c:636) kasan_check_range (mm/kasan/generic.c:175 (discriminator 1) mm/kasan/generic.c:189 (discriminator 1)) __asan_memmove (mm/kasan/shadow.c:94 (discriminator 2)) rpl_do_srh_inline.isra.0 (net/ipv6/rpl_iptunnel.c:174) rpl_input (net/ipv6/rpl_iptunnel.c:201 net/ipv6/rpl_iptunnel.c:282) lwtunnel_input (net/core/lwtunnel.c:459) ipv6_rcv (./include/net/dst.h:471 (discriminator 1)

CVE	Severity	CVSS	CVSS Vector	Component	Description
					./include/net/dst.h:469 (discriminator 1) net/ipv6/ip6_input.c:79 (discriminator 1) ./include/linux/netfilter.h:317 (discriminator 1) ./include/linux/netfilter.h:311 (discriminator 1) net/ipv6/ip6_input.c:311 (discriminator 1)) __netif_receive_skb_one_core (net/core/dev.c:5967) process_backlog (./include/linux/rcupdate.h:869 net/core/dev.c:6440) __napi_poll.constprop.0 (net/core/dev.c:7452) net_rx_action (net/core/dev.c:7518 net/core/dev.c:7643) handle_softirqs (kernel/softirq.c:579) do_softirq (kernel/softirq.c:480 (discriminator 20)) __local_bh_enable_ip (kernel/softirq.c:407) __dev_queue_xmit (net/core/dev.c:4740) ip6_finish_output2 (./include/linux/netdevice.h:3358 ./include/net/neighbour.h:526 ./include/net/neighbour.h:540 net/ipv6/ip6_output.c:141) ip6_finish_output (net/ipv6/ip6_output.c:215 net/ipv6/ip6_output.c:226) ip6_output (./include/linux/netfilter.h:306 net/ipv6/ip6_output.c:248) ip6_send_skb (net/ipv6/ip6_output.c:1983) rawv6_sendmsg (net/ipv6/raw.c:588 net/ipv6/raw.c:918) __sys_sendto (net/socket.c:714 (discriminator 1) net/socket.c:729 (discriminator 1) net/socket.c:2228 (discriminator 1)) __x64_sys_sendto (net/socket.c:2231) do_syscall_64 (arch/x86/entry/syscall_64.c:63 (discriminator 1) arch/x86/entry/syscall_64.c:94 (discriminator 1)) entry_SYSCALL_64_after_hwframe (arch/x86/entry/entry_64.S:130) RIP: 0033:0x7f68cffb2a06 Code: 5d e8 41 8b 93 08 03 00 00 59 5e 48 83 f8 fc 75 19 83 e2 39 83 fa 08 75 11 e8 26 ff ff ff 66 0f 1f 44 00 00 48 8b 45 10 0f 05 <48> 8b 5d f8 c9 c3 0f 1f 40 00 f3 0f 1e fa 55 48 89 e5 48 83 ec 08 RSP: 002b:00007ffefb7c53d0 EFLAGS: 00000202 ORIG_RAX: 000000000000002c RAX: ffffffffffffffda RBX: 0000564cd69f10a0 RCX: 00007f68cffb2a06 RDX: 0000000000000040 RSI: 0000564cd69f10a4 RDI: 0000000000000003 RBP: 00007ffefb7c53f0 R08: 0000564cd6a032ac R09: 000000000000001c R10: 0000000000000000 R11: 0000000000000202 R12: 0000564cd69f10a4 R13: 0000000000000040 R14: 00007ffefb7c66e0 R15: 0000564cd69f10a0 Allocated by task 1543: kasan_save_stack (mm/kasan/common.c:48) kasan_save_track (mm/kasan/common.c:60 (discriminator 1) mm/kasan/common.c:69 (discriminator 1)) __kasan_slab_alloc (mm/kasan/common.c:319 mm/kasan/common.c:345) kmem_cache_alloc_node_noprof (./include/linux/kasan.h:250 mm/slub.c:4148 mm/slub.c:4197 mm/slub.c:4249) kmalloc_reserve (net/core/skbuff.c:581 (discriminator 88)) __alloc_skb (net/core/skbuff.c:669) __ip6_append_data (net/ipv6/ip6_output.c:1672 (discriminator 1)) ip6_ ---truncated---
CVE-2025-71075	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: scsi: aic94xx: fix use-after-free in device removal path The asd_pci_remove() function fails to synchronize with pending tasklets before freeing the asd_ha structure, leading to a potential use-after-free vulnerability. When a device removal is triggered (via hot-unplug or module unload), race condition can occur. The fix adds tasklet_kill() before freeing the asd_ha structure, ensuring all scheduled tasklets complete before cleanup proceeds.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-37861	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: scsi: mpi3mr: Synchronous access b/w reset and tm thread for reply queue When the task management thread processes reply queues while the reset thread resets them, the task management thread accesses an invalid queue ID (0xFFFF), set by the reset thread, which points to unallocated memory, causing a crash. Add flag 'io_admin_reset_sync' to synchronize access between the reset, I/O, and admin threads. Before a reset, the reset handler sets this flag to block I/O and admin processing threads. If any thread bypasses the initial check, the reset thread waits up to 10 seconds for processing to finish. If the wait exceeds 10 seconds, the controller is marked as unrecoverable.
CVE-2026-23216	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: scsi: target: iscsi: Fix use-after-free in iscsit_dec_conn_usage_count() In iscsit_dec_conn_usage_count(), the function calls complete() while holding the conn->conn_usage_lock. As soon as complete() is invoked, the waiter (such as iscsit_close_connection()) may wake up and proceed to free the iscsit_conn structure. If the waiter frees the memory before the current thread reaches spin_unlock_bh(), it results in a KASAN slab-use-after-free as the function attempts to release a lock within the already-freed connection structure. Fix this by releasing the spinlock before calling complete().
CVE-2026-23193	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: scsi: target: iscsi: Fix use-after-free in iscsit_dec_session_usage_count() In iscsit_dec_session_usage_count(), the function calls complete() while holding the sess->session_usage_lock. Similar to the connection usage count logic, the waiter signaled by complete() (e.g., in the session release path) may wake up and free the iscsit_session structure immediately. This creates a race condition where the current thread may attempt to execute spin_unlock_bh() on a session structure that has already been deallocated, resulting in a KASAN slab-use-after-free. To resolve this, release the session_usage_lock before calling complete() to ensure all dereferences of the sess pointer are finished before the waiter is allowed to proceed with deallocation.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-37750	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: smb: client: fix UAF in decryption with multichannel After commit f7025d861694 ("smb: client: allocate crypto only for primary server") and commit b0abcd65ec54 ("smb: client: fix UAF in async decryption"), the channels started reusing AEAD TFM from primary channel to perform synchronous decryption, but that can't done as there could be multiple cifsd threads (one per channel) simultaneously accessing it to perform decryption. This fixes the following KASAN splat when running fstest generic/249 with 'vers=3.1.1,multichannel,max_channels=4,seal' against Windows Server 2022: BUG: KASAN: slab-use-after-free in gf128mul_4k_lle+0xba/0x110 Read of size 8 at addr ffff8881046c18a0 by task cifsd/986 CPU: 3 UID: 0 PID: 986 Comm: cifsd Not tainted 6.15.0-rc1 #1 PREEMPT(voluntary) Hardware name: QEMU Standard PC (Q35 + ICH9, 2009), BIOS 1.16.3-3.fc41 04/01/2014 Call Trace: dump_stack_lvl+0x5d/0x80 print_report+0x156/0x528 ? gf128mul_4k_lle+0xba/0x110 ? __virt_addr_valid+0x145/0x300 ? __phys_addr+0x46/0x90 ? gf128mul_4k_lle+0xba/0x110 kasan_report+0xdf/0x1a0 ? gf128mul_4k_lle+0xba/0x110 gf128mul_4k_lle+0xba/0x110 ghash_update+0x189/0x210 shash_ahash_update+0x295/0x370 ? __pfx_shash_ahash_update+0x10/0x10 ? __pfx_shash_ahash_update+0x10/0x10 ? __pfx_extract_iter_to_sg+0x10/0x10 ? ___kmalloc_large_node+0x10e/0x180 ? __asan_memset+0x23/0x50 crypto_ahash_update+0x3c/0xc0 gcm_hash_assoc_remain_continue+0x93/0xc0 crypt_message+0xe09/0xec0 [cifs] ? __pfx_crypt_message+0x10/0x10 [cifs] ? _raw_spin_unlock+0x23/0x40 ? __pfx_cifs_readv_from_socket+0x10/0x10 [cifs] decrypt_raw_data+0x229/0x380 [cifs] ? __pfx_decrypt_raw_data+0x10/0x10 [cifs] ? __pfx_cifs_read_iter_from_socket+0x10/0x10 [cifs] smb3_receive_transform+0x837/0xc80 [cifs] ? __pfx_smb3_receive_transform+0x10/0x10 [cifs] ? __pfx___might_resched+0x10/0x10 ? __pfx_smb3_is_transform_hdr+0x10/0x10 [cifs] cifs_demultiplex_thread+0x692/0x1570 [cifs] ? __pfx_cifs_demultiplex_thread+0x10/0x10 [cifs] ? rcu_is_watching+0x20/0x50 ? rcu_lockdep_current_cpu_online+0x62/0xb0 ? find_held_lock+0x32/0x90 ? kvm_sched_clock_read+0x11/0x20 ? local_clock_noinstr+0xd/0xd0 ? trace_irq_enable.constprop.0+0xa8/0xe0 ? __pfx_cifs_demultiplex_thread+0x10/0x10 [cifs] kthread+0x1fe/0x380 ? kthread+0x10f/0x380 ? __pfx_kthread+0x10/0x10 ? local_clock_noinstr+0xd/0xd0 ? ret_from_fork+0x1b/0x60 ? local_clock+0x15/0x30 ? lock_release+0x29b/0x390 ? rcu_is_watching+0x20/0x50 ? __pfx_kthread+0x10/0x10 ret_from_fork+0x31/0x60 ? __pfx_kthread+0x10/0x10 ret_from_fork_asm+0x1a/0x30

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38527	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: smb: client: fix use-after-free in cifs_oplock_break A race condition can occur in cifs_oplock_break() leading to a use-after-free of the cinode structure when unmounting: cifs_oplock_break() _cifsFileInfo_put(cfile) cifsFileInfo_put_final() cifs_sb_deactive() [last ref, start releasing sb] kill_sb() kill_anon_super() generic_shutdown_super() evict_inodes() dispose_list() evict() destroy_inode() call_rcu(&inode->i_rcu, i_callback) spin_lock(&cinode->open_file_lock) <- OK [later] i_callback() cifs_free_inode() kmem_cache_free(cinode) spin_unlock(&cinode->open_file_lock) <- UAF cifs_done_oplock_break(cinode) <- UAF The issue occurs when umount has already released its reference to the superblock. When _cifsFileInfo_put() calls cifs_sb_deactive(), this releases the last reference, triggering the immediate cleanup of all inodes under RCU. However, cifs_oplock_break() continues to access the cinode after this point, resulting in use-after-free. Fix this by holding an extra reference to the superblock during the entire oplock break operation. This ensures that the superblock and its inodes remain valid until the oplock break completes.
CVE-2025-38488	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: smb: client: fix use-after-free in crypt_message when using async crypto The CVE-2024-50047 fix removed asynchronous crypto handling from crypt_message(), assuming all crypto operations are synchronous. However, when hardware crypto accelerators are used, this can cause use-after-free crashes: crypt_message() // Allocate the creq buffer containing the req creq = smb2_get_aead_req(..., &req); // Async encryption returns -EINPROGRESS immediately rc = enc ? crypto_aead_encrypt(req) : crypto_aead_decrypt(req); // Free creq while async operation is still in progress kvfree_sensitive(creq, ...); Hardware crypto modules often implement async AEAD operations for performance. When crypto_aead_encrypt/decrypt() returns -EINPROGRESS, the operation completes asynchronously. Without crypto_wait_req(), the function immediately frees the request buffer, leading to crashes when the driver later accesses the freed memory. This results in a use-after-free condition when the hardware crypto driver later accesses the freed request structure, leading to kernel crashes with NULL pointer dereferences. The issue occurs because crypto_alloc_aead() with mask=0 doesn't guarantee synchronous operation. Even without CRYPTO_ALG_ASYNC in the mask, async implementations can be selected. Fix by restoring the async crypto handling: - DECLARE_CRYPTO_WAIT(wait) for completion tracking - aead_request_set_callback() for async completion notification - crypto_wait_req() to wait for operation completion This ensures the request buffer isn't freed until the crypto operation completes, whether synchronous or asynchronous, while preserving the CVE-2024-50047 fix.
CVE-2025-38464	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: tipc: Fix use-after-free in tipc_conn_close(). syzbot reported a null-ptr-deref in tipc_conn_close() during netns dismantle. [0] tipc_topsrv_stop() iterates tipc_net(net)->topsrv->conn_idr and calls tipc_conn_close() for each tipc_conn. The problem is that tipc_conn_close() is called after releasing the IDR lock. At the same time, there might be tipc_conn_recv_work() running and it could call tipc_conn_close() for the same tipc_conn and release its last ->kref. Once we release the IDR lock in tipc_topsrv_stop(), there is no guarantee that the tipc_conn is alive. Let's

CVE	Severity	CVSS	CVSS Vector	Component	Description
					hold the ref before releasing the lock and put the ref after tipc_conn_close() in tipc_topsrv_stop(). [0]: BUG: KASAN: use-after-free in tipc_conn_close+0x122/0x140 net/tipc/topsrv.c:165 Read of size 8 at addr ffff888099305a08 by task kworker/u4:3/435 CPU: 0 PID: 435 Comm: kworker/u4:3 Not tainted 4.19.204-syzkaller #0 Hardware name: Google Google Compute Engine/Google Compute Engine, BIOS Google 01/01/2011 Workqueue: netns cleanup_net Call Trace: __dump_stack lib/dump_stack.c:77 [inline] dump_stack+0x1fc/0x2ef lib/dump_stack.c:118 print_address_description.cold+0x54/0x219 mm/kasan/report.c:256 kasan_report_error.cold+0x8a/0x1b9 mm/kasan/report.c:354 kasan_report mm/kasan/report.c:412 [inline] __asan_report_load8_noabort+0x88/0x90 mm/kasan/report.c:433 tipc_conn_close+0x122/0x140 net/tipc/topsrv.c:165 tipc_topsrv_stop net/tipc/topsrv.c:701 [inline] tipc_topsrv_exit_net+0x27b/0x5c0 net/tipc/topsrv.c:722 ops_exit_list+0xa5/0x150 net/core/net_namespace.c:153 cleanup_net+0x3b4/0x8b0 net/core/net_namespace.c:553 process_one_work+0x864/0x1570 kernel/workqueue.c:2153 worker_thread+0x64c/0x1130 kernel/workqueue.c:2296 kthread+0x33f/0x460 kernel/kthread.c:259 ret_from_fork+0x24/0x30 arch/x86/entry/entry_64.S:415 Allocated by task 23: kmem_cache_alloc_trace+0x12f/0x380 mm/slab.c:3625 kmalloc include/linux/slab.h:515 [inline] kzalloc include/linux/slab.h:709 [inline] tipc_conn_alloc+0x43/0x4f0 net/tipc/topsrv.c:192 tipc_topsrv_accept+0x1b5/0x280 net/tipc/topsrv.c:470 process_one_work+0x864/0x1570 kernel/workqueue.c:2153 worker_thread+0x64c/0x1130 kernel/workqueue.c:2296 kthread+0x33f/0x460 kernel/kthread.c:259 ret_from_fork+0x24/0x30 arch/x86/entry/entry_64.S:415 Freed by task 23: __cache_free mm/slab.c:3503 [inline] kfree+0xcc/0x210 mm/slab.c:3822 tipc_conn_kref_release net/tipc/topsrv.c:150 [inline] kref_put include/linux/kref.h:70 [inline] conn_put+0x2cd/0x3a0 net/tipc/topsrv.c:155 process_one_work+0x864/0x1570 kernel/workqueue.c:2153 worker_thread+0x64c/0x1130 kernel/workqueue.c:2296 kthread+0x33f/0x460 kernel/kthread.c:259 ret_from_fork+0x24/0x30 arch/x86/entry/entry_64.S:415 The buggy address belongs to the object at ffff888099305a00 which belongs to the cache kmalloc-512 of size 512 The buggy address is located 8 bytes inside of 512-byte region [ffff888099305a00, ffff888099305c00) The buggy address belongs to the page: page:ffffea000264c140 count:1 mapcount:0 mapping:ffff88813bff0940 index:0x0 flags: 0xfff00000000100(slab) raw: 00fff00000000100 ffffea00028b6b88 ffffea0002cd2b08 ffff88813bff0940 raw: 0000000000000000 ffff888099305000 0000000100000006 0000000000000000 page dumped because: kasan: bad access detected Memory state around the buggy address: ffff888099305900: fb fb fb fb fb fb fb fb fb fb fb fb fb fb fb fb ffff888099305980: fc fc fc fc fc fc fc fc fc fc fc fc fc fc fc fc >ffff888099305a00: fb fb fb fb fb fb fb fb fb fb fb fb fb fb fb fb ^ ffff888099305a80: fb fb fb fb fb fb fb fb fb fb fb fb fb fb fb fb ffff888099305b00: fb fb fb fb fb fb fb fb fb fb fb fb fb fb fb fb

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-40149	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: tls: Use __sk_dst_get() and dst_dev_rcu() in get_netdev_for_sock(). get_netdev_for_sock() is called during setsockopt(), so not under RCU. Using sk_dst_get(sk)->dev could trigger UAF. Let's use __sk_dst_get() and dst_dev_rcu(). Note that the only ->ndo_sk_get_lower_dev() user is bond_sk_get_lower_dev(), which uses RCU.
CVE-2025-38471	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: tls: always refresh the queue when reading sock After recent changes in net-next TCP compacts skbs much more aggressively. This unearthed a bug in TLS where we may try to operate on an old skb when checking if all skbs in the queue have matching decrypt state and geometry. BUG: KASAN: slab-use-after-free in tls_strp_check_rcv+0x898/0x9a0 [tls] (net/tls/tls_strp.c:436 net/tls/tls_strp.c:530 net/tls/tls_strp.c:544) Read of size 4 at addr ffff888013085750 by task tls/13529 CPU: 2 UID: 0 PID: 13529 Comm: tls Not tainted 6.16.0-rc5-virtme Call Trace: kasan_report+0xca/0x100 tls_strp_check_rcv+0x898/0x9a0 [tls] tls_rx_rec_wait+0x2c9/0x8d0 [tls] tls_sw_recvmsg+0x40f/0x1aa0 [tls] inet_recvmsg+0x1c3/0x1f0 Always reload the queue, fast path is to have the record in the queue when we wake, anyway (IOW the path going down "if !strp->stm.full_len").
CVE-2025-39951	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: um: virtio_uml: Fix use-after-free after put_device in probe When register_virtio_device() fails in virtio_uml_probe(), the code sets vu_dev->registered = 1 even though the device was not successfully registered. This can lead to use-after-free or other issues.
CVE-2025-38555	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: usb: gadget : fix use-after-free in composite_dev_cleanup() 1. In func configfs_composite_bind() -> composite_os_desc_req_prepare(): if kmalloc fails, the pointer cdev->os_desc_req will be freed but not set to NULL. Then it will return a failure to the upper-level function. 2. in func configfs_composite_bind() -> composite_dev_cleanup(): it will checks whether cdev->os_desc_req is NULL. If it is not NULL, it will attempt to use it.This will lead to a use-after-free issue. BUG: KASAN: use-after-free in composite_dev_cleanup+0xf4/0x2c0 Read of size 8 at addr 0000004827837a00 by task init/1 CPU: 10 PID: 1 Comm: init Tainted: G O 5.10.97-oh #1 kasan_report+0x188/0x1cc __asan_load8+0xb4/0xbc composite_dev_cleanup+0xf4/0x2c0 configfs_composite_bind+0x210/0x7ac udc_bind_to_driver+0xb4/0x1ec usb_gadget_probe_driver+0xec/0x21c gadget_dev_desc_UDC_store+0x264/0x27c

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-37882	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: usb: xhci: Fix isochronous Ring Underrun/Overrun event handling The TRB pointer of these events points at enqueue at the time of error occurrence on xHCI 1.1+ HCs or it's NULL on older ones. By the time we are handling the event, a new TD may be queued at this ring position. I can trigger this race by rising interrupt moderation to increase IRQ handling delay. Similar delay may occur naturally due to system load. If this ever happens after a Missed Service Error, missed TDs will be skipped and the new TD processed as if it matched the event. It could be given back prematurely, risking data loss or buffer UAF by the xHC. Don't complete TDs on xrun events and don't warn if queued TDs don't match the event's TRB pointer, which can be NULL or a link/no-op TRB. Don't warn if there are no queued TDs at all. Now that it's safe, also handle xrun events if the skip flag is clear. This ensures completion of any TD stuck in 'error mid TD' state right before the xrun event, which could happen if a driver submits a finite number of URBs to a buggy HC and then an error occurs on the last TD.
CVE-2025-38618	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: vsock: Do not allow binding to VMADDR_PORT_ANY It is possible for a vsock to autobind to VMADDR_PORT_ANY. This can cause a use-after-free when a connection is made to the bound socket. The socket returned by accept() also has port VMADDR_PORT_ANY but is not on the list of unbound sockets. Binding it will result in an extra refcount decrement similar to the one fixed in fcdd2242c023 (vsock: Keep the binding until socket destruction). Modify the check in __vsock_bind_connectible() to also prevent binding to VMADDR_PORT_ANY.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-39863	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: wifi: brcmfmac: fix use-after-free when rescheduling brcmf_btcoex_info work The brcmf_btcoex_detach() only shuts down the btcoex timer, if the flag timer_on is false. However, the brcmf_btcoex_timerfunc(), which runs as timer handler, sets timer_on to false. This creates critical race conditions: 1.If brcmf_btcoex_detach() is called while brcmf_btcoex_timerfunc() is executing, it may observe timer_on as false and skip the call to timer_shutdown_sync(). 2.The brcmf_btcoex_timerfunc() may then reschedule the brcmf_btcoex_info worker after the cancel_work_sync() has been executed, resulting in use-after-free bugs. The use-after-free bugs occur in two distinct scenarios, depending on the timing of when the brcmf_btcoex_info struct is freed relative to the execution of its worker thread. Scenario 1: Freed before the worker is scheduled The brcmf_btcoex_info is deallocated before the worker is scheduled. A race condition can occur when schedule_work(&bt_local->work) is called after the target memory has been freed. The sequence of events is detailed below: CPU0 | CPU1 brcmf_btcoex_detach | brcmf_btcoex_timerfunc | bt_local->timer_on = false; if (cfg->btcoex->timer_on) | ... | cancel_work_sync(); | ... | kfree(cfg->btcoex); // FREE | | schedule_work(&bt_local->work); // USE Scenario 2: Freed after the worker is scheduled The brcmf_btcoex_info is freed after the worker has been scheduled but before or during its execution. In this case, statements within the brcmf_btcoex_handler() — such as the container_of macro and subsequent dereferences of the brcmf_btcoex_info object will cause a use-after-free access. The following timeline illustrates this scenario: CPU0 | CPU1 brcmf_btcoex_detach | brcmf_btcoex_timerfunc | bt_local->timer_on = false; if (cfg->btcoex->timer_on) | ... | cancel_work_sync(); | ... | schedule_work(); // Reschedule | kfree(cfg->btcoex); // FREE | brcmf_btcoex_handler() // Worker /* | btci = container_of(....); // USE The kfree() above could | ... also occur at any point | btci-> // USE during the worker's execution| */ | To resolve the race conditions, drop the conditional check and call timer_shutdown_sync() directly. It can deactivate the timer reliably, regardless of its current state. Once stopped, the timer_on state is then set to false.
CVE-2025-39864	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: wifi: cfg80211: fix use-after-free in cmp_bss() Following bss_free() quirk introduced in commit 776b3580178f ("cfg80211: track hidden SSID networks properly"), adjust cfg80211_update_known_bss() to free the last beacon frame elements only if they're not shared via the corresponding 'hidden_beacon_bss' pointer.
CVE-2025-21786	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: workqueue: Put the pwq after detaching the rescuer from the pool The commit 68f83057b913("workqueue: Reap workers via kthread_stop() and remove detach_completion") adds code to reap the normal workers but mistakenly does not handle the rescuer and also removes the code waiting for the rescuer in put_unbound_pool(), which caused a use-after-free bug reported by Cheung Wall. To avoid the use-after-free bug, the pool's reference must be held until the detachment is complete. Therefore, move the code that puts the pwq after detaching the rescuer from the pool.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38595	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: xen: fix UAF in dmabuf_exp_from_pages() [dma_buf_fd() fixes; no preferences regarding the tree it goes through - up to xen folks] As soon as we'd inserted a file reference into descriptor table, another thread could close it. That's fine for the case when all we are doing is returning that descriptor to userland (it's a race, but it's a userland race and there's nothing the kernel can do about it). However, if we follow fd_install() with any kind of access to objects that would be destroyed on close (be it the struct file itself or anything destroyed by its ->release()), we have a UAF. dma_buf_fd() is a combination of reserving a descriptor and fd_install(). gntdev dmabuf_exp_from_pages() calls it and then proceeds to access the objects destroyed on close - starting with gntdev_dmabuf itself. Fix that by doing reserving descriptor before anything else and do fd_install() only when everything had been set up.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38500	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: xfrm: interface: fix use-after-free after changing collect_md xfrm interface collect_md property on xfrm interfaces can only be set on device creation, thus xfrmi_changelink() should fail when called on such interfaces. The check to enforce this was done only in the case where the xi was returned from xfrmi_locate() which doesn't look for the collect_md interface, and thus the validation was never reached. Calling changelink would thus erroneously place the special interface xi in the xfrmi_net->xfrmi hash, but since it also exists in the xfrmi_net->collect_md_xfrmi pointer it would lead to a double free when the net namespace was taken down [1]. Change the check to use the xi from netdev_priv which is available earlier in the function to prevent changes in xfrm collect_md interfaces. [1] resulting oops: [8.516540] kernel BUG at net/core/dev.c:12029! [8.516552] Oops: invalid opcode: 0000 [#1] SMP NOPTI [8.516559] CPU: 0 UID: 0 PID: 12 Comm: kworker/u80:0 Not tainted 6.15.0-virtme #5 PREEMPT(voluntary) [8.516565] Hardware name: QEMU Ubuntu 24.04 PC (i440FX + PIIX, 1996), BIOS 1.16.3-debian-1.16.3-2 04/01/2014 [8.516569] Workqueue: netns cleanup_net [8.516579] RIP: 0010:unregister_netdevice_many_notify+0x101/0xab0 [8.516590] Code: 90 0f 0b 90 48 8b b0 78 01 00 00 48 8b 90 80 01 00 00 48 89 56 08 48 89 32 4c 89 80 78 01 00 00 48 89 b8 80 01 00 00 eb ac 90 <0f> 0b 48 8b 45 00 4c 8d a0 88 fe ff ff 48 39 c5 74 5c 41 80 bc 24 [8.516593] RSP: 0018:ffffa93b8006bd30 EFLAGS: 00010206 [8.516598] RAX: ffff98fe4226e000 RBX: ffffa93b8006bd58 RCX: ffffa93b8006bc60 [8.516601] RDX: 0000000000000004 RSI: 0000000000000000 RDI: dead000000000122 [8.516603] RBP: ffffa93b8006bdd8 R08: dead000000000100 R09: ffff98fe4133c100 [8.516605] R10: 0000000000000000 R11: 00000000000003d2 R12: ffffa93b8006be00 [8.516608] R13: ffffffff96c1a510 R14: ffffffff96c1a510 R15: ffffa93b8006be00 [8.516615] FS: 0000000000000000(0000) GS:ffff98fee73b7000(0000) knlGS:0000000000000000 [8.516619] CS: 0010 DS: 0000 ES: 0000 CR0: 0000000080050033 [8.516622] CR2: 00007fcd2abd0700 CR3: 000000003aa40000 CR4: 0000000000752ef0 [8.516625] PKRU: 55555554 [8.516627] Call Trace: [8.516632] [8.516635] ? rtnl_is_locked+0x15/0x20 [8.516641] ? unregister_netdevice_queue+0x29/0xf0 [8.516650] ops_undo_list+0x1f2/0x220 [8.516659] cleanup_net+0x1ad/0x2e0 [8.516664] process_one_work+0x160/0x380 [8.516673] worker_thread+0x2aa/0x3c0 [8.516679] ? __pfx_worker_thread+0x10/0x10 [8.516686] kthread+0xfb/0x200 [8.516690] ? __pfx_kthread+0x10/0x10 [8.516693] ? __pfx_kthread+0x10/0x10 [8.516697] ret_from_fork+0x82/0xf0 [8.516705] ? __pfx_kthread+0x10/0x10 [8.516709] ret_from_fork_asm+0x1a/0x30 [8.516718]

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38459	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: atm: clip: Fix infinite recursive call of clip_push(). syzbot reported the splat below. [0] This happens if we call ioctl(ATMARP_MKIP) more than once. During the first call, clip_mkip() sets clip_push() to vcc->push(), and the second call copies it to clip_vcc->old_push(). Later, when the socket is close()d, vcc_destroy_socket() passes NULL skb to clip_push(), which calls clip_vcc->old_push(), triggering the infinite recursion. Let's prevent the second ioctl(ATMARP_MKIP) by checking vcc->user_back, which is allocated by the first call as clip_vcc. Note also that we use lock_sock() to prevent racy calls. [0]: BUG: TASK stack guard page was hit at ffffc9000d66fff8 (stack is ffffc9000d670000..ffffc9000d678000) Oops: stack guard page: 0000 [#1] SMP KASAN NOPTI CPU: 0 UID: 0 PID: 5322 Comm: syz.0.0 Not tainted 6.16.0-rc4-syzkaller #0 PREEMPT(full) Hardware name: QEMU Standard PC (Q35 + ICH9, 2009), BIOS 1.16.3-debian-1.16.3-2~bpo12+1 04/01/2014 RIP: 0010:clip_push+0x5/0x720 net/atm/clip.c:191 Code: e0 8f aa 8c e8 1c ad 5b fa eb ae 66 2e 0f 1f 84 00 00 00 00 00 90 90 90 90 90 90 90 90 90 90 90 90 90 90 f3 0f 1e fa 55 <41> 57 41 56 41 55 41 54 53 48 83 ec 20 48 89 f3 49 89 fd 48 bd 00 RSP: 0018:ffffc9000d670000 EFLAGS: 00010246 RAX: 1ffff1100235a4a5 RBX: ffff888011ad2508 RCX: ffff8880003c0000 RDX: 0000000000000000 RSI: 0000000000000000 RDI: ffff888037f01000 RBP: dffffc0000000000 R08: ffffffff8fa104f7 R09: 1ffffffff1f4209e R10: dffffc0000000000 R11: fffffffff8a99b300 R12: fffffffff8a99b300 R13: ffff888037f01000 R14: ffff888011ad2500 R15: ffff888037f01578 FS: 000055557ab6d500(0000) GS:ffff88808d250000(0000) knlGS:0000000000000000 CS: 0010 DS: 0000 ES: 0000 CR0: 0000000080050033 CR2: ffffc9000d66fff8 CR3: 0000000043172000 CR4: 0000000000352ef0 Call Trace: clip_push+0x6dc/0x720 net/atm/clip.c:200 clip_push+0x6dc/0x720 net/atm/clip.c:200 clip_push+0x6dc/0x720 net/atm/clip.c:200 ... clip_push+0x6dc/0x720 net/atm/clip.c:200 clip_push+0x6dc/0x720 net/atm/clip.c:200 clip_push+0x6dc/0x720 net/atm/clip.c:200 vcc_destroy_socket net/atm/common.c:183 [inline] vcc_release+0x157/0x460 net/atm/common.c:205 __sock_release net/socket.c:647 [inline] sock_close+0xc0/0x240 net/socket.c:1391 __fput+0x449/0xa70 fs/file_table.c:465 task_work_run+0x1d1/0x260 kernel/task_work.c:227 resume_user_mode_work include/linux/resume_user_mode.h:50 [inline] exit_to_user_mode_loop+0xec/0x110 kernel/entry/common.c:114 exit_to_user_mode_prepare include/linux/entry-common.h:330 [inline] syscall_exit_to_user_mode_work include/linux/entry-common.h:414 [inline] syscall_exit_to_user_mode include/linux/entry-common.h:449 [inline] do_syscall_64+0x2bd/0x3b0 arch/x86/entry/syscall_64.c:100 entry_SYSCALL_64_after_hwframe+0x77/0x7f RIP: 0033:0x7ff31c98e929 Code: ff ff c3 66 2e 0f 1f 84 00 00 00 00 00 0f 1f 40 00 48 89 f8 48 89 f7 48 89 d6 48 89 ca 4d 89 c2 4d 89 c8 4c 8b 4c 24 08 0f 05 <48> 3d 01 f0 ff ff 73 01 c3 48 c7 c1 a8 ff ff ff f7 d8 64 89 01 48 RSP: 002b:00007fffb5aa1f78 EFLAGS: 00000246 ORIG_RAX: 00000000000001b4 RAX: 0000000000000000 RBX: 0000000000012747 RCX: 00007ff31c98e929 RDX: 0000000000000000 RSI: 000000000000001e RDI: 0000000000000003 RBP: 00007ff31cbb7ba0 R08: 0000000000000001 R09: 0000000db5aa226f R10: 00007ff31c7ff030 R11: 0000000000000246 R12: 00007ff31cbb608c R13: 00007ff31cbb6080 R14: ffffffffffffffff R15: 00007fffb5aa2090 Modules linked

CVE	Severity	CVSS	CVSS Vector	Component	in: Description
CVE-2025-39880	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: libceph: fix invalid accesses to ceph_connection_v1_info There is a place where generic code in messenger.c is reading and another place where it is writing to con->v1 union member without checking that the union member is active (i.e. msgr1 is in use). On 64-bit systems, con->v1.auth_retry overlaps with con->v2.out_iter, so such a read is almost guaranteed to return a bogus value instead of 0 when msgr2 is in use. This ends up being fairly benign because the side effect is just the invalidation of the authorizer and successive fetching of new tickets. con->v1.connect_seq overlaps with con->v2.conn_bufs and the fact that it's being written to can cause more serious consequences, but luckily it's not something that happens often.
CVE-2026-23078	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ALSA: scarlett2: Fix buffer overflow in config retrieval The scarlett2_usb_get_config() function has a logic error in the endianness conversion code that can cause buffer overflows when count > 1. The code checks `if (size == 2)` where `size` is the total buffer size in bytes, then loops `count` times treating each element as u16 (2 bytes). This causes the loop to access `count * 2` bytes when the buffer only has `size` bytes allocated. Fix by checking the element size (config_item->size) instead of the total buffer size. This ensures the endianness conversion matches the actual element type.
CVE-2025-38729	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ALSA: usb-audio: Validate UAC3 power domain descriptors, too UAC3 power domain descriptors need to be verified with its variable bLength for avoiding the unexpected OOB accesses by malicious firmware, too.
CVE-2025-39783	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: PCI: endpoint: Fix configfs group list head handling Doing a list_del() on the epf_group field of struct pci_epf_driver in pci_epf_remove_cfs() is not correct as this field is a list head, not a list entry. This list_del() call triggers a KASAN warning when an endpoint function driver which has a configfs attribute group is torn down: ================================== ========== BUG: KASAN: slab-use-after-free in pci_epf_remove_cfs+0x17c/0x198 Write of size 8 at addr ffff00010f4a0d80 by task rmmod/319 CPU: 3 UID: 0 PID: 319 Comm: rmmod Not tainted 6.16.0-rc2 #1 NONE Hardware name: Radxa ROCK 5B (DT) Call trace: show_stack+0x2c/0x84 (C) dump_stack_lvl+0x70/0x98 print_report+0x17c/0x538 kasan_report+0xb8/0x190 __asan_report_store8_noabort+0x20/0x2c pci_epf_remove_cfs+0x17c/0x198 pci_epf_unregister_driver+0x18/0x30 nvmet_pci_epf_cleanup_module+0x24/0x30 [nvmet_pci_epf] __arm64_sys_delete_module+0x264/0x424 invoke_syscall+0x70/0x260 el0_svc_common.constprop.0+0xac/0x230 do_el0_svc+0x40/0x58 el0_svc+0x48/0xdc el0t_64_sync_handler+0x10c/0x138 el0t_64_sync+0x198/0x19c ... Remove this incorrect list_del() call from pci_epf_remove_cfs().

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-39810	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: bnxt_en: Fix memory corruption when FW resources change during ifdown bnxt_set_dflt_rings() assumes that it is always called before any TC has been created. So it doesn't take bp->num_tc into account and assumes that it is always 0 or 1. In the FW resource or capability change scenario, the FW will return flags in bnxt_hwrm_if_change() that will cause the driver to reinitialize and call bnxt_cancel_reservations(). This will lead to bnxt_init_dflt_ring_mode() calling bnxt_set_dflt_rings() and bp->num_tc may be greater than 1. This will cause bp->tx_ring[] to be sized too small and cause memory corruption in bnxt_alloc_cp_rings(). Fix it by properly scaling the TX rings by bp->num_tc in the code paths mentioned above. Add 2 helper functions to determine bp->tx_nr_rings and bp->tx_nr_rings_per_tc.
CVE-2025-38538	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: dmaengine: nbpfaxi: Fix memory corruption in probe() The nbpf->chan[] array is allocated earlier in the nbpf_probe() function and it has "num_channels" elements. These three loops iterate one element farther than they should and corrupt memory. The changes to the second loop are more involved. In this case, we're copying data from the irqbuf[] array into the nbpf->chan[] array. If the data in irqbuf[i] is the error IRQ then we skip it, so the iterators are not in sync. I added a check to ensure that we don't go beyond the end of the irqbuf[] array. I'm pretty sure this can't happen, but it seemed harmless to add a check. On the other hand, after the loop has ended there is a check to ensure that the "chan" iterator is where we expect it to be. In the original code we went one element beyond the end of the array so the iterator wasn't in the correct place and it would always return -EINVAL. However, now it will always be in the correct place. I deleted the check since we know the result.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2026-23233	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: f2fs: fix to avoid mapping wrong physical block for swapfile Xiaolong Guo reported a f2fs bug in bugzilla [1] [1] https://bugzilla.kernel.org/show_bug.cgi?id=220951 Quoted: "When using stress-ng's swap stress test on F2FS filesystem with kernel 6.6+, the system experiences data corruption leading to either: 1 dm-verity corruption errors and device reboot 2 F2FS node corruption errors and boot hangs The issue occurs specifically when: 1 Using F2FS filesystem (ext4 is unaffected) 2 Swapfile size is less than F2FS section size (2MB) 3 Swapfile has fragmented physical layout (multiple non-contiguous extents) 4 Kernel version is 6.6+ (6.1 is unaffected) The root cause is in check_swap_activate() function in fs/f2fs/data.c. When the first extent of a small swapfile (< 2MB) is not aligned to section boundaries, the function incorrectly treats it as the last extent, failing to map subsequent extents. This results in incorrect swap_extent creation where only the first extent is mapped, causing subsequent swap writes to overwrite wrong physical locations (other files' data). Steps to Reproduce 1 Setup a device with F2FS-formatted userdata partition 2 Compile stress-ng from https://github.com/ColinIanKing/stress-ng 3 Run swap stress test: (Android devices) adb shell "cd /data/stressng; ./stress-ng-64 --metrics-brief --timeout 60 --swap 0" Log: 1 Ftrace shows in kernel 6.6, only first extent is mapped during second f2fs_map_blocks call in check_swap_activate(): stress-ng-swap-8990: f2fs_map_blocks: ino=11002, file offset=0, start blkaddr=0x43143, len=0x1 (Only 4KB mapped, not the full swapfile) 2 in kernel 6.1, both extents are correctly mapped: stress-ng-swap-5966: f2fs_map_blocks: ino=28011, file offset=0, start blkaddr=0x13cd4, len=0x1 stress-ng-swap-5966: f2fs_map_blocks: ino=28011, file offset=1, start blkaddr=0x60c84b, len=0xff The problematic code is in check_swap_activate(): if ((pblock - SM_I(sbi)->main_blkaddr) % blks_per_sec || nr_pblocks % blks_per_sec || !f2fs_valid_pinned_area(sbi, pblock)) { bool last_extent = false; not_aligned++; nr_pblocks = roundup(nr_pblocks, blks_per_sec); if (cur_lblock + nr_pblocks > sis->max) nr_pblocks -= blks_per_sec; /* this extent is last one */ if (!nr_pblocks) { nr_pblocks = last_lblock - cur_lblock; last_extent = true; } ret = f2fs_migrate_blocks(inode, cur_lblock, nr_pblocks); if (ret) { if (ret == -ENOENT) ret = -EINVAL; goto out; } if (!last_extent) goto retry; } When the first extent is unaligned and roundup(nr_pblocks, blks_per_sec) exceeds sis->max, we subtract blks_per_sec resulting in nr_pblocks = 0. The code then incorrectly assumes this is the last extent, sets nr_pblocks = last_lblock - cur_lblock (entire swapfile), and performs migration. After migration, it doesn't retry mapping, so subsequent extents are never processed. " In order to fix this issue, we need to lookup block mapping info after we migrate all blocks in the tail of swapfile.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38685	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: fbdev: Fix vmalloc out-of-bounds write in fast_imageblit This issue triggers when a userspace program does an ioctl FBIOPUT_CON2FBMAP by passing console number and frame buffer number. Ideally this maps console to frame buffer and updates the screen if console is visible. As part of mapping it has to do resize of console according to frame buffer info. if this resize fails and returns from vc_do_resize() and continues further. At this point console and new frame buffer are mapped and sets display vars. Despite failure still it continue to proceed updating the screen at later stages where vc_data is related to previous frame buffer and frame buffer info and display vars are mapped to new frame buffer and eventully leading to out-of-bounds write in fast_imageblit(). This bheviour is excepted only when fg_console is equal to requested console which is a visible console and updates screen with invalid struct references in fbcon_putcs().
CVE-2025-38702	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: fbdev: fix potential buffer overflow in do_register_framebuffer() The current implementation may lead to buffer overflow when: 1. Unregistration creates NULL gaps in registered_fb[] 2. All array slots become occupied despite num_registered_fb < FB_MAX 3. The registration loop exceeds array bounds Add boundary check to prevent registered_fb[FB_MAX] access.
CVE-2025-38730	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: io_uring/net: commit partial buffers on retry Ring provided buffers are potentially only valid within the single execution context in which they were acquired. io_uring deals with this and invalidates them on retry. But on the networking side, if MSG_WAITALL is set, or if the socket is of the streaming type and too little was processed, then it will hang on to the buffer rather than recycle or commit it. This is problematic for two reasons: 1) If someone unregisters the provided buffer ring before a later retry, then the req->buf_list will no longer be valid. 2) If multiple sockers are using the same buffer group, then multiple receives can consume the same memory. This can cause data corruption in the application, as either receive could land in the same userspace buffer. Fix this by disallowing partial retries from pinning a provided buffer across multiple executions, if ring provided buffers are used.
CVE-2025-38676	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: iommu/amd: Avoid stack buffer overflow from kernel cmdline While the kernel command line is considered trusted in most environments, avoid writing 1 byte past the end of "acpiid" if the "str" argument is maximum length.
CVE-2025-38456	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ipmi:msghandler: Fix potential memory corruption in ipmi_create_user() The "intf" list iterator is an invalid pointer if the correct "intf->intf_num" is not found. Calling atomic_dec(&intf->nr_users) on and invalid pointer will lead to memory corruption. We don't really need to call atomic_dec() if we haven't called atomic_add_return() so update the if (intf->in_shutdown) path as well.
CVE-2025-38401	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: mtk-sd: Prevent memory corruption from DMA map failure If msdc_prepare_data() fails to map the DMA region, the request is not prepared for data receiving, but msdc_start_data() proceeds the DMA with previous setting. Since this will lead a memory corruption, we have to stop the request operation soon after the msdc_prepare_data() fails to prepare it.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38568	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net/sched: mqprio: fix stack out-of-bounds write in tc entry parsing TCA_MQPRIO_TC_ENTRY_INDEX is validated using NLA_POLICY_MAX(NLA_U32, TC_QOPT_MAX_QUEUE), which allows the value TC_QOPT_MAX_QUEUE (16). This leads to a 4-byte out-of-bounds stack write in the fp[] array, which only has room for 16 elements (0–15). Fix this by changing the policy to allow only up to TC_QOPT_MAX_QUEUE - 1.
CVE-2025-38533	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net: libwx: fix the using of Rx buffer DMA The wx_rx_buffer structure contained two DMA address fields: 'dma' and 'page_dma'. However, only 'page_dma' was actually initialized and used to program the Rx descriptor. But 'dma' was uninitialized and used in some paths. This could lead to undefined behavior, including DMA errors or use-after-free, if the uninitialized 'dma' was used. Although such error has not yet occurred, it is worth fixing in the code.
CVE-2025-21927	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: nvme-tcp: fix potential memory corruption in nvme_tcp_recv_pdu() nvme_tcp_recv_pdu() doesn't check the validity of the header length. When header digests are enabled, a target might send a packet with an invalid header length (e.g. 255), causing nvme_tcp_verify_hdgst() to access memory outside the allocated area and cause memory corruptions by overwriting it with the calculated digest. Fix this by rejecting packets with an unexpected header length.
CVE-2025-71137	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: octeontx2-pf: fix "UBSAN: shift-out-of-bounds error" This patch ensures that the RX ring size (rx_pending) is not set below the permitted length. This avoids UBSAN shift-out-of-bounds errors when users passes small or zero ring sizes via ethtool -G.
CVE-2025-39841	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: scsi: lpfc: Fix buffer free/clear order in deferred receive path Fix a use-after-free window by correcting the buffer release sequence in the deferred receive path. The code freed the RQ buffer first and only then cleared the context pointer under the lock. Concurrent paths (e.g., ABTS and the repost path) also inspect and release the same pointer under the lock, so the old order could lead to double-free/UAF. Note that the repost path already uses the correct pattern: detach the pointer under the lock, then free it after dropping the lock. The deferred path should do the same.
CVE-2025-39788	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: scsi: ufs: exynos: Fix programming of HCI_UTRL_NEXUS_TYPE On Google gs101, the number of UTP transfer request slots (nutrs) is 32, and in this case the driver ends up programming the UTRL_NEXUS_TYPE incorrectly as 0. This is because the left hand side of the shift is 1, which is of type int, i.e. 31 bits wide. Shifting by more than that width results in undefined behaviour. Fix this by switching to the BIT() macro, which applies correct type casting as required. This ensures the correct value is written to UTRL_NEXUS_TYPE (0xffffffff on gs101), and it also fixes a UBSAN shift warning: UBSAN: shift-out-of-bounds in drivers/ufs/host/ufs-exynos.c:1113:21 shift exponent 32 is too large for 32-bit type 'int' For consistency, apply the same change to the nutmrs / UTMRL_NEXUS_TYPE write.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38585	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: staging: media: atomisp: Fix stack buffer overflow in gmin_get_var_int() When gmin_get_config_var() calls efi.get_variable() and the EFI variable is larger than the expected buffer size, two behaviors combine to create a stack buffer overflow: 1. gmin_get_config_var() does not return the proper error code when efi.get_variable() fails. It returns the stale 'ret' value from earlier operations instead of indicating the EFI failure. 2. When efi.get_variable() returns EFI_BUFFER_TOO_SMALL, it updates *out_len to the required buffer size but writes no data to the output buffer. However, due to bug #1, gmin_get_var_int() believes the call succeeded. The caller gmin_get_var_int() then performs: - Allocates val[CFG_VAR_NAME_MAX + 1] (65 bytes) on stack - Calls gmin_get_config_var(dev, is_gmin, var, val, &len) with len=64 - If EFI variable is >64 bytes, efi.get_variable() sets len=required_size - Due to bug #1, thinks call succeeded with len=required_size - Executes val[len] = 0, writing past end of 65-byte stack buffer This creates a stack buffer overflow when EFI variables are larger than 64 bytes. Since EFI variables can be controlled by firmware or system configuration, this could potentially be exploited for code execution. Fix the bug by returning proper error codes from gmin_get_config_var() based on EFI status instead of stale 'ret' value. The gmin_get_var_int() function is called during device initialization for camera sensor configuration on Intel Bay Trail and Cherry Trail platforms using the atomisp camera stack.
CVE-2025-22022	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: usb: xhci: Apply the link chain quirk on NEC isoc endpoints Two clearly different specimens of NEC uPD720200 (one with start/stop bug, one without) were seen to cause IOMMU faults after some Missed Service Errors. Faulting address is immediately after a transfer ring segment and patched dynamic debug messages revealed that the MSE was received when waiting for a TD near the end of that segment: [1.041954] xhci_hcd: Miss service interval error for slot 1 ep 2 expected TD DMA ffa08fe0 [1.042120] xhci_hcd: AMD-Vi: Event logged [IO_PAGE_FAULT domain=0x0005 address=0xffa09000 flags=0x0000] [1.042146] xhci_hcd: AMD-Vi: Event logged [IO_PAGE_FAULT domain=0x0005 address=0xffa09040 flags=0x0000] It gets even funnier if the next page is a ring segment accessible to the HC. Below, it reports MSE in segment at ff1e8000, plows through a zero-filled page at ff1e9000 and starts reporting events for TRBs in page at ff1ea000 every microframe, instead of jumping to seg ff1e6000. [7.041671] xhci_hcd: Miss service interval error for slot 1 ep 2 expected TD DMA ff1e8fe0 [7.041999] xhci_hcd: Miss service interval error for slot 1 ep 2 expected TD DMA ff1e8fe0 [7.042011] xhci_hcd: WARN: buffer overrun event for slot 1 ep 2 on endpoint [7.042028] xhci_hcd: All TDs skipped for slot 1 ep 2. Clear skip flag. [7.042134] xhci_hcd: WARN: buffer overrun event for slot 1 ep 2 on endpoint [7.042138] xhci_hcd: ERROR Transfer event TRB DMA ptr not part of current TD ep_index 2 comp_code 31 [7.042144] xhci_hcd: Looking for event-dma 00000000ff1ea040 trb-start 00000000ff1e6820 trb-end 00000000ff1e6820 [7.042259] xhci_hcd: WARN: buffer overrun event for slot 1 ep 2 on endpoint [7.042262] xhci_hcd: ERROR Transfer event TRB DMA ptr not part of current TD ep_index 2 comp_code 31 [7.042266] xhci_hcd: Looking for event-dma 00000000ff1ea050 trb-start 00000000ff1e6820 trb-end 00000000ff1e6820 At some point completion events change from Isoch Buffer Overrun to Short Packet and the HC finally finds cycle bit mismatch in ff1ec000. [7.098130] xhci_hcd:

CVE	Severity	CVSS	CVSS Vector	Component	Description
					ERROR Transfer event TRB DMA ptr not part of current TD ep_index 2 comp_code 13 [7.098132] xhci_hcd: Looking for event-dma 00000000ff1ecc50 trb-start 00000000ff1e6820 trb-end 00000000ff1e6820 [7.098254] xhci_hcd: ERROR Transfer event TRB DMA ptr not part of current TD ep_index 2 comp_code 13 [7.098256] xhci_hcd: Looking for event-dma 00000000ff1ecc60 trb-start 00000000ff1e6820 trb-end 00000000ff1e6820 [7.098379] xhci_hcd: Overrun event on slot 1 ep 2 It's possible that data from the isochronous device were written to random buffers of pending TDs on other endpoints (either IN or OUT), other devices or even other HCs in the same IOMMU domain. Lastly, an error from a different USB device on another HC. Was it caused by the above? I don't know, but it may have been. The disk was working without any other issues and generated PCIe traffic to starve the NEC of upstream BW and trigger those MSEs. The two HCs shared one x1 slot by means of a commercial "PCIe splitter" board. [7.162604] usb 10-2: reset SuperSpeed USB device number 3 using xhci_hcd [7.178990] sd 9:0:0:0: [sdb] tag#0 UNKNOWN(0x2003) Result: hostbyte=0x07 driverbyte=DRIVER_OK cmd_age=0s [7.179001] sd 9:0:0:0: [sdb] tag#0 CDB: opcode=0x28 28 00 04 02 ae 00 00 02 00 00 [7.179004] I/O error, dev sdb, sector 67284480 op 0x0:(READ) flags 0x80700 phys_seg 5 prio class 0 Fortunately, it appears that this ridiculous bug is avoided by setting the chain bit of Link TRBs on isochronous rings. Other ancient HCs are known which also expect the bit to be set and they ignore Link TRBs if it's not. Reportedly, 0.95 spec guaranteed that the bit is set. The bandwidth-starved NEC HC running a 32KB/uframe UVC endpoint reports tens of MSEs per second and runs into the bug within seconds. Chaining Link TRBs allows the same workload to run for many minutes, many times. No ne ---truncated---
CVE-2025-23133	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: wifi: ath11k: update channel list in reg notifier instead reg worker Currently when ath11k gets a new channel list, it will be processed according to the following steps: 1. update new channel list to cfg80211 and queue reg_work. 2. cfg80211 handles new channel list during reg_work. 3. update cfg80211's handled channel list to firmware by ath11k_reg_update_chan_list(). But ath11k will immediately execute step 3 after reg_work is just queued. Since step 2 is asynchronous, cfg80211 may not have completed handling the new channel list, which may leading to an out-of-bounds write error: BUG: KASAN: slab-out-of-bounds in ath11k_reg_update_chan_list Call Trace: ath11k_reg_update_chan_list+0xbfe/0xfe0 [ath11k] kfree+0x109/0x3a0 ath11k_regd_update+0x1cf/0x350 [ath11k] ath11k_regd_update_work+0x14/0x20 [ath11k] process_one_work+0xe35/0x14c0 Should ensure step 2 is completely done before executing step 3. Thus Wen raised patch[1]. When flag NL80211_REGDOM_SET_BY_DRIVER is set, cfg80211 will notify ath11k after step 2 is done. So enable the flag NL80211_REGDOM_SET_BY_DRIVER then cfg80211 will notify ath11k after step 2 is done. At this time, there will be no KASAN bug during the execution of the step 3. [1] https://patchwork.kernel.org/project/linux-wireless/patch/20230201065313.27203-1-quic_wgong@quicinc.com/ Tested-on: WCN6855 hw2.0 PCI WLAN.HSP.1.1-03125-QCAHSPSWPL_V1_V2_SILICONZ_LITE-3

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-39849	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: wifi: cfg80211: sme: cap SSID length in __cfg80211_connect_result() If the ssid->datalen is more than IEEE80211_MAX_SSID_LEN (32) it would lead to memory corruption so add some bounds checking.
CVE-2025-39862	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: wifi: mt76: mt7915: fix list corruption after hardware restart Since stations are recreated from scratch, all lists that wcids are added to must be cleared before calling ieee80211_restart_hw. Set wcid->sta = 0 for each wcid entry in order to ensure that they are not added again before they are ready.
CVE-2026-23073	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: wifi: rsi: Fix memory corruption due to not set vif driver data size The struct ieee80211_vif contains trailing space for vif driver data, when struct ieee80211_vif is allocated, the total memory size that is allocated is sizeof(struct ieee80211_vif) + size of vif driver data. The size of vif driver data is set by each WiFi driver as needed. The RSI911x driver does not set vif driver data size, no trailing space for vif driver data is therefore allocated past struct ieee80211_vif . The RSI911x driver does however use the vif driver data to store its vif driver data structure "struct vif_priv". An access to vif->drv_priv leads to access out of struct ieee80211_vif bounds and corruption of some memory. In case of the failure observed locally, rsi_mac80211_add_interface() would write struct vif_priv *vif_info = (struct vif_priv *)vif->drv_priv; vif_info->vap_id = vap_idx. This write corrupts struct fq_tin member struct list_head new_flows . The flow = list_first_entry(head, struct fq_flow, flowchain); in fq_tin_reset() then reports non-NULL bogus address, which when accessed causes a crash. The trigger is very simple, boot the machine with init=/bin/sh , mount devtmpfs, sysfs, procfs, and then do "ip link set wlan0 up", "sleep 1", "ip link set wlan0 down" and the crash occurs. Fix this by setting the correct size of vif driver data, which is the size of "struct vif_priv", so that memory is allocated and the driver can store its driver data in it, instead of corrupting memory around it.
CVE-2025-39952	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: wifi: wilc1000: avoid buffer overflow in WID string configuration Fix the following copy overflow warning identified by Smatch checker. drivers/net/wireless/microchip/wilc1000/wlan_cfg .c:184 wilc_wlan_parse_response_frame() error: '__memcpy()' 'cfg->s[i]->str' copy overflow (512 vs 65537) This patch introduces size check before accessing the memory buffer. The checks are base on the WID type of received data from the firmware. For WID string configuration, the size limit is determined by individual element size in 'struct wilc_cfg_str_vals' that is maintained in 'len' field of 'struct wilc_cfg_str'.
CVE-2025-38579	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: f2fs: fix KMSAN uninit-value in extent_info usage KMSAN reported a use of uninitialized value in `__is_extent_mergeable()` and `__is_back_mergeable()` via the read extent tree path. The root cause is that `get_read_extent_info()` only initializes three fields (`fofs`, `blk`, `len`) of `struct extent_info`, leaving the remaining fields uninitialized. This leads to undefined behavior when those fields are accessed later, especially during extent merging. Fix it by zero-initializing the `extent_info` struct before population.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38574	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: pptp: ensure minimal skb length in pptp_xmit() Commit aabc6596ffb3 ("net: ppp: Add bound checking for skb data on ppp_sync_txmung") fixed ppp_sync_txmunge() We need a similar fix in pptp_xmit(), otherwise we might read uninit data as reported by syzbot. BUG: KMSAN: uninit-value in pptp_xmit+0xc34/0x2720 drivers/net/ppp/pptp.c:193 pptp_xmit+0xc34/0x2720 drivers/net/ppp/pptp.c:193 ppp_channel_bridge_input drivers/net/ppp/ppp_generic.c:2290 [inline] ppp_input+0x1d6/0xe60 drivers/net/ppp/ppp_generic.c:2314 pppoe_rcv_core+0x1e8/0x760 drivers/net/ppp/pppoe.c:379 sk_backlog_rcv+0x142/0x420 include/net/sock.h:1148 __release_sock+0x1d3/0x330 net/core/sock.c:3213 release_sock+0x6b/0x270 net/core/sock.c:3767 pppoe_sendmsg+0x15d/0xcb0 drivers/net/ppp/pppoe.c:904 sock_sendmsg_nosec net/socket.c:712 [inline] __sock_sendmsg+0x330/0x3d0 net/socket.c:727 ____sys_sendmsg+0x893/0xd80 net/socket.c:2566 ___sys_sendmsg+0x271/0x3b0 net/socket.c:2620 __sys_sendmmsg+0x2d9/0x7c0 net/socket.c:2709

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38718	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: sctp: linearize cloned gso packets in sctp_rcv A cloned head skb still shares these frag skbs in fraglist with the original head skb. It's not safe to access these frag skbs. syzbot reported two use-of-uninitialized-memory bugs caused by this: BUG: KMSAN: uninit-value in sctp_inq_pop+0x15b7/0x1920 net/sctp/inqueue.c:211 sctp_inq_pop+0x15b7/0x1920 net/sctp/inqueue.c:211 sctp_assoc_bh_rcv+0x1a7/0xc50 net/sctp/associola.c:998 sctp_inq_push+0x2ef/0x380 net/sctp/inqueue.c:88 sctp_backlog_rcv+0x397/0xdb0 net/sctp/input.c:331 sk_backlog_rcv+0x13b/0x420 include/net/sock.h:1122 __release_sock+0x1da/0x330 net/core/sock.c:3106 release_sock+0x6b/0x250 net/core/sock.c:3660 sctp_wait_for_connect+0x487/0x820 net/sctp/socket.c:9360 sctp_sendmsg_to_asoc+0x1ec1/0x1f00 net/sctp/socket.c:1885 sctp_sendmsg+0x32b9/0x4a80 net/sctp/socket.c:2031 inet_sendmsg+0x25a/0x280 net/ipv4/af_inet.c:851 sock_sendmsg_nosec net/socket.c:718 [inline] and BUG: KMSAN: uninit-value in sctp_assoc_bh_rcv+0x34e/0xbc0 net/sctp/associola.c:987 sctp_assoc_bh_rcv+0x34e/0xbc0 net/sctp/associola.c:987 sctp_inq_push+0x2a3/0x350 net/sctp/inqueue.c:88 sctp_backlog_rcv+0x3c7/0xda0 net/sctp/input.c:331 sk_backlog_rcv+0x142/0x420 include/net/sock.h:1148 __release_sock+0x1d3/0x330 net/core/sock.c:3213 release_sock+0x6b/0x270 net/core/sock.c:3767 sctp_wait_for_connect+0x458/0x820 net/sctp/socket.c:9367 sctp_sendmsg_to_asoc+0x223a/0x2260 net/sctp/socket.c:1886 sctp_sendmsg+0x3910/0x49f0 net/sctp/socket.c:2032 inet_sendmsg+0x269/0x2a0 net/ipv4/af_inet.c:851 sock_sendmsg_nosec net/socket.c:712 [inline] This patch fixes it by linearizing cloned gso packets in sctp_rcv().
CVE-2024-26945	HIGH	8.4	CVSS:3.1/AV:L/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: crypto: iaa - Fix nr_cpus < nr_iaa case If nr_cpus < nr_iaa, the calculated cpus_per_iaa will be 0, which causes a divide-by-0 in rebalance_wq_table(). Make sure cpus_per_iaa is 1 in that case, and also in the nr_iaa == 0 case, even though cpus_per_iaa is never used if nr_iaa == 0, for paranoia.
CVE-2024-48208	HIGH	8.6	CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:L/A:H	pure-ftpd 1.0.51	pure-ftpd before 1.0.52 is vulnerable to Buffer Overflow. There is an out of bounds read in the domlsd() function of the ls.c file.
CVE-2020-17163	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:N/UI:R/S:U/C:H/I:H/A:H	python3 3.12.11	Visual Studio Code Python Extension Remote Code Execution Vulnerability
CVE-2025-49714	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:N/UI:R/S:U/C:H/I:H/A:H	python3 3.12.11	Trust boundary violation in Visual Studio Code - Python extension allows an unauthorized attacker to execute code locally.
CVE-2024-49050	HIGH	8.8	CVSS:3.1/AV:N/AC:L/PR:N/UI:R/S:U/C:H/I:H/A:H	python3 3.12.11	Visual Studio Code Python Extension Remote Code Execution Vulnerability
CVE-2020-1192	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:N/UI:R/S:U/C:H/I:H/A:H	python3 3.12.11	A remote code execution vulnerability exists in Visual Studio Code when the Python extension loads workspace settings from a notebook file, aka 'Visual Studio Code Python Extension Remote Code Execution Vulnerability'. This CVE ID is unique from CVE-2020-1171.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2020-1171	HIGH	8.8	CVSS:3.1/AV:N/AC:L/PR:N/UI:R/S:U/C:H/I:H/A:H	python3 3.12.11	A remote code execution vulnerability exists in Visual Studio Code when the Python extension loads configuration files after opening a project, aka 'Visual Studio Code Python Extension Remote Code Execution Vulnerability'. This CVE ID is unique from CVE-2020-1192.
CVE-2022-2347	HIGH	7.1	CVSS:3.1/AV:P/AC:H/PR:N/UI:N/S:C/C:H/I:H/A:H	u-boot-rittal 2018.09	There exists an unchecked length field in UBoot. The U-Boot DFU implementation does not bound the length field in USB DFU download setup packets, and it does not verify that the transfer direction corresponds to the specified command. Consequently, if a physical attacker crafts a USB DFU download setup packet with a `wLength` greater than 4096 bytes, they can write beyond the heap-allocated request buffer.
CVE-2024-57258	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	u-boot-rittal 2018.09	Integer overflows in memory allocation in Das U-Boot before 2025.01-rc1 occur for a crafted squashfs filesystem via sbrk, via request2size, or because ptrdiff_t is mishandled on x86_64.
CVE-2024-42040	HIGH	8.1	CVSS:3.1/AV:A/AC:L/PR:N/UI:N/S:U/C:H/I:N/A:H	u-boot-rittal 2018.09	Buffer Overflow vulnerability in the net/bootp.c in DENEX U-Boot from its initial commit in 2002 (3861aa5) up to today on any platform allows an attacker on the local network to leak memory from four up to 32 bytes of memory stored behind the packet to the network depending on the later use of DHCP-provided parameters via crafted DHCP responses.
CVE-2026-33243	HIGH	8.2	CVSS:3.1/AV:L/AC:L/PR:H/UI:N/S:C/C:H/I:H/A:H	u-boot-rittal 2018.09	barebox is a bootloader. In barebox from version 2016.03.0 to before version 2026.03.1 (and the corresponding backport to 2025.09.3), an attacker could exploit a FIT signature verification vulnerability to trick the bootloader into booting different images than those that were verified as part of a signed configuration. mkimage(1) sets the hashed-nodes property of the FIT signature node to list which nodes of the FIT were hashed as part of the signing process as these will need to be verified later on by the bootloader. However, hashed-nodes itself is not part of the hash and could therefore be modified to allow booting different images than those that have been verified. This issue has been patched in barebox versions 2026.03.1 and backported to 2025.09.3.
CVE-2019-13103	HIGH	7.1	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:H/A:H	u-boot-rittal 2018.09	A crafted self-referential DOS partition table will cause all Das U-Boot versions through 2019.07-rc4 to infinitely recurse, causing the stack to grow infinitely and eventually either crash or overwrite other data.
CVE-2021-27097	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:N/UI:R/S:U/C:H/I:H/A:H	u-boot-rittal 2018.09	The boot loader in Das U-Boot before 2021.04-rc2 mishandles a modified FIT.
CVE-2021-27138	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:N/UI:R/S:U/C:H/I:H/A:H	u-boot-rittal 2018.09	The boot loader in Das U-Boot before 2021.04-rc2 mishandles use of unit addresses in a FIT.
CVE-2019-13104	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:N/UI:R/S:U/C:H/I:H/A:H	u-boot-rittal 2018.09	In Das U-Boot versions 2016.11-rc1 through 2019.07-rc4, an underflow can cause memcpy() to overwrite a very large amount of data (including the whole stack) while reading a crafted ext4 filesystem.
CVE-2018-18440	HIGH	7.8	CVSS:3.0/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	u-boot-rittal 2018.09	DENX U-Boot through 2018.09-rc1 has a locally exploitable buffer overflow via a crafted kernel image because filesystem loading is mishandled.
CVE-2019-13106	HIGH	7.8	CVSS:3.1/AV:L/AC:L/PR:N/UI:R/S:U/C:H/I:H/A:H	u-boot-rittal 2018.09	Das U-Boot versions 2016.09 through 2019.07-rc4 can memset() too much data while reading a crafted ext4 filesystem, which results in a stack buffer overflow and likely code execution.
CVE-2025-10966	MEDIUM	4.3	CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:U/C:L/I:N/A:N	curl 8.7.1	curl's code for managing SSH connections when SFTP was done using the wolfSSH powered backend was flawed and missed host verification mechanisms. This prevents curl from detecting MITM attackers and more.
CVE-2024-49934	MEDIUM	4.6	CVSS:3.1/AV:P/AC:L/PR:N/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: fs/inode: Prevent dump_mapping() accessing invalid dentry.d_name.name It's observed that a crash

CVE	Severity	CVSS	CVSS Vector	Component	Description
					occurs during hot-remove a memory device, in which user is accessing the hugetlb. See calltrace as following: ----------[cut here]---------- WARNING: CPU: 1 PID: 14045 at arch/x86/mm/fault.c:1278 do_user_addr_fault+0x2a0/0x790 Modules linked in: kmem device_dax cxl_mem cxl_pmem cxl_port cxl_pci dax_hmem dax_pmem nd_pmem cxl_acpi nd_btt cxl_core crc32c_intel nvme virtiofs fuse nvme_core nfit libnvdimm dm_multipath scsi_dh_rdac scsi_dh_emc s mirror dm_region_hash dm_log dm_mod CPU: 1 PID: 14045 Comm: daxctl Not tainted 6.10.0-rc2-lizhijian+ #492 Hardware name: QEMU Standard PC (Q35 + ICH9, 2009), BIOS rel-1.16.3-0-ga6ed6b701f0a-prebuilt.qemu.org 04/01/2014 RIP: 0010:do_user_addr_fault+0x2a0/0x790 Code: 48 8b 00 a8 04 0f 84 b5 fe ff ff e9 1c ff ff ff 4c 89 e9 4c 89 e2 be 01 00 00 00 bf 02 00 00 00 e8 b5 ef 24 00 e9 42 fe ff ff <0f> 0b 48 83 c4 08 4c 89 ea 48 89 ee 4c 89 e7 5b 5d 41 5c 41 5d 41 RSP: 0000:ffffc90000a575f0 EFLAGS: 00010046 RAX: ffff88800c303600 RBX: 0000000000000000 RCX: 0000000000000000 RDX: 0000000000001000 RSI: ffffffff82504162 RDI: ffffffff824b2c36 RBP: 0000000000000000 R08: 0000000000000000 R09: 0000000000000000 R10: 0000000000000000 R11: 0000000000000000 R12: ffffc90000a57658 R13: 0000000000001000 R14: ffff88800bc2e040 R15: 0000000000000000 FS: 00007f51cb57d880(0000) GS:ffff88807fd00000(0000) knlGS:0000000000000000 CS: 0010 DS: 0000 ES: 0000 CR0: 0000000080050033 CR2: 0000000000001000 CR3: 00000000072e2004 CR4: 00000000001706f0 DR0: 0000000000000000 DR1: 0000000000000000 DR2: 0000000000000000 DR3: 0000000000000000 DR6: 00000000fffe0ff0 DR7: 0000000000000400 Call Trace: ? __warn+0x8d/0x190 ? do_user_addr_fault+0x2a0/0x790 ? report_bug+0x1c3/0x1d0 ? handle_bug+0x3c/0x70 ? exc_invalid_op+0x14/0x70 ? asm_exc_invalid_op+0x16/0x20 ? do_user_addr_fault+0x2a0/0x790 ? exc_page_fault+0x31/0x200 exc_page_fault+0x68/0x200 <...snip...> BUG: unable to handle page fault for address: 0000000000001000 #PF: supervisor read access in kernel mode #PF: error_code(0x0000) - not-present page PGD 800000000ad92067 P4D 800000000ad92067 PUD 7677067 PMD 0 Oops: Oops: 0000 [#1] PREEMPT SMP PTI ---[end trace 0000000000000000]--- BUG: unable to handle page fault for address: 0000000000001000 #PF: supervisor read access in kernel mode #PF: error_code(0x0000) - not-present page PGD 800000000ad92067 P4D 800000000ad92067 PUD 7677067 PMD 0 Oops: Oops: 0000 [#1] PREEMPT SMP PTI CPU: 1 PID: 14045 Comm: daxctl Kdump: loaded Tainted: G W 6.10.0-rc2-lizhijian+ #492 Hardware name: QEMU Standard PC (Q35 + ICH9, 2009), BIOS rel-1.16.3-0-ga6ed6b701f0a-prebuilt.qemu.org 04/01/2014 RIP: 0010:dentry_name+0x1f4/0x440 <...snip...> ? dentry_name+0x2fa/0x440 vsnprintf+0x1f3/0x4f0 vprintk_store+0x23a/0x540 vprintk_emit+0x6d/0x330 _printk+0x58/0x80 dump_mapping+0x10b/0x1a0 ? __pfx_free_object_rcu+0x10/0x10 __dump_page+0x26b/0x3e0 ? vprintk_emit+0xe0/0x330 ? _printk+0x58/0x80 ? dump_page+0x17/0x50 dump_page+0x17/0x50 do_migrate_range+0x2f7/0x7f0 ? do_migrate_range+0x42/0x7f0 ? offline_pages+0x2f4/0x8c0 offline_pages+0x60a/0x8c0 memory_subsys_offline+0x9f/0x1c0 ? lockdep_hardirqs_on+0x77/0x100 ? _raw_spin_unlock_irqrestore+0x38/0x60 device_offline+0xe3/0x110 state_store+0x6e/0xc0

CVE	Severity	CVSS	CVSS Vector	Component	Description
					kernfs_fop_write_iter+0x143/0x200 vfs_write+0x39f/0x560 ksys_write+0x65/0xf0 do_syscall_64+0x62/0x130 Previously, some sanity check have been done in dump_mapping() before the print facility parsing '%pd' though, it's still possible to run into an invalid dentry.d_name.name. Since dump_mapping() only needs to dump the filename only, retrieve it by itself in a safer way to prevent an unnecessary crash. Note that either retrieving the filename with '%pd' or strncpy_from_kernel_nofault(), the filename could be unreliable.
CVE-2025-38515	MEDIUM	4.7	CVSS:3.1/AV:L/AC:H/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: drm/sched: Increment job count before swapping tail spsc queue A small race exists between spsc_queue_push and the run-job worker, in which spsc_queue_push may return not-first while the run-job worker has already idled due to the job count being zero. If this race occurs, job scheduling stops, leading to hangs while waiting on the job's DMA fences. Seal this race by incrementing the job count before appending to the SPSC queue. This race was observed on a drm-tip 6.16-rc1 build with the Xe driver in an SVM test case.
CVE-2024-50010	MEDIUM	4.7	CVSS:3.1/AV:L/AC:H/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: exec: don't WARN for racy path_noexec check Both i_mode and noexec checks wrapped in WARN_ON stem from an artifact of the previous implementation. They used to legitimately check for the condition, but that got moved up in two commits: 633fb6ac3980 ("exec: move S_ISREG() check earlier") 0fd338b2d2cd ("exec: move path_noexec() check earlier") Instead of being removed said checks are WARN_ON'ed instead, which has some debug value. However, the spurious path_noexec check is racy, resulting in unwarranted warnings should someone race with setting the noexec flag. One can note there is more to perm-checking whether execve is allowed and none of the conditions are guaranteed to still hold after they were tested for. Additionally this does not validate whether the code path did any perm checking to begin with -- it will pass if the inode happens to be regular. Keep the redundant path_noexec() check even though it's mindless nonsense checking for guarantee that isn't given so drop the WARN. Reword the commentary and do small tidy ups while here. [brauner: keep redundant path_noexec() check]
CVE-2025-39697	MEDIUM	4.7	CVSS:3.1/AV:L/AC:H/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: NFS: Fix a race when updating an existing write After nfs_lock_and_join_requests() tests for whether the request is still attached to the mapping, nothing prevents a call to nfs_inode_remove_request() from succeeding until we actually lock the page group. The reason is that whoever called nfs_inode_remove_request() doesn't necessarily have a lock on the page group head. So in order to avoid races, let's take the page group lock earlier in nfs_lock_and_join_requests(), and hold it across the removal of the request in nfs_inode_remove_request().

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38232	MEDIUM	4.7	CVSS:3.1/AV:L/AC:H/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: NFSD: fix race between nfsd registration and exports_proc As of now nfsd calls create_proc_exports_entry() at start of init_nfsd and cleanup by remove_proc_entry() at last of exit_nfsd. Which causes kernel OOPs if there is race between below 2 operations: (i) exportfs -r (ii) mount -t nfsd none /proc/fs/nfsd for 5.4 kernel ARM64: CPU 1: el1_irq+0xbc/0x180 arch_counter_get_cntvct+0x14/0x18 running_clock+0xc/0x18 preempt_count_add+0x88/0x110 prep_new_page+0xb0/0x220 get_page_from_freelist+0x2d8/0x1778 __alloc_pages_nodemask+0x15c/0xef0 __vmalloc_node_range+0x28c/0x478 __vmalloc_node_flags_caller+0x8c/0xb0 kvmalloc_node+0x88/0xe0 nfsd_init_net+0x6c/0x108 [nfsd] ops_init+0x44/0x170 register_pernet_operations+0x114/0x270 register_pernet_subsys+0x34/0x50 init_nfsd+0xa8/0x718 [nfsd] do_one_initcall+0x54/0x2e0 CPU 2 : Unable to handle kernel NULL pointer dereference at virtual address 0000000000000010 PC is at : exports_net_open+0x50/0x68 [nfsd] Call trace: exports_net_open+0x50/0x68 [nfsd] exports_proc_open+0x2c/0x38 [nfsd] proc_reg_open+0xb8/0x198 do_dentry_open+0x1c4/0x418 vfs_open+0x38/0x48 path_openat+0x28c/0xf18 do_filp_open+0x70/0xe8 do_sys_open+0x154/0x248 Sometimes it crashes at exports_net_open() and sometimes cache_seq_next_rcu(). and same is happening on latest 6.14 kernel as well: [0.000000] Linux version 6.14.0-rc5-next-20250304-dirty ... [285.455918] Unable to handle kernel paging request at virtual address 00001f4800001f48 ... [285.464902] pc : cache_seq_next_rcu+0x78/0xa4 ... [285.469695] Call trace: [285.470083] cache_seq_next_rcu+0x78/0xa4 (P) [285.470488] seq_read+0xe0/0x11c [285.470675] proc_reg_read+0x9c/0xf0 [285.470874] vfs_read+0xc4/0x2fc [285.471057] ksys_read+0x6c/0xf4 [285.471231] __arm64_sys_read+0x1c/0x28 [285.471428] invoke_syscall+0x44/0x100 [285.471633] el0_svc_common.constprop.0+0x40/0xe0 [285.471870] do_el0_svc_compat+0x1c/0x34 [285.472073] el0_svc_compat+0x2c/0x80 [285.472265] el0t_32_sync_handler+0x90/0x140 [285.472473] el0t_32_sync+0x19c/0x1a0 [285.472887] Code: f9400885 93407c23 937d7c27 11000421 (f86378a3) [285.473422] ---[end trace 0000000000000000]--- It reproduced simply with below script: while [1] do /exportfs -r done & while [1] do insmod /nfsd.ko mount -t nfsd none /proc/fs/nfsd umount /proc/fs/nfsd rmmod nfsd done & So exporting interfaces to user space shall be done at last and cleanup at first place. With change there is no Kernel OOPs.
CVE-2025-38393	MEDIUM	4.7	CVSS:3.1/AV:L/AC:H/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: NFSv4/pNFS: Fix a race to wake on NFS_LAYOUT_DRAIN We found a few different systems hung up in writeback waiting on the same page lock, and one task waiting on the NFS_LAYOUT_DRAIN bit in pnfs_update_layout(), however the pnfs_layout_hdr's plh_outstanding count was zero. It seems most likely that this is another race between the waiter and waker similar to commit ed0172af5d6f ("SUNRPC: Fix a race to wake a sync task"). Fix it up by applying the advised barrier.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-21732	MEDIUM	4.7	CVSS:3.1/AV:L/AC:H/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: RDMA/mlx5: Fix a race for an ODP MR which leads to CQE with error This patch addresses a race condition for an ODP MR that can result in a CQE with an error on the UMR QP. During the __mlx5_ib_dereg_mr() flow, the following sequence of calls occurs: mlx5_revoke_mr() mlx5r_umr_revoke_mr() mlx5r_umr_post_send_wait() At this point, the lkey is freed from the hardware's perspective. However, concurrently, mlx5_ib_invalidate_range() might be triggered by another task attempting to invalidate a range for the same freed lkey. This task will: - Acquire the umem_odp->umem_mutex lock. - Call mlx5r_umr_update_xlt() on the UMR QP. - Since the lkey has already been freed, this can lead to a CQE error, causing the UMR QP to enter an error state [1]. To resolve this race condition, the umem_odp->umem_mutex lock is now also acquired as part of the mlx5_revoke_mr() scope. Upon successful revoke, we set umem_odp->private which points to that MR to NULL, preventing any further invalidation attempts on its lkey. [1] From dmesg: infiniband rocep8s0f0: dump_cqe:277:(pid 0): WC error: 6, Message: memory bind operation error cqe_dump: 00000000: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 cqe_dump: 00000010: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 cqe_dump: 00000020: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 cqe_dump: 00000030: 00 00 00 00 08 00 78 06 25 00 11 b9 00 0e dd d2 WARNING: CPU: 15 PID: 1506 at drivers/infiniband/hw/mlx5/umr.c:394 mlx5r_umr_post_send_wait+0x15a/0x2b0 [mlx5_ib] Modules linked in: ip6table_mangle ip6table_natip6table_filter ip6_tables iptable_mangle xt_conntrack xt_MASQUERADE nf_conntrack_netlink nfnetlink xt_addrtype iptable_nat nf_nat br_netfilter rpcsec_gss_krb5 auth_rpcgss oid_registry overlay rpcrdma rdma_ucm ib_iser libiscsi scsi_transport_iscsi rdma_cm iw_cm ib_umad ib_ipoib ib_cm mlx5_ib ib_uverbs ib_core fuse mlx5_core CPU: 15 UID: 0 PID: 1506 Comm: ibv_rc_pingpong Not tainted 6.12.0-rc7+ #1626 Hardware name: QEMU Standard PC (Q35 + ICH9, 2009), BIOS rel-1.13.0-0-gf21b5a4aeb02-prebuilt.qemu.org 04/01/2014 RIP: 0010:mlx5r_umr_post_send_wait+0x15a/0x2b0 [mlx5_ib] [..] Call Trace: mlx5r_umr_update_xlt+0x23c/0x3e0 [mlx5_ib] mlx5_ib_invalidate_range+0x2e1/0x330 [mlx5_ib] __mmu_notifier_invalidate_range_start+0x1e1/0x240 zap_page_range_single+0xf1/0x1a0 madvise_vma_behavior+0x677/0x6e0 do_madvise+0x1a2/0x4b0 __x64_sys_madvise+0x25/0x30 do_syscall_64+0x6b/0x140 entry_SYSCALL_64_after_hwframe+0x76/0x7e

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-21892	MEDIUM	4.7	CVSS:3.1/AV:L/AC:H/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: RDMA/mlx5: Fix the recovery flow of the UMR QP This patch addresses an issue in the recovery flow of the UMR QP, ensuring tasks do not get stuck, as highlighted by the call trace [1]. During recovery, before transitioning the QP to the RESET state, the software must wait for all outstanding WRs to complete. Failing to do so can cause the firmware to skip sending some flushed CQEs with errors and simply discard them upon the RESET, as per the IB specification. This race condition can result in lost CQEs and tasks becoming stuck. To resolve this, the patch sends a final WR which serves only as a barrier before moving the QP state to RESET. Once a CQE is received for that final WR, it guarantees that no outstanding WRs remain, making it safe to transition the QP to RESET and subsequently back to RTS, restoring proper functionality. Note: For the barrier WR, we simply reuse the failed and ready WR. Since the QP is in an error state, it will only receive IB_WC_WR_FLUSH_ERR. However, as it serves only as a barrier we don't care about its status. [1] INFO: task rdma_resource_l:1922 blocked for more than 120 seconds. Tainted: G W 6.12.0-rc7+ #1626 "echo 0 > /proc/sys/kernel/hung_task_timeout_secs" disables this message. task:rdma_resource_l state:D stack:0 pid:1922 tgid:1922 ppid:1369 flags:0x00004004 Call Trace: __schedule+0x420/0xd30 schedule+0x47/0x130 schedule_timeout+0x280/0x300 ? mark_held_locks+0x48/0x80 ? lockdep_hardirqs_on_prepare+0xe5/0x1a0 wait_for_completion+0x75/0x130 mlx5r_umr_post_send_wait+0x3c2/0x5b0 [mlx5_ib] ? __pfx_mlx5r_umr_done+0x10/0x10 [mlx5_ib] mlx5r_umr_revoke_mr+0x93/0xc0 [mlx5_ib] __mlx5_ib_dereg_mr+0x299/0x520 [mlx5_ib] ? _raw_spin_unlock_irq+0x24/0x40 ? wait_for_completion+0xfe/0x130 ? rdma_restrack_put+0x63/0xe0 [ib_core] ib_dereg_mr_user+0x5f/0x120 [ib_core] ? lock_release+0xc6/0x280 destroy_hw_idr_uobject+0x1d/0x60 [ib_uverbs] uverbs_destroy_uobject+0x58/0x1d0 [ib_uverbs] uobj_destroy+0x3f/0x70 [ib_uverbs] ib_uverbs_cmd_verbs+0x3e4/0xbb0 [ib_uverbs] ? __pfx_uverbs_destroy_def_handler+0x10/0x10 [ib_uverbs] ? __lock_acquire+0x64e/0x2080 ? mark_held_locks+0x48/0x80 ? find_held_lock+0x2d/0xa0 ? lock_acquire+0xc1/0x2f0 ? ib_uverbs_ioctl+0xcb/0x170 [ib_uverbs] ? __fget_files+0xc3/0x1b0 ib_uverbs_ioctl+0xe7/0x170 [ib_uverbs] ? ib_uverbs_ioctl+0xcb/0x170 [ib_uverbs] __x64_sys_ioctl+0x1b0/0xa70 do_syscall_64+0x6b/0x140 entry_SYSCALL_64_after_hwframe+0x76/0x7e RIP: 0033:0x7f99c918b17b RSP: 002b:00007ffc766d0468 EFLAGS: 00000246 ORIG_RAX: 0000000000000010 RAX: ffffffffffffffda RBX: 00007ffc766d0578 RCX: 00007f99c918b17b RDX: 00007ffc766d0560 RSI: 00000000c0181b01 RDI: 0000000000000003 RBP: 00007ffc766d0540 R08: 00007f99c8f99010 R09: 000000000000bd7e R10: 00007f99c94c1c70 R11: 0000000000000246 R12: 00007ffc766d0530 R13: 000000000000001c R14: 0000000040246a80 R15: 0000000000000000
CVE-2025-22115	MEDIUM	4.7	CVSS:3.1/AV:L/AC:H/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: btrfs: fix block group refcount race in btrfs_create_pending_block_groups() Block group creation is done in two phases, which results in a slightly unintuitive property: a block group can be allocated/deallocated from after btrfs_make_block_group() adds it to the space_info with btrfs_add_bg_to_space_info(), but before creation is completely completed in

CVE	Severity	CVSS	CVSS Vector	Component	Description
					btrfs_create_pending_block_groups(). As a result, it is possible for a block group to go unused and have 'btrfs_mark_bg_unused' called on it concurrently with 'btrfs_create_pending_block_groups'. This causes a number of issues, which were fixed with the block group flag 'BLOCK_GROUP_FLAG_NEW'. However, this fix is not quite complete. Since it does not use the unused_bg_lock, it is possible for the following race to occur: btrfs_create_pending_block_groups btrfs_mark_bg_unused if list_empty // false list_del_init clear_bit else if (test_bit) // true list_move_tail And we get into the exact same broken ref count and invalid new_bgs state for transaction cleanup that BLOCK_GROUP_FLAG_NEW was designed to prevent. The broken refcount aspect will result in a warning like: [1272.943527] refcount_t: underflow; use-after-free. [1272.943967] WARNING: CPU: 1 PID: 61 at lib/refcount.c:28 refcount_warn_saturate+0xba/0x110 [1272.944731] Modules linked in: btrfs virtio_net xor zstd_compress raid6_pq null_blk [last unloaded: btrfs] [1272.945550] CPU: 1 UID: 0 PID: 61 Comm: kworker/u32:1 Kdump: loaded Tainted: G W 6.14.0-rc5+ #108 [1272.946368] Tainted: [W]=WARN [1272.946585] Hardware name: QEMU Standard PC (i440FX + PIIX, 1996), BIOS Arch Linux 1.16.3-1-1 04/01/2014 [1272.947273] Workqueue: btrfs_discard btrfs_discard_workfn [btrfs] [1272.947788] RIP: 0010:refcount_warn_saturate+0xba/0x110 [1272.949532] RSP: 0018:ffffbf1200247df0 EFLAGS: 00010282 [1272.949901] RAX: 0000000000000000 RBX: ffffa14b00e3f800 RCX: 0000000000000000 [1272.950437] RDX: 0000000000000000 RSI: ffffbf1200247c78 RDI: 00000000ffffdfff [1272.950986] RBP: ffffa14b00dc2860 R08: 00000000ffffdfff R09: ffffffff90526268 [1272.951512] R10: ffffffff904762c0 R11: 0000000063666572 R12: ffffa14b00dc28c0 [1272.952024] R13: 0000000000000000 R14: ffffa14b00dc2868 R15: 000001285dcd12c0 [1272.952850] FS: 0000000000000000(0000) GS:ffffa14d33c40000(0000) knlGS:0000000000000000 [1272.953458] CS: 0010 DS: 0000 ES: 0000 CR0: 0000000080050033 [1272.953931] CR2: 00007f838cbda000 CR3: 000000010104e000 CR4: 00000000000006f0 [1272.954474] Call Trace: [1272.954655] [1272.954812] ? refcount_warn_saturate+0xba/0x110 [1272.955173] ? __warn.cold+0x93/0xd7 [1272.955487] ? refcount_warn_saturate+0xba/0x110 [1272.955816] ? report_bug+0xe7/0x120 [1272.956103] ? handle_bug+0x53/0x90 [1272.956424] ? exc_invalid_op+0x13/0x60 [1272.956700] ? asm_exc_invalid_op+0x16/0x20 [1272.957011] ? refcount_warn_saturate+0xba/0x110 [1272.957399] btrfs_discard_cancel_work.cold+0x26/0x2b [btrfs] [1272.957853] btrfs_put_block_group.cold+0x5d/0x8e [btrfs] [1272.958289] btrfs_discard_workfn+0x194/0x380 [btrfs] [1272.958729] process_one_work+0x130/0x290 [1272.959026] worker_thread+0x2ea/0x420 [1272.959335] ? __pfx_worker_thread+0x10/0x10 [1272.959644] kthread+0xd7/0x1c0 [1272.959872] ? __pfx_kthread+0x10/0x10 [1272.960172] ret_from_fork+0x30/0x50 [1272.960474] ? __pfx_kthread+0x10/0x10 [1272.960745] ret_from_fork_asm+0x1a/0x30 [1272.961035] [1272.961238] ---[end trace 0000000000000000]--- Though we have seen them in the async discard workfn as well. It is most likely to happen after a relocation finishes which cancels discard, tears down the block group, etc. Fix this fully by taking the lock arou --- truncated---

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-39927	MEDIUM	4.7	CVSS:3.1/AV:L/AC:H/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ceph: fix race condition validating r_parent before applying state Add validation to ensure the cached parent directory inode matches the directory info in MDS replies. This prevents client-side race conditions where concurrent operations (e.g. rename) cause r_parent to become stale between request initiation and reply processing, which could lead to applying state changes to incorrect directory inodes. [idryomov: folded a kerneldoc fixup and a follow-up fix from Alex to move CEPH_CAP_PIN reference when r_parent is updated: When the parent directory lock is not held, req->r_parent can become stale and is updated to point to the correct inode. However, the associated CEPH_CAP_PIN reference was not being adjusted. The CEPH_CAP_PIN is a reference on an inode that is tracked for accounting purposes. Moving this pin is important to keep the accounting balanced. When the pin was not moved from the old parent to the new one, it created two problems: The reference on the old, stale parent was never released, causing a reference leak. A reference for the new parent was never acquired, creating the risk of a reference underflow later in ceph_mdsc_release_request(). This patch corrects the logic by releasing the pin from the old parent and acquiring it for the new parent when r_parent is switched. This ensures reference accounting stays balanced.]
CVE-2025-38687	MEDIUM	4.7	CVSS:3.1/AV:L/AC:H/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: comedi: fix race between polling and detaching syzbot reports a use-after-free in comedi in the below link, which is due to comedi gladly removing the allocated async area even though poll requests are still active on the wait_queue_head inside of it. This can cause a use-after-free when the poll entries are later triggered or removed, as the memory for the wait_queue_head has been freed. We need to check there are no tasks queued on any of the subdevices' wait queues before allowing the device to be detached by the `COMEDI_DEVCONFIG` ioctl. Tasks will read-lock `dev->attach_lock` before adding themselves to the subdevice wait queue, so fix the problem in the `COMEDI_DEVCONFIG` ioctl handler by write-locking `dev->attach_lock` before checking that all of the subdevices are safe to be deleted. This includes testing for any sleepers on the subdevices' wait queues. It remains locked until the device has been detached. This requires the `comedi_device_detach()` function to be refactored slightly, moving the bulk of it into new function `comedi_device_detach_locked()`. Note that the refactor of `comedi_device_detach()` results in `comedi_device_cancel_all()` now being called while `dev->attach_lock` is write-locked, which wasn't the case previously, but that does not matter. Thanks to Jens Axboe for diagnosing the problem and co-developing this patch.
CVE-2024-36024	MEDIUM	4.7	CVSS:3.1/AV:L/AC:H/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: drm/amd/display: Disable idle reallow as part of command/gpint execution [Why] Workaroud for a race condition where DMCUB is in the process of committing to IPS1 during the handshake causing us to miss the transition into IPS2 and touch the INBOX1 RPTR causing a HW hang. [How] Disable the reallow to ensure that we have enough of a gap between entry and exit and we're not seeing back-to-back wake_and_executes.
CVE-2025-38104	MEDIUM	4.7	CVSS:3.1/AV:L/AC:H/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: drm/amdgpu: Replace Mutex with Spinlock for RLCG register access to avoid Priority Inversion in SRIOV RLCG Register Access is a way for virtual functions to safely

CVE	Severity	CVSS	CVSS Vector	Component	Description
					access GPU registers in a virtualized environment., including TLB flushes and register reads. When multiple threads or VFs try to access the same registers simultaneously, it can lead to race conditions. By using the RLCG interface, the driver can serialize access to the registers. This means that only one thread can access the registers at a time, preventing conflicts and ensuring that operations are performed correctly. Additionally, when a low-priority task holds a mutex that a high-priority task needs, ie., If a thread holding a spinlock tries to acquire a mutex, it can lead to priority inversion. register access in amdgpu_virt_rlcg_reg_rw especially in a fast code path is critical. The call stack shows that the function amdgpu_virt_rlcg_reg_rw is being called, which attempts to acquire the mutex. This function is invoked from amdgpu_sriov_wreg, which in turn is called from gmc_v11_0_flush_gpu_tlb. The [BUG: Invalid wait context] indicates that a thread is trying to acquire a mutex while it is in a context that does not allow it to sleep (like holding a spinlock). Fixes the below: [253.013423] ============================= [253.013434] [BUG: Invalid wait context] [253.013446] 6.12.0-amdstaging-drm-next-lol-050225 #14 Tainted: G U OE [253.013464] ----------------------------- [253.013475] kworker/0:1/10 is trying to lock: [253.013487] ffff9f30542e3cf8 (&adev->virt.rlcg_reg_lock){+.+.}-{3:3}, at: amdgpu_virt_rlcg_reg_rw+0xf6/0x330 [amdgpu] [253.013815] other info that might help us debug this: [253.013827] context-{4:4} [253.013835] 3 locks held by kworker/0:1/10: [253.013847] #0: ffff9f3040050f58 ((wq_completion)events){+.+.}-{0:0}, at: process_one_work+0x3f5/0x680 [253.013877] #1: ffffb789c008be40 ((work_completion)(&wfc.work)){+.+.}-{0:0}, at: process_one_work+0x1d6/0x680 [253.013905] #2: ffff9f3054281838 (&adev->gmc.invalidate_lock){+.+.}-{2:2}, at: gmc_v11_0_flush_gpu_tlb+0x198/0x4f0 [amdgpu] [253.014154] stack backtrace: [253.014164] CPU: 0 UID: 0 PID: 10 Comm: kworker/0:1 Tainted: G U OE 6.12.0-amdstaging-drm-next-lol-050225 #14 [253.014189] Tainted: [U]=USER, [O]=OOT_MODULE, [E]=UNSIGNED_MODULE [253.014203] Hardware name: Microsoft Corporation Virtual Machine/Virtual Machine, BIOS Hyper-V UEFI Release v4.1 11/18/2024 [253.014224] Workqueue: events work_for_cpu_fn [253.014241] Call Trace: [253.014250] [253.014260] dump_stack_lvl+0x9b/0xf0 [253.014275] dump_stack+0x10/0x20 [253.014287] __lock_acquire+0xa47/0x2810 [253.014303] ? srso_alias_return_thunk+0x5/0xfbef5 [253.014321] lock_acquire+0xd1/0x300 [253.014333] ? amdgpu_virt_rlcg_reg_rw+0xf6/0x330 [amdgpu] [253.014562] ? __lock_acquire+0xa6b/0x2810 [253.014578] __mutex_lock+0x85/0xe20 [253.014591] ? amdgpu_virt_rlcg_reg_rw+0xf6/0x330 [amdgpu] [253.014782] ? sched_clock_noinstr+0x9/0x10 [253.014795] ? srso_alias_return_thunk+0x5/0xfbef5 [253.014808] ? local_clock_noinstr+0xe/0xc0 [253.014822] ? amdgpu_virt_rlcg_reg_rw+0xf6/0x330 [amdgpu] [253.015012] ? srso_alias_return_thunk+0x5/0xfbef5 [253.015029] mutex_lock_nested+0x1b/0x30 [253.015044] ? mutex_lock_nested+0x1b/0x30 [253.015057] amdgpu_virt_rlcg_reg_rw+0xf6/0x330 [amdgpu] [253.015249] amdgpu_sriov_wreg+0xc5/0xd0 [amdgpu] [253.015435] gmc_v11_0_flush_gpu_tlb+0x44b/0x4f0 [amdgpu] [253.015667] gfx_v11_0_hw_init+0x499/0x29c0 [amdgpu] [253.015901] ? __pfx_smu_v13_0_update_pcie_parameters+0x10/0x10 [amdgpu] [253.016159] ? srso_alias_return_thunk+0x5/0xfbef5 [

CVE	Severity	CVSS	CVSS Vector	Component	Description
					253.016173] ? smu_hw_init+0x18d/0x300 [amdgpu] [253.016403] amdgpu_device_init+0x29ad/0x36a0 [amdgpu] [253.016614] amdgpu_driver_load_kms+0x1a/0xc0 [amdgpu] [253.0170 ---truncated---
CVE-2025-39813	MEDIUM	4.7	CVSS:3.1/AV:L/AC:H/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ftrace: Fix potential warning in trace_printk_seq during ftrace_dump When calling ftrace_dump_one() concurrently with reading trace_pipe, a WARN_ON_ONCE() in trace_printk_seq() can be triggered due to a race condition. The issue occurs because: CPU0 (ftrace_dump) CPU1 (reader) echo z > /proc/sysrq-trigger !trace_empty(&iter) trace_iterator_reset(&iter) <- len = size = 0 cat /sys/kernel/tracing/trace_pipe trace_find_next_entry_inc(&iter) __find_next_entry ring_buffer_empty_cpu <- all empty return NULL trace_printk_seq(&iter.seq) WARN_ON_ONCE(s->seq.len >= s->seq.size) In the context between trace_empty() and trace_find_next_entry_inc() during ftrace_dump, the ring buffer data was consumed by other readers. This caused trace_find_next_entry_inc to return NULL, failing to populate `iter.seq`. At this point, due to the prior trace_iterator_reset, both `iter.seq.len` and `iter.seq.size` were set to 0. Since they are equal, the WARN_ON_ONCE condition is triggered. Move the trace_printk_seq() into the if block that checks to make sure the return value of trace_find_next_entry_inc() is non-NULL in ftrace_dump_one(), ensuring the 'iter.seq' is properly populated before subsequent operations.
CVE-2025-39961	MEDIUM	4.7	CVSS:3.1/AV:L/AC:H/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: iommu/amd/pgtbl: Fix possible race while increase page table level The AMD IOMMU host page table implementation supports dynamic page table levels (up to 6 levels), starting with a 3-level configuration that expands based on IOVA address. The kernel maintains a root pointer and current page table level to enable proper page table walks in alloc_pte()/fetch_pte() operations. The IOMMU IOVA allocator initially starts with 32-bit address and onces its exhuasted it switches to 64-bit address (max address is determined based on IOMMU and device DMA capability). To support larger IOVA, AMD IOMMU driver increases page table level. But in unmap path (iommu_v1_unmap_pages()), fetch_pte() reads pgtable->[root/mode] without lock. So its possible that in exteme corner case, when increase_address_space() is updating pgtable->[root/mode], fetch_pte() reads wrong page table level (pgtable->mode). It does compare the value with level encoded in page table and returns NULL. This will result is iommu_unmap ops to fail and upper layer may retry/log WARN_ON. CPU 0 CPU 1 ------ ------ map pages unmap pages alloc_pte() -> increase_address_space() iommu_v1_unmap_pages() -> fetch_pte() pgtable->root = pte (new root value) READ pgtable->[mode/root] Reads new root, old mode Updates mode (pgtable->mode += 1) Since Page table level updates are infrequent and already synchronized with a spinlock, implement seqcount to enable lock-free read operations on the read path.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-40039	MEDIUM	4.7	CVSS:3.1/AV:L/AC:H/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ksmbd: Fix race condition in RPC handle list access The 'sess->rpc_handle_list' XArray manages RPC handles within a ksmbd session. Access to this list is intended to be protected by 'sess->rpc_lock' (an rw_semaphore). However, the locking implementation was flawed, leading to potential race conditions. In ksmbd_session_rpc_open(), the code incorrectly acquired only a read lock before calling xa_store() and xa_erase(). Since these operations modify the XArray structure, a write lock is required to ensure exclusive access and prevent data corruption from concurrent modifications. Furthermore, ksmbd_session_rpc_method() accessed the list using xa_load() without holding any lock at all. This could lead to reading inconsistent data or a potential use-after-free if an entry is concurrently removed and the pointer is dereferenced. Fix these issues by: 1. Using down_write() and up_write() in ksmbd_session_rpc_open() to ensure exclusive access during XArray modification, and ensuring the lock is correctly released on error paths. 2. Adding down_read() and up_read() in ksmbd_session_rpc_method() to safely protect the lookup.
CVE-2025-38561	MEDIUM	4.7	CVSS:3.1/AV:L/AC:H/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ksmbd: fix Preauh_HashValue race condition If client send multiple session setup requests to ksmbd, Preauh_HashValue race condition could happen. There is no need to free sess->Preauh_HashValue at session setup phase. It can be freed together with session at connection termination phase.
CVE-2025-38681	MEDIUM	4.7	CVSS:3.1/AV:L/AC:H/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: mm/ptdump: take the memory hotplug lock inside ptdump_walk_pgd() Memory hot remove unmaps and tears down various kernel page table regions as required. The ptdump code can race with concurrent modifications of the kernel page tables. When leaf entries are modified concurrently, the dump code may log stale or inconsistent information for a VA range, but this is otherwise not harmful. But when intermediate levels of kernel page table are freed, the dump code will continue to use memory that has been freed and potentially reallocated for another purpose. In such cases, the ptdump code may dereference bogus addresses, leading to a number of potential problems. To avoid the above mentioned race condition, platforms such as arm64, riscv and s390 take memory hotplug lock, while dumping kernel page table via the sysfs interface /sys/kernel/debug/kernel_page_tables. Similar race condition exists while checking for pages that might have been marked W+X via /sys/kernel/debug/kernel_page_tables/check_wx_pages which in turn calls ptdump_check_wx(). Instead of solving this race condition again, let's just move the memory hotplug lock inside generic ptdump_check_wx() which will benefit both the scenarios. Drop get_online_mems() and put_online_mems() combination from all existing platform ptdump code paths.
CVE-2025-39754	MEDIUM	4.7	CVSS:3.1/AV:L/AC:H/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: mm/smaps: fix race between smaps_hugetlb_range and migration smaps_hugetlb_range() handles the pte without holdling ptl, and may be concurrenct with migration, leaing to BUG_ON in pfn_swap_entry_to_page(). The race is as follows. smaps_hugetlb_range migrate_pages huge_ptep_get remove_migration_ptes folio_unlock pfn_swap_entry_folio BUG_ON To fix it, hold ptl lock in smaps_hugetlb_range().

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2026-23169	MEDIUM	4.7	CVSS:3.1/AV:L/AC:H/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: mptcp: fix race in mptcp_pm_nl_flush_addrs_doit() syzbot and Eulgyu Kim reported crashes in mptcp_pm_nl_get_local_id() and/or mptcp_pm_nl_is_backup() Root cause is list_splice_init() in mptcp_pm_nl_flush_addrs_doit() which is not RCU ready. list_splice_init_rcu() can not be called here while holding pernet->lock spinlock. Many thanks to Eulgyu Kim for providing a repro and testing our patches.
CVE-2025-38617	MEDIUM	4.7	CVSS:3.1/AV:L/AC:H/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net/packet: fix a race in packet_set_ring() and packet_notifier() When packet_set_ring() releases po->bind_lock, another thread can run packet_notifier() and process an NETDEV_UP event. This race and the fix are both similar to that of commit 15fe076edea7 ("net/packet: fix a race in packet_bind() and packet_notifier()"). There too the packet_notifier NETDEV_UP event managed to run while a po->bind_lock critical section had to be temporarily released. And the fix was similarly to temporarily set po->num to zero to keep the socket unhooked until the lock is retaken. The po->bind_lock in packet_set_ring and packet_notifier precede the introduction of git history.
CVE-2025-38477	MEDIUM	4.7	CVSS:3.1/AV:L/AC:H/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net/sched: sch_qfq: Fix race condition on qfq_aggregate A race condition can occur when 'agg' is modified in qfq_change_agg (called during qfq_enqueue) while other threads access it concurrently. For example, qfq_dump_class may trigger a NULL dereference, and qfq_delete_class may cause a use-after-free. This patch addresses the issue by: 1. Moved qfq_destroy_class into the critical section. 2. Added sch_tree_lock protection to qfq_dump_class and qfq_dump_class_stats.
CVE-2024-53124	MEDIUM	4.7	CVSS:3.1/AV:L/AC:H/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net: fix data-races around sk->sk_forward_alloc Syzkaller reported this warning: ------------[cut here]------------ WARNING: CPU: 0 PID: 16 at net/ipv4/af_inet.c:156 inet_sock_destruct+0x1c5/0x1e0 Modules linked in: CPU: 0 UID: 0 PID: 16 Comm: ksoftirqd/0 Not tainted 6.12.0-rc5 #26 Hardware name: QEMU Standard PC (i440FX + PIIX, 1996), BIOS 1.15.0-1 04/01/2014 RIP: 0010:inet_sock_destruct+0x1c5/0x1e0 Code: 24 12 4c 89 e2 5b 48 c7 c7 98 ec bb 82 41 5c e9 d1 18 17 ff 4c 89 e6 5b 48 c7 c7 d0 ec bb 82 41 5c e9 bf 18 17 ff 0f 0b eb 83 <0f> 0b eb 97 0f 0b eb 87 0f 0b e9 68 ff ff ff 66 66 2e 0f 1f 84 00 RSP: 0018:ffffc9000008bd90 EFLAGS: 00010206 RAX: 0000000000000300 RBX: ffff88810b172a90 RCX: 0000000000000007 RDX: 0000000000000002 RSI: 0000000000000300 RDI: ffff88810b172a00 RBP: ffff88810b172a00 R08: ffff888104273c00 R09: 0000000000100007 R10: 0000000000020000 R11: 0000000000000006 R12: ffff88810b172a00 R13: 0000000000000004 R14: 0000000000000000 R15: ffff888237c31f78 FS: 0000000000000000(0000) GS:ffff888237c00000(0000) knlGS:0000000000000000 CS: 0010 DS: 0000 ES: 0000 CR0: 0000000080050033 CR2: 00007ffc63fecac8 CR3: 000000000342e000 CR4: 00000000000006f0 DR0: 0000000000000000 DR1: 0000000000000000 DR2: 0000000000000000 DR3: 0000000000000000 DR6: 00000000fffe0ff0 DR7: 0000000000000400 Call Trace: ? __warn+0x88/0x130 ? inet_sock_destruct+0x1c5/0x1e0 ? report_bug+0x18e/0x1a0 ? handle_bug+0x53/0x90 ? exc_invalid_op+0x18/0x70 ? asm_exc_invalid_op+0x1a/0x20 ?

CVE	Severity	CVSS	CVSS Vector	Component	Description
					inet_sock_destruct+0x1c5/0x1e0 __sk_destruct+0x2a/0x200 rcu_do_batch+0x1aa/0x530 ? rcu_do_batch+0x13b/0x530 rcu_core+0x159/0x2f0 handle_softirqs+0xd3/0x2b0 ? __pfx_smpboot_thread_fn+0x10/0x10 run_ksoftirqd+0x25/0x30 smpboot_thread_fn+0xdd/0x1d0 kthread+0xd3/0x100 ? __pfx_kthread+0x10/0x10 ret_from_fork+0x34/0x50 ? __pfx_kthread+0x10/0x10 ret_from_fork_asm+0x1a/0x30 ---[end trace 0000000000000000]--- Its possible that two threads call tcp_v6_do_rcv()/sk_forward_alloc_add() concurrently when sk->sk_state == TCP_LISTEN with sk->sk_lock unlocked, which triggers a data-race around sk->sk_forward_alloc: tcp_v6_rcv tcp_v6_do_rcv skb_clone_and_charge_r sk_rmem_schedule __sk_mem_schedule sk_forward_alloc_add() skb_set_owner_r sk_mem_charge sk_forward_alloc_add() __kfree_skb skb_release_all skb_release_head_state sock_rfree sk_mem_uncharge sk_forward_alloc_add() sk_mem_reclaim // set local var reclaimable __sk_mem_reclaim sk_forward_alloc_add() In this syzkaller testcase, two threads call tcp_v6_do_rcv() with skb->truesize=768, the sk_forward_alloc changes like this: (cpu 1) | (cpu 2) | sk_forward_alloc ... | ... | 0 __sk_mem_schedule() | | +4096 = 4096 | __sk_mem_schedule() | +4096 = 8192 sk_mem_charge() | | -768 = 7424 | sk_mem_charge() | -768 = 6656 ... | ... | sk_mem_uncharge() | | +768 = 7424 reclaimable=7424 | | sk_mem_uncharge() | +768 = 8192 | reclaimable=8192 | __sk_mem_reclaim() | | -4096 = 4096 | __sk_mem_reclaim() | -8192 = -4096 != 0 The skb_clone_and_charge_r() should not be called in tcp_v6_do_rcv() when sk->sk_state is TCP_LISTEN, it happens later in tcp_v6_syn_recv_sock(). Fix the same issue in dccp_v6_do_rcv().
CVE-2025-21651	MEDIUM	4.7	CVSS:3.1/AV:L/AC:H/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net: hns3: don't auto enable misc vector Currently, there is a time window between misc irq enabled and service task inited. If an interrupte is reported at this time, it will cause warning like below: [16.324639] Call trace: [16.324641] __queue_delayed_work+0xb8/0xe0 [16.324643] mod_delayed_work_on+0x78/0xd0 [16.324655] hclge_errhand_task_schedule+0x58/0x90 [hclge] [16.324662] hclge_misc_irq_handle+0x168/0x240 [hclge] [16.324666] __handle_irq_event_percpu+0x64/0x1e0 [16.324667] handle_irq_event+0x80/0x170 [16.324670] handle_fasteoi_edge_irq+0x110/0x2bc [16.324671] __handle_domain_irq+0x84/0xfc [16.324673] gic_handle_irq+0x88/0x2c0 [16.324674] el1_irq+0xb8/0x140 [16.324677] arch_cpu_idle+0x18/0x40 [16.324679] default_idle_call+0x5c/0x1bc [16.324682] cpuidle_idle_call+0x18c/0x1c4 [16.324684] do_idle+0x174/0x17c [16.324685] cpu_startup_entry+0x30/0x6c [16.324687] secondary_start_kernel+0x1a4/0x280 [16.324688] ---[end trace 6aa0bff672a964aa]--- So don't auto enable misc vector when request irq..

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38717	MEDIUM	4.7	CVSS:3.1/AV:L/AC:H/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net: kcm: Fix race condition in kcm_unattach() syzbot found a race condition when kcm_unattach(psock) and kcm_release(kcm) are executed at the same time. kcm_unattach() is missing a check of the flag kcm->tx_stopped before calling queue_work(). If the kcm has a reserved psock, kcm_unattach() might get executed between cancel_work_sync() and unreserve_psock() in kcm_release(), requeuing kcm->tx_work right before kcm gets freed in kcm_done(). Remove kcm->tx_stopped and replace it by the less error-prone disable_work_sync().
CVE-2026-23126	MEDIUM	4.7	CVSS:3.1/AV:L/AC:H/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: netdevsim: fix a race issue related to the operation on bpf_bound_progs list The netdevsim driver lacks a protection mechanism for operations on the bpf_bound_progs list. When the nsim_bpf_create_prog() performs list_add_tail, it is possible that nsim_bpf_destroy_prog() is simultaneously performs list_del. Concurrent operations on the list may lead to list corruption and trigger a kernel crash as follows: [417.290971] kernel BUG at lib/list_debug.c:62! [417.290983] invalid opcode: 0000 [#1] PREEMPT SMP NOPTI [417.290992] CPU: 10 PID: 168 Comm: kworker/10:1 Kdump: loaded Not tainted 6.19.0-rc5 #1 [417.291003] Hardware name: QEMU Standard PC (Q35 + ICH9, 2009), BIOS 1.16.3-debian-1.16.3-2 04/01/2014 [417.291007] Workqueue: events bpf_prog_free_deferred [417.291021] RIP: 0010:__list_del_entry_valid_or_report+0xa7/0xc0 [417.291034] Code: a8 ff 0f 0b 48 89 fe 48 89 ca 48 c7 c7 48 a1 eb ae e8 ed fb a8 ff 0f 0b 48 89 fe 48 89 c2 48 c7 c7 80 a1 eb ae e8 d9 fb a8 ff <0f> 0b 48 89 d1 48 c7 c7 d0 a1 eb ae 48 89 f2 48 89 c6 e8 c2 fb a8 [417.291040] RSP: 0018:ffffb16a40807df8 EFLAGS: 00010246 [417.291046] RAX: 000000000000006d RBX: ffff8e589866f500 RCX: 0000000000000000 [417.291051] RDX: 0000000000000000 RSI: ffff8e59f7b23180 RDI: ffff8e59f7b23180 [417.291055] RBP: ffffb16a412c9000 R08: 0000000000000000 R09: 0000000000000003 [417.291059] R10: ffffb16a40807c80 R11: ffffffffaf9edce8 R12: ffff8e594427ac20 [417.291063] R13: ffff8e59f7b44780 R14: ffff8e58800b7a05 R15: 0000000000000000 [417.291074] FS: 0000000000000000(0000) GS:ffff8e59f7b00000(0000) knlGS:0000000000000000 [417.291079] CS: 0010 DS: 0000 ES: 0000 CR0: 0000000080050033 [417.291083] CR2: 00007fc4083efe08 CR3: 00000001c3626006 CR4: 0000000000770ee0 [417.291088] PKRU: 55555554 [417.291091] Call Trace: [417.291096] [417.291103] nsim_bpf_destroy_prog+0x31/0x80 [netdevsim] [417.291154] __bpf_prog_offload_destroy+0x2a/0x80 [417.291163] bpf_prog_dev_bound_destroy+0x6f/0xb0 [417.291171] bpf_prog_free_deferred+0x18e/0x1a0 [417.291178] process_one_work+0x18a/0x3a0 [417.291188] worker_thread+0x27b/0x3a0 [417.291197] ? __pfx_worker_thread+0x10/0x10 [417.291207] kthread+0xe5/0x120 [417.291214] ? __pfx_kthread+0x10/0x10 [417.291221] ret_from_fork+0x31/0x50 [417.291230] ? __pfx_kthread+0x10/0x10 [417.291236] ret_from_fork_asm+0x1a/0x30 [417.291246] Add a mutex lock, to prevent simultaneous addition and deletion operations on the list.
CVE-2026-23167	MEDIUM	4.7	CVSS:3.1/AV:L/AC:H/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: nfc: nci: Fix race between rfkill and nci_unregister_device(). syzbot reported the splat below [0] without a repro. It indicates that struct nci_dev.cmd_wq had been destroyed before nci_close_device() was called via rfkill.

CVE	Severity	CVSS	CVSS Vector	Component	Description
					nci_dev.cmd_wq is only destroyed in nci_unregister_device(), which (I think) was called from virtual_ncidev_close() when syzbot close()d an fd of virtual_ncidev. The problem is that nci_unregister_device() destroys nci_dev.cmd_wq first and then calls nfc_unregister_device(), which removes the device from rfkill by rfkill_unregister(). So, the device is still visible via rfkill even after nci_dev.cmd_wq is destroyed. Let's unregister the device from rfkill first in nci_unregister_device(). Note that we cannot call nfc_unregister_device() before nci_close_device() because 1) nfc_unregister_device() calls device_del() which frees all memory allocated by devm_kzalloc() and linked to ndev->conn_info_list 2) nci_rx_work() could try to queue nci_conn_info to ndev->conn_info_list which could be leaked Thus, nfc_unregister_device() is split into two functions so we can remove rfkill interfaces only before nci_close_device(). [0]: DEBUG_LOCKS_WARN_ON(1) WARNING: kernel/locking/lockdep.c:238 at hlock_class kernel/locking/lockdep.c:238 [inline], CPU#0: syz.0.8675/6349 WARNING: kernel/locking/lockdep.c:238 at check_wait_context kernel/locking/lockdep.c:4854 [inline], CPU#0: syz.0.8675/6349 WARNING: kernel/locking/lockdep.c:238 at __lock_acquire+0x39d/0x2cf0 kernel/locking/lockdep.c:5187, CPU#0: syz.0.8675/6349 Modules linked in: CPU: 0 UID: 0 PID: 6349 Comm: syz.0.8675 Not tainted syzkaller #0 PREEMPT(full) Hardware name: Google Google Compute Engine/Google Compute Engine, BIOS Google 01/13/2026 RIP: 0010:hlock_class kernel/locking/lockdep.c:238 [inline] RIP: 0010:check_wait_context kernel/locking/lockdep.c:4854 [inline] RIP: 0010:__lock_acquire+0x3a4/0x2cf0 kernel/locking/lockdep.c:5187 Code: 18 00 4c 8b 74 24 08 75 27 90 e8 17 f2 fc 02 85 c0 74 1c 83 3d 50 e0 4e 0e 00 75 13 48 8d 3d 43 f7 51 0e 48 c7 c6 8b 3a de 8d <67> 48 0f b9 3a 90 31 c0 0f b6 98 c4 00 00 00 41 8b 45 20 25 ff 1f RSP: 0018:ffffc9000c767680 EFLAGS: 00010046 RAX: 0000000000000001 RBX: 0000000000040000 RCX: 0000000000080000 RDX: ffffc90013080000 RSI: ffffffff8dde3a8b RDI: ffffffff8ff24ca0 RBP: 0000000000000003 R08: ffffffff8fef35a3 R09: 1ffffffff1fde6b4 R10: dffffc0000000000 R11: fffffbfff1fde6b5 R12: 00000000000012a2 R13: ffff888030338ba8 R14: ffff888030338000 R15: ffff888030338b30 FS: 00007fa5995f66c0(0000) GS:ffff8881256f8000(0000) knlGS:0000000000000000 CS: 0010 DS: 0000 ES: 0000 CR0: 0000000080050033 CR2: 00007f7e72f842d0 CR3: 00000000485a0000 CR4: 00000000003526f0 Call Trace: lock_acquire+0x106/0x330 kernel/locking/lockdep.c:5868 touch_wq_lockdep_map+0xcb/0x180 kernel/workqueue.c:3940 __flush_workqueue+0x14b/0x14f0 kernel/workqueue.c:3982 nci_close_device+0x302/0x630 net/nfc/nci/core.c:567 nci_dev_down+0x3b/0x50 net/nfc/nci/core.c:639 nfc_dev_down+0x152/0x290 net/nfc/core.c:161 nfc_rfkill_set_block+0x2d/0x100 net/nfc/core.c:179 rfkill_set_block+0x1d2/0x440 net/rfkill/core.c:346 rfkill_fop_write+0x461/0x5a0 net/rfkill/core.c:1301 vfs_write+0x29a/0xb90 fs/read_write.c:684 ksys_write+0x150/0x270 fs/read_write.c:738 do_syscall_x64 arch/x86/entry/syscall_64.c:63 [inline] do_syscall_64+0xe2/0xf80 arch/x86/entry/syscall_64.c:94 entry_SYSCALL_64_after_hwframe+0x77/0x7f RIP: 0033:0x7fa59b39acb9 Code: ff c3 66 2e 0f 1f 84 00 00 00 00 00 0f 1f 44 00 00 48 89 f8 48 89 f7 48 89 d6 48 89 ca 4d 89 c2 4d 89 c8 4c 8b 4c 24 08 0f 05 <48> 3d 01 f0 ff ff 73 01 c3 48 c7 c1 e8 ff ff ff f7 d8 64 89 01 48 RSP:

CVE	Severity	CVSS	CVSS Vector	Component	Description
					002b:00007fa5995f6028 EFLAGS: 00000246 ORIG_RAX: 0000000000000001 RAX: ffffffffffffffda RBX: 00007fa59b615fa0 RCX: 00007fa59b39acb9 RDX: 0000000000000008 RSI: 0000200000000080 RDI: 0000000000000007 RBP: 00007fa59b408bf7 R08: ---truncated---
CVE-2025-39673	MEDIUM	4.7	CVSS:3.1/AV:L/AC:H/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ppp: fix race conditions in ppp_fill_forward_path ppp_fill_forward_path() has two race conditions: 1. The ppp->channels list can change between list_empty() and list_first_entry(), as ppp_lock() is not held. If the only channel is deleted in ppp_disconnect_channel(), list_first_entry() may access an empty head or a freed entry, and trigger a panic. 2. pch->chan can be NULL. When ppp_unregister_channel() is called, pch->chan is set to NULL before pch is removed from ppp->channels. Fix these by using a lockless RCU approach: - Use list_first_or_null_rcu() to safely test and access the first list entry. - Convert list modifications on ppp->channels to their RCU variants and add synchronize_net() after removal. - Check for a NULL pch->chan before dereferencing it.
CVE-2026-23071	MEDIUM	4.7	CVSS:3.1/AV:L/AC:H/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: regmap: Fix race condition in hwspinlock irqsave routine Previously, the address of the shared member '&map->spinlock_flags' was passed directly to 'hwspin_lock_timeout_irqsave'. This creates a race condition where multiple contexts contending for the lock could overwrite the shared flags variable, potentially corrupting the state for the current lock owner. Fix this by using a local stack variable 'flags' to store the IRQ state temporarily.
CVE-2026-23118	MEDIUM	4.7	CVSS:3.1/AV:L/AC:H/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: rxrpc: Fix data-race warning and potential load/store tearing Fix the following: BUG: KCSAN: data-race in rxrpc_peer_keepalive_worker / rxrpc_send_data_packet which is reporting an issue with the reads and writes to ->last_tx_at in: conn->peer->last_tx_at = ktime_get_seconds(); and: keepalive_at = peer->last_tx_at + RXRPC_KEEPALIVE_TIME; The lockless accesses to these to values aren't actually a problem as the read only needs an approximate time of last transmission for the purposes of deciding whether or not the transmission of a keepalive packet is warranted yet. Also, as ->last_tx_at is a 64-bit value, tearing can occur on a 32-bit arch. Fix both of these by switching to an unsigned int for ->last_tx_at and only storing the LSW of the time64_t. It can then be reconstructed at need provided no more than 68 years has elapsed since the last transmission.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38524	MEDIUM	4.7	CVSS:3.1/AV:L/AC:H/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: rxrpc: Fix recv-recv race of completed call If a call receives an event (such as incoming data), the call gets placed on the socket's queue and a thread in recvmsg can be awakened to go and process it. Once the thread has picked up the call off of the queue, further events will cause it to be requeued, and once the socket lock is dropped (recvmsg uses call->user_mutex to allow the socket to be used in parallel), a second thread can come in and its recvmsg can pop the call off the socket queue again. In such a case, the first thread will be receiving stuff from the call and the second thread will be blocked on call->user_mutex. The first thread can, at this point, process both the event that it picked call for and the event that the second thread picked the call for and may see the call terminate - in which case the call will be "released", decoupling the call from the user call ID assigned to it (RXRPC_USER_CALL_ID in the control message). The first thread will return okay, but then the second thread will wake up holding the user_mutex and, if it sees that the call has been released by the first thread, it will BUG thusly: kernel BUG at net/rxrpc/recvmsg.c:474! Fix this by just dequeuing the call and ignoring it if it is seen to be already released. We can't tell userspace about it anyway as the user call ID has become stale.
CVE-2025-39726	MEDIUM	4.7	CVSS:3.1/AV:L/AC:H/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: s390/ism: fix concurrency management in ism_cmd() The s390x ISM device data sheet clearly states that only one request-response sequence is allowable per ISM function at any point in time. Unfortunately as of today the s390/ism driver in Linux does not honor that requirement. This patch aims to rectify that. This problem was discovered based on Aliaksei's bug report which states that for certain workloads the ISM functions end up entering error state (with PEC 2 as seen from the logs) after a while and as a consequence connections handled by the respective function break, and for future connection requests the ISM device is not considered -- given it is in a dysfunctional state. During further debugging PEC 3A was observed as well. A kernel message like [1211.244319] zpci: 061a:00:00.0: Event 0x2 reports an error for PCI function 0x61a is a reliable indicator of the stated function entering error state with PEC 2. Let me also point out that a kernel message like [1211.244325] zpci: 061a:00:00.0: The ism driver bound to the device does not support error recovery is a reliable indicator that the ISM function won't be auto-recovered because the ISM driver currently lacks support for it. On a technical level, without this synchronization, commands (inputs to the FW) may be partially or fully overwritten (corrupted) by another CPU trying to issue commands on the same function. There is hard evidence that this can lead to DMB token values being used as DMB IOVAs, leading to PEC 2 PCI events indicating invalid DMA. But this is only one of the failure modes imaginable. In theory even completely losing one command and executing another one twice and then trying to interpret the outputs as if the command we intended to execute was actually executed and not the other one is also possible. Frankly, I don't feel confident about providing an exhaustive list of possible consequences.
CVE-2025-38234	MEDIUM	4.7	CVSS:3.1/AV:L/AC:H/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: sched/rt: Fix race in push_rt_task Overview ======== When a CPU chooses to call push_rt_task and picks a task to push to another CPU's runqueue then it will call find_lock_lowest_rq method which would take a double lock on both CPUs' runqueues. If one of the locks aren't readily available, it may lead to dropping the current runqueue lock and reacquiring both the locks at once. During this

CVE	Severity	CVSS	CVSS Vector	Component	Description
					window it is possible that the task is already migrated and is running on some other CPU. These cases are already handled. However, if the task is migrated and has already been executed and another CPU is now trying to wake it up (ttwu) such that it is queued again on the runqeue (on_rq is 1) and also if the task was run by the same CPU, then the current checks will pass even though the task was migrated out and is no longer in the pushable tasks list. Crashes ======= This bug resulted in quite a few flavors of crashes triggering kernel panics with various crash signatures such as assert failures, page faults, null pointer dereferences, and queue corruption errors all coming from scheduler itself. Some of the crashes: -> kernel BUG at kernel/sched/rt.c:1616! BUG_ON(idx >= MAX_RT_PRIO) Call Trace: ? __die_body+0x1a/0x60 ? die+0x2a/0x50 ? do_trap+0x85/0x100 ? pick_next_task_rt+0x6e/0x1d0 ? do_error_trap+0x64/0xa0 ? pick_next_task_rt+0x6e/0x1d0 ? exc_invalid_op+0x4c/0x60 ? pick_next_task_rt+0x6e/0x1d0 ? asm_exc_invalid_op+0x12/0x20 ? pick_next_task_rt+0x6e/0x1d0 __schedule+0x5cb/0x790 ? update_ts_time_stats+0x55/0x70 schedule_idle+0x1e/0x40 do_idle+0x15e/0x200 cpu_startup_entry+0x19/0x20 start_secondary+0x117/0x160 secondary_startup_64_no_verify+0xb0/0xbb -> BUG: kernel NULL pointer dereference, address: 00000000000000c0 Call Trace: ? __die_body+0x1a/0x60 ? no_context+0x183/0x350 ? __warn+0x8a/0xe0 ? exc_page_fault+0x3d6/0x520 ? asm_exc_page_fault+0x1e/0x30 ? pick_next_task_rt+0xb5/0x1d0 ? pick_next_task_rt+0x8c/0x1d0 __schedule+0x583/0x7e0 ? update_ts_time_stats+0x55/0x70 schedule_idle+0x1e/0x40 do_idle+0x15e/0x200 cpu_startup_entry+0x19/0x20 start_secondary+0x117/0x160 secondary_startup_64_no_verify+0xb0/0xbb -> BUG: unable to handle page fault for address: ffff9464daea5900 kernel BUG at kernel/sched/rt.c:1861! BUG_ON(rq->cpu != task_cpu(p)) -> kernel BUG at kernel/sched/rt.c:1055! BUG_ON(!rq->nr_running) Call Trace: ? __die_body+0x1a/0x60 ? die+0x2a/0x50 ? do_trap+0x85/0x100 ? dequeue_top_rt_rq+0xa2/0xb0 ? do_error_trap+0x64/0xa0 ? dequeue_top_rt_rq+0xa2/0xb0 ? exc_invalid_op+0x4c/0x60 ? dequeue_top_rt_rq+0xa2/0xb0 ? asm_exc_invalid_op+0x12/0x20 ? dequeue_top_rt_rq+0xa2/0xb0 dequeue_rt_entity+0x1f/0x70 dequeue_task_rt+0x2d/0x70 __schedule+0x1a8/0x7e0 ? blk_finish_plug+0x25/0x40 schedule+0x3c/0xb0 futex_wait_queue_me+0xb6/0x120 futex_wait+0xd9/0x240 do_futex+0x344/0xa90 ? get_mm_exe_file+0x30/0x60 ? audit_exe_compare+0x58/0x70 ? audit_filter_rules.constprop.26+0x65e/0x1220 __x64_sys_futex+0x148/0x1f0 do_syscall_64+0x30/0x80 entry_SYSCALL_64_after_hwframe+0x62/0xc7 -> BUG: unable to handle page fault for address: ffff8cf3608bc2c0 Call Trace: ? __die_body+0x1a/0x60 ? no_context+0x183/0x350 ? spurious_kernel_fault+0x171/0x1c0 ? exc_page_fault+0x3b6/0x520 ? plist_check_list+0x15/0x40 ? plist_check_list+0x2e/0x40 ? asm_exc_page_fault+0x1e/0x30 ? _cond_resched+0x15/0x30 ? futex_wait_queue_me+0xc8/0x120 ? futex_wait+0xd9/0x240 ? try_to_wake_up+0x1b8/0x490 ?

CVE	Severity	CVSS	CVSS Vector	Component	Description
					futex_wake+0x78/0x160 ? do_futex+0xcd/0xa90 ? plist_check_list+0x15/0x40 ? plist_check_list+0x2e/0x40 ? plist_del+0x6a/0xd0 ? plist_check_list+0x15/0x40 ? plist_check_list+0x2e/0x40 ? dequeue_pushable_task+0x20/0x70 ? __schedule+0x382/0x7e0 ? asm_sysvec_reschedule_i ---truncated---
CVE-2026-23110	MEDIUM	4.7	CVSS:3.1/AV:L/AC:H/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: scsi: core: Wake up the error handler when final completions race against each other The fragile ordering between marking commands completed or failed so that the error handler only wakes when the last running command completes or times out has race conditions. These race conditions can cause the SCSI layer to fail to wake the error handler, leaving I/O through the SCSI host stuck as the error state cannot advance. First, there is an memory ordering issue within scsi_dec_host_busy(). The write which clears SCMD_STATE_INFLIGHT may be reordered with reads counting in scsi_host_busy(). While the local CPU will see its own write, reordering can allow other CPUs in scsi_dec_host_busy() or scsi_eh_inc_host_failed() to see a raised busy count, causing no CPU to see a host busy equal to the host_failed count. This race condition can be prevented with a memory barrier on the error path to force the write to be visible before counting host busy commands. Second, there is a general ordering issue with scsi_eh_inc_host_failed(). By counting busy commands before incrementing host_failed, it can race with a final command in scsi_dec_host_busy(), such that scsi_dec_host_busy() does not see host_failed incremented but scsi_eh_inc_host_failed() counts busy commands before SCMD_STATE_INFLIGHT is cleared by scsi_dec_host_busy(), resulting in neither waking the error handler task. This needs the call to scsi_host_busy() to be moved after host_failed is incremented to close the race condition.
CVE-2025-39825	MEDIUM	4.7	CVSS:3.1/AV:L/AC:H/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: smb: client: fix race with concurrent opens in rename(2) Besides sending the rename request to the server, the rename process also involves closing any deferred close, waiting for outstanding I/O to complete as well as marking all existing open handles as deleted to prevent them from deferring closes, which increases the race window for potential concurrent opens on the target file. Fix this by unhashing the dentry in advance to prevent any concurrent opens on the target.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-68214	MEDIUM	4.7	CVSS:3.1/AV:L/AC:H/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: timers: Fix NULL function pointer race in timer_shutdown_sync() There is a race condition between timer_shutdown_sync() and timer expiration that can lead to hitting a WARN_ON in expire_timers(). The issue occurs when timer_shutdown_sync() clears the timer function to NULL while the timer is still running on another CPU. The race scenario looks like this: CPU0 CPU1 lock_timer_base() expire_timers() base->running_timer = timer; unlock_timer_base() [call_timer_fn enter] mod_timer() ... timer_shutdown_sync() lock_timer_base() // For now, will not detach the timer but only clear its function to NULL if (base->running_timer != timer) ret = detach_if_pending(timer, base, true); if (shutdown) timer->function = NULL; unlock_timer_base() [call_timer_fn exit] lock_timer_base() base->running_timer = NULL; unlock_timer_base() ... // Now timer is pending while its function set to NULL. // next timer trigger expire_timers() WARN_ON_ONCE(!fn) // hit ... lock_timer_base() // Now timer will detach if (base->running_timer != timer) ret = detach_if_pending(timer, base, true); if (shutdown) timer->function = NULL; unlock_timer_base() The problem is that timer_shutdown_sync() clears the timer function regardless of whether the timer is currently running. This can leave a pending timer with a NULL function pointer, which triggers the WARN_ON_ONCE(!fn) check in expire_timers(). Fix this by only clearing the timer function when actually detaching the timer. If the timer is running, leave the function pointer intact, which is safe because the timer will be properly detached when it finishes running.
CVE-2024-57974	MEDIUM	4.7	CVSS:3.1/AV:L/AC:H/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: udp: Deal with race between UDP socket address change and rehash If a UDP socket changes its local address while it's receiving datagrams, as a result of connect(), there is a period during which a lookup operation might fail to find it, after the address is changed but before the secondary hash (port and address) and the four-tuple hash (local and remote ports and addresses) are updated. Secondary hash chains were introduced by commit 30fff9231fad ("udp: bind() optimisation") and, as a result, a rehash operation became needed to make a bound socket reachable again after a connect(). This operation was introduced by commit 719f835853a9 ("udp: add rehash on connect()") which isn't however a complete fix the socket will be found once the rehashing completes, but not while it's pending. This is noticeable with a socat(1) server in UDP4-LISTEN mode, and a client sending datagrams to it. After the server receives the first datagram (cf. _xioopen_ipdgram_listen()), it issues a connect() to the address of the sender, in order to set up a directed flow. Now, if the client, running on a different CPU thread, happens to send a (subsequent) datagram while the server's socket changes its address, but is not rehashed yet, this will result in a failed lookup and a port unreachable error delivered to the client, as apparent from the following reproducer: LEN=$(($(cat /proc/sys/net/core/wmem_default) / 4)) dd if=/dev/urandom bs=1 count=${LEN} of=tmp.in while :; do taskset -c 1 socat UDP4-LISTEN:1337,null-eof OPEN:tmp.out,create,trunc & sleep 0.1 || sleep 1 taskset -c 2 socat OPEN:tmp.in UDP4:localhost:1337,shut-null wait done where the client will eventually get ECONNREFUSED on a write() (typically the second or third one of a given iteration): 2024/11/13 21:28:23 socat[46901] E write(6, 0x556db2e3c000, 8192): Connection refused This issue was first observed as a seldom failure in Podman's tests checking UDP functionality while using pasta(1) to connect the

CVE	Severity	CVSS	CVSS Vector	Component	Description
					container's network namespace, which leads us to a reproducer with the lookup error resulting in an ICMP packet on a tap device: LOCAL_ADDR="$(ip -j -4 addr show\|jq -rM '.[] \| .addr_info[0] \| select(.scope == "global").local')" while :; do ./pasta --config-net -p pasta.pcap -u 1337 socat UDP4-LISTEN:1337,null-eof OPEN:tmp.out,create,trunc & sleep 0.2 \|\| sleep 1 socat OPEN:tmp.in UDP4:${LOCAL_ADDR}:1337,shut-null wait cmp tmp.in tmp.out done Once this fails: tmp.in tmp.out differ: char 8193, line 29 we can finally have a look at what's going on: $ tshark -r pasta.pcap 1 0.000000 :: ? ff02::16 ICMPv6 110 Multicast Listener Report Message v2 2 0.168690 88.198.0.161 ? 88.198.0.164 UDP 8234 60260 ? 1337 Len=8192 3 0.168767 88.198.0.161 ? 88.198.0.164 UDP 8234 60260 ? 1337 Len=8192 4 0.168806 88.198.0.161 ? 88.198.0.164 UDP 8234 60260 ? 1337 Len=8192 5 0.168827 c6:47:05:8d:dc:04 ? Broadcast ARP 42 Who has 88.198.0.161? Tell 88.198.0.164 6 0.168851 9a:55:9a:55:9a:55 ? c6:47:05:8d:dc:04 ARP 42 88.198.0.161 is at 9a:55:9a:55:9a:55 7 0.168875 88.198.0.161 ? 88.198.0.164 UDP 8234 60260 ? 1337 Len=8192 8 0.168896 88.198.0.164 ? 88.198.0.161 ICMP 590 Destination unreachable (Port unreachable) 9 0.168926 88.198.0.161 ? 88.198.0.164 UDP 8234 60260 ? 1337 Len=8192 10 0.168959 88.198.0.161 ? 88.198.0.164 UDP 8234 60260 ? 1337 Len=8192 11 0.168989 88.198.0.161 ? 88.198.0.164 UDP 4138 60260 ? 1337 Len=4096 12 0.169010 88.198.0.161 ? 88.198.0.164 UDP 42 60260 ? 1337 Len=0 On the third datagram received, the network namespace of the container initiates an ARP lookup to deliver the ICMP message. In another variant of this reproducer, starting the client with: strace -f pasta --config-net -u 1337 socat UDP4-LISTEN:1337,null-eof OPEN:tmp.out,create,tru ---truncated---
CVE-2025-38448	MEDIUM	4.7	CVSS:3.1/AV:L/AC:H/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: usb: gadget: u_serial: Fix race condition in TTY wakeup A race condition occurs when gs_start_io() calls either gs_start_rx() or gs_start_tx(), as those functions briefly drop the port_lock for usb_ep_queue(). This allows gs_close() and gserial_disconnect() to clear port.tty and port_usb, respectively. Use the null-safe TTY Port helper function to wake up TTY. Example CPU1: CPU2: gserial_connect() // lock gs_close() // await lock gs_start_rx() // unlock usb_ep_queue() gs_close() // lock, reset port.tty and unlock gs_start_rx() // lock tty_wakeup() // NPE
CVE-2025-37920	MEDIUM	4.7	CVSS:3.1/AV:L/AC:H/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: xsk: Fix race condition in AF_XDP generic RX path Move rx_lock from xsk_socket to xsk_buff_pool. Fix synchronization for shared umem mode in generic RX path where multiple sockets share single xsk_buff_pool. RX queue is exclusive to xsk_socket, while FILL queue can be shared between multiple sockets. This could result in race condition where two CPU cores access RX path of two different sockets sharing the same umem. Protect both queues by acquiring spinlock in shared xsk_buff_pool. Lock contention may be minimized in the future by some per-thread FQ buffering. It's safe and necessary to move spin_lock_bh(rx_lock) after xsk_rcv_check(): * xs->pool and spinlock_init is synchronized by xsk_bind() -> xsk_is_bound() memory barriers. * xsk_rcv_check() may return true at the moment of xsk_release() or xsk_unbind_dev(), however this will not cause any data races or race conditions. xsk_unbind_dev() removes xdp socket from all maps and waits for completion of all outstanding rx operations. Packets in RX path will either complete safely or drop.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2024-24864	MEDIUM	4.7	CVSS:3.1/AV:L/AC:H/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	A race condition was found in the Linux kernel's media/dvb-core in dvbdmx_write() function. This can result in a null pointer dereference issue, possibly leading to a kernel panic or denial of service issue.
CVE-2025-38632	MEDIUM	4.7	CVSS:3.1/AV:L/AC:H/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: pinmux: fix race causing mux_owner NULL with active mux_usecount commit 5a3e85c3c397 ("pinmux: Use sequential access to access desc->pinmux data") tried to address the issue when two client of the same gpio calls pinctrl_select_state() for the same functionality, was resulting in NULL pointer issue while accessing desc->mux_owner. However, issue was not completely fixed due to the way it was handled and it can still result in the same NULL pointer. The issue occurs due to the following interleaving: cpu0 (process A) cpu1 (process B) pin_request() { pin_free() { mutex_lock() desc->mux_usecount--; //becomes 0 .. mutex_unlock() mutex_lock(desc->mux) desc->mux_usecount++; // becomes 1 desc->mux_owner = owner; mutex_unlock(desc->mux) mutex_lock(desc->mux) desc->mux_owner = NULL; mutex_unlock(desc->mux) This sequence leads to a state where the pin appears to be in use (`mux_usecount == 1`) but has no owner (`mux_owner == NULL`), which can cause NULL pointer on next pin_request on the same pin. Ensure that updates to mux_usecount and mux_owner are performed atomically under the same lock. Only clear mux_owner when mux_usecount reaches zero and no new owner has been assigned.
CVE-2026-23212	MEDIUM	4.7	CVSS:3.1/AV:L/AC:H/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: bonding: annotate data-races around slave->last_rx slave->last_rx and slave->target_last_arp_rx[...] can be read and written locklessly. Add READ_ONCE() and WRITE_ONCE() annotations. syzbot reported: BUG: KCSAN: data-race in bond_rcv_validate / bond_rcv_validate write to 0xffff888149f0d428 of 8 bytes by interrupt on cpu 1: bond_rcv_validate+0x202/0x7a0 drivers/net/bonding/bond_main.c:3335 bond_handle_frame+0xde/0x5e0 drivers/net/bonding/bond_main.c:1533 __netif_receive_skb_core+0x5b1/0x1950 net/core/dev.c:6039 __netif_receive_skb_one_core net/core/dev.c:6150 [inline] __netif_receive_skb+0x59/0x270 net/core/dev.c:6265 netif_receive_skb_internal net/core/dev.c:6351 [inline] netif_receive_skb+0x4b/0x2d0 net/core/dev.c:6410 ... write to 0xffff888149f0d428 of 8 bytes by interrupt on cpu 0: bond_rcv_validate+0x202/0x7a0 drivers/net/bonding/bond_main.c:3335 bond_handle_frame+0xde/0x5e0 drivers/net/bonding/bond_main.c:1533 __netif_receive_skb_core+0x5b1/0x1950 net/core/dev.c:6039 __netif_receive_skb_one_core net/core/dev.c:6150 [inline] __netif_receive_skb+0x59/0x270 net/core/dev.c:6265 netif_receive_skb_internal net/core/dev.c:6351 [inline] netif_receive_skb+0x4b/0x2d0 net/core/dev.c:6410 br_netif_receive_skb net/bridge/br_input.c:30 [inline] NF_HOOK include/linux/netfilter.h:318 [inline] ... value changed: 0x0000000100005365 -> 0x0000000100005366

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-71111	MEDIUM	4.7	CVSS:3.1/AV:L/AC:H/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: hwmon: (w83791d) Convert macros to functions to avoid TOCTOU The macro FAN_FROM_REG evaluates its arguments multiple times. When used in lockless contexts involving shared driver data, this leads to Time-of-Check to Time-of-Use (TOCTOU) race conditions, potentially causing divide-by-zero errors. Convert the macro to a static function. This guarantees that arguments are evaluated only once (pass-by-value), preventing the race conditions. Additionally, in store_fan_div, move the calculation of the minimum limit inside the update lock. This ensures that the read-modify-write sequence operates on consistent data. Adhere to the principle of minimal changes by only converting macros that evaluate arguments multiple times and are used in lockless contexts.
CVE-2025-39713	MEDIUM	4.7	CVSS:3.1/AV:L/AC:H/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: media: rainshadow-cec: fix TOCTOU race condition in rain_interrupt() In the interrupt handler rain_interrupt(), the buffer full check on rain->buf_len is performed before acquiring rain->buf_lock. This creates a Time-of-Check to Time-of-Use (TOCTOU) race condition, as rain->buf_len is concurrently accessed and modified in the work handler rain_irq_work_handler() under the same lock. Multiple interrupt invocations can race, with each reading buf_len before it becomes full and then proceeding. This can lead to both interrupts attempting to write to the buffer, incrementing buf_len beyond its capacity (DATA_SIZE) and causing a buffer overflow. Fix this bug by moving the spin_lock() to before the buffer full check. This ensures that the check and the subsequent buffer modification are performed atomically, preventing the race condition. An corresponding spin_unlock() is added to the overflow path to correctly release the lock. This possible bug was found by an experimental static analysis tool developed by our team.
CVE-2025-38461	MEDIUM	4.7	CVSS:3.1/AV:L/AC:H/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: vsock: Fix transport_* TOCTOU Transport assignment may race with module unload. Protect new_transport from becoming a stale pointer. This also takes care of an insecure call in vsock_use_local_transport(); add a lockdep assert. BUG: unable to handle page fault for address: ffffffbfff8056000 Oops: Oops: 0000 [#1] SMP KASAN RIP: 0010:vsock_assign_transport+0x366/0x600 Call Trace: vsock_connect+0x59c/0xc40 __sys_connect+0xe8/0x100 __x64_sys_connect+0x6e/0xc0 do_syscall_64+0x92/0x1c0 entry_SYSCALL_64_after_hwframe+0x4b/0x53
CVE-2025-38462	MEDIUM	4.7	CVSS:3.1/AV:L/AC:H/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: vsock: Fix transport_{g2h,h2g} TOCTOU vsock_find_cid() and vsock_dev_do_ioctl() may race with module unload. transport_{g2h,h2g} may become NULL after the NULL check. Introduce vsock_transport_local_cid() to protect from a potential null-ptr-deref. KASAN: null-ptr-deref in range [0x0000000000000118-0x000000000000011f] RIP: 0010:vsock_find_cid+0x47/0x90 Call Trace: __vsock_bind+0x4b2/0x720 vsock_bind+0x90/0xe0 __sys_bind+0x14d/0x1e0 __x64_sys_bind+0x6e/0xc0 do_syscall_64+0x92/0x1c0 entry_SYSCALL_64_after_hwframe+0x4b/0x53 KASAN: null-ptr-deref in range [0x0000000000000118-0x000000000000011f] RIP: 0010:vsock_dev_do_ioctl.isra.0+0x58/0xf0 Call Trace: __x64_sys_ioctl+0x12d/0x190 do_syscall_64+0x92/0x1c0 entry_SYSCALL_64_after_hwframe+0x4b/0x53

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2024-49998	MEDIUM	4.7	CVSS:3.1/AV:L/AC:H/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net: dsa: improve shutdown sequence Alexander Sverdlin presents 2 problems during shutdown with the lan9303 driver. One is specific to lan9303 and the other just happens to reproduce there. The first problem is that lan9303 is unique among DSA drivers in that it calls dev_get_drvdata() at "arbitrary runtime" (not probe, not shutdown, not remove): phy_state_machine() -> ... -> dsa_user_phy_read() -> ds->ops->phy_read() -> lan9303_phy_read() -> chip->ops->phy_read() -> lan9303_mdio_phy_read() -> dev_get_drvdata() But we never stop the phy_state_machine(), so it may continue to run after dsa_switch_shutdown(). Our common pattern in all DSA drivers is to set drvdata to NULL to suppress the remove() method that may come afterwards. But in this case it will result in an NPD. The second problem is that the way in which we set dp->conduit->dsa_ptr = NULL; is concurrent with receive packet processing. dsa_switch_rcv() checks once whether dev->dsa_ptr is NULL, but afterwards, rather than continuing to use that non-NULL value, dev->dsa_ptr is dereferenced again and again without NULL checks: dsa_conduit_find_user() and many other places. In between dereferences, there is no locking to ensure that what was valid once continues to be valid. Both problems have the common aspect that closing the conduit interface solves them. In the first case, dev_close(conduit) triggers the NETDEV_GOING_DOWN event in dsa_user_netdevice_event() which closes user ports as well. dsa_port_disable_rt() calls phylink_stop(), which synchronously stops the phylink state machine, and ds->ops->phy_read() will thus no longer call into the driver after this point. In the second case, dev_close(conduit) should do this, as per Documentation/networking/driver.rst: | Quiescence | ---------- | | After the ndo_stop routine has been called, the hardware must | not receive or transmit any data. All in flight packets must | be aborted. If necessary, poll or wait for completion of | any reset commands. So it should be sufficient to ensure that later, when we zeroize conduit->dsa_ptr, there will be no concurrent dsa_switch_rcv() call on this conduit. The addition of the netif_device_detach() function is to ensure that ioctls, rtnetlinks and ethtool requests on the user ports no longer propagate down to the driver - we're no longer prepared to handle them. The race condition actually did not exist when commit 0650bf52b31f ("net: dsa: be compatible with masters which unregister on shutdown") first introduced dsa_switch_shutdown(). It was created later, when we stopped unregistering the user interfaces from a bad spot, and we just replaced that sequence with a racy zeroization of conduit->dsa_ptr (one which doesn't ensure that the interfaces aren't up).
CVE-2025-21825	MEDIUM	4.7	CVSS:3.1/AV:L/AC:H/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: bpf: Cancel the running bpf_timer through kworker for PREEMPT_RT During the update procedure, when overwrite element in a pre-allocated htab, the freeing of old_element is protected by the bucket lock. The reason why the bucket lock is necessary is that the old_element has already been stashed in htab->extra_elems after alloc_htab_elem() returns. If freeing the old_element after the bucket lock is unlocked, the stashed element may be reused by concurrent update procedure and the freeing of old_element will run concurrently with the reuse of the old_element. However, the invocation of check_and_free_fields() may acquire a spin-lock which violates the lockdep rule because its caller has already held a raw-spin-lock (bucket lock). The following warning will be reported when such race happens: BUG: scheduling while

CVE	Severity	CVSS	CVSS Vector	Component	Description
					atomic: test_progs/676/0x00000003 3 locks held by test_progs/676: #0: ffffffff864b0240 (rcu_read_lock_trace){....}-{0:0}, at: bpf_prog_test_run_syscall+0x2c0/0x830 #1: ffff88810e961188 (&htab->lockdep_key){....}-{2:2}, at: htab_map_update_elem+0x306/0x1500 #2: ffff8881f4eac1b8 (&base->softirq_expiry_lock){....}-{2:2}, at: hrtimer_cancel_wait_running+0xe9/0x1b0 Modules linked in: bpf_testmod(O) Preemption disabled at: [] htab_map_update_elem+0x293/0x1500 CPU: 0 UID: 0 PID: 676 Comm: test_progs Tainted: G ... 6.12.0+ #11 Tainted: [W]=WARN, [O]=OOT_MODULE Hardware name: QEMU Standard PC (i440FX + PIIX, 1996)... Call Trace: dump_stack_lvl+0x57/0x70 dump_stack+0x10/0x20 __schedule_bug+0x120/0x170 __schedule+0x300c/0x4800 schedule_rtlock+0x37/0x60 rtlock_slowlock_locked+0x6d9/0x54c0 rt_spin_lock+0x168/0x230 hrtimer_cancel_wait_running+0xe9/0x1b0 hrtimer_cancel+0x24/0x30 bpf_timer_delete_work+0x1d/0x40 bpf_timer_cancel_and_free+0x5e/0x80 bpf_obj_free_fields+0x262/0x4a0 check_and_free_fields+0x1d0/0x280 htab_map_update_elem+0x7fc/0x1500 bpf_prog_9f90bc20768e0cb9_overwrite_cb+0x3f/0x43 bpf_prog_ea601c4649694dbd_overwrite_timer+0x5d/0x7e bpf_prog_test_run_syscall+0x322/0x830 __sys_bpf+0x135d/0x3ca0 __x64_sys_bpf+0x75/0xb0 x64_sys_call+0x1b5/0xa10 do_syscall_64+0x3b/0xc0 entry_SYSCALL_64_after_hwframe+0x4b/0x53 ... It seems feasible to break the reuse and refill of per-cpu extra_elems into two independent parts: reuse the per-cpu extra_elems with bucket lock being held and refill the old_element as per-cpu extra_elems after the bucket lock is unlocked. However, it will make the concurrent overwrite procedures on the same CPU return unexpected -E2BIG error when the map is full. Therefore, the patch fixes the lock problem by breaking the cancelling of bpf_timer into two steps for PREEMPT_RT: 1) use hrtimer_try_to_cancel() and check its return value 2) if the timer is running, use hrtimer_cancel() through a kworker to cancel it again Considering that the current implementation of hrtimer_cancel() will try to acquire a being held softirq_expiry_lock when the current timer is running, these steps above are reasonable. However, it also has downside. When the timer is running, the cancelling of the timer is delayed when releasing the last map uref. The delay is also fixable (e.g., break the cancelling of bpf timer into two parts: one part in locked scope, another one in unlocked scope), it can be revised later if necessary. It is a bit hard to decide the right fix tag. One reason is that the problem depends on PREEMPT_RT which is enabled in v6.12. Considering the softirq_expiry_lock lock exists since v5.4 and bpf_timer is introduced in v5.15, the bpf_timer commit is used in the fixes tag and an extra depends-on tag is added to state the dependency on PREEMPT_RT. Depends-on: v6.12+ with PREEMPT_RT enabled

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2026-23101	MEDIUM	4.7	CVSS:3.1/AV:L/AC:H/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: leds: led-class: Only Add LED to leds_list when it is fully ready Before this change the LED was added to leds_list before led_init_core() gets called adding it the list before led_classdev.set_brightness_work gets initialized. This leaves a window where led_trigger_register() of a LED's default trigger will call led_trigger_set() which calls led_set_brightness() which in turn will end up queueing the *uninitialized* led_classdev.set_brightness_work. This race gets hit by the lenovo-thinkpad-t14s EC driver which registers 2 LEDs with a default trigger provided by snd_ctl_led.ko in quick succession. The first led_classdev_register() causes an async modprobe of snd_ctl_led to run and that async modprobe manages to exactly hit the window where the second LED is on the leds_list without led_init_core() being called for it, resulting in: ------------[cut here]------------ WARNING: CPU: 11 PID: 5608 at kernel/workqueue.c:4234 __flush_work+0x344/0x390 Hardware name: LENOVO 21N2S01F0B/21N2S01F0B, BIOS N42ET93W (2.23) 09/01/2025 ... Call trace: __flush_work+0x344/0x390 (P) flush_work+0x2c/0x50 led_trigger_set+0x1c8/0x340 led_trigger_register+0x17c/0x1c0 led_trigger_register_simple+0x84/0xe8 snd_ctl_led_init+0x40/0xf88 [snd_ctl_led] do_one_initcall+0x5c/0x318 do_init_module+0x9c/0x2b8 load_module+0x7e0/0x998 Close the race window by moving the adding of the LED to leds_list to after the led_init_core() call.
CVE-2024-24859	MEDIUM	4.8	CVSS:3.1/AV:A/AC:H/PR:N/UI:R/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	A race condition was found in the Linux kernel's net/bluetooth in sniff_{min,max}_interval_set() function. This can result in a bluetooth sniffing exception issue, possibly leading denial of service.
CVE-2025-71225	MEDIUM	5.3	CVSS:3.1/AV:L/AC:H/PR:L/UI:N/S:U/C:L/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: md: suspend array while updating raid_disks via sysfs In raid1_reshape(), freeze_array() is called before modifying the r1bio memory pool (conf->r1bio_pool) and conf->raid_disks, and unfreeze_array() is called after the update is completed. However, freeze_array() only waits until nr_sync_pending and (nr_pending - nr_queued) of all buckets reaches zero. When an I/O error occurs, nr_queued is increased and the corresponding r1bio is queued to either retry_list or bio_end_io_list. As a result, freeze_array() may unblock before these r1bios are released. This can lead to a situation where conf->raid_disks and the mempool have already been updated while queued r1bios, allocated with the old raid_disks value, are later released. Consequently, free_r1bio() may access memory out of bounds in put_all_bios() and release r1bios of the wrong size to the new mempool, potentially causing issues with the mempool as well. Since only normal I/O might increase nr_queued while an I/O error occurs, suspending the array avoids this issue. Note: Updating raid_disks via ioctl SET_ARRAY_INFO already suspends the array. Therefore, we suspend the array when updating raid_disks via sysfs to avoid this issue too.
CVE-2022-4543	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:N	linux-rittal 6.6.96+git	A flaw named "EntryBleed" was found in the Linux Kernel Page Table Isolation (KPTI). This issue could allow a local attacker to leak KASLR base via prefetch side-channels based on TLB timing for Intel systems.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-71151	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:N	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: cifs: Fix memory and information leak in smb3_reconfigure() In smb3_reconfigure(), if smb3_sync_session_ctx_passwords() fails, the function returns immediately without freeing and erasing the newly allocated new_password and new_password2. This causes both a memory leak and a potential information leak. Fix this by calling kfree_sensitive() on both password buffers before returning in this error case.
CVE-2025-38207	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:H/A:N	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: mm: fix uprobe pte be overwritten when expanding vma Patch series "Fix uprobe pte be overwritten when expanding vma". This patch (of 4): We encountered a BUG alert triggered by Syzkaller as follows: BUG: Bad rss-counter state mm:00000000b4a60fca type:MM_ANONPAGES val:1 And we can reproduce it with the following steps: 1. register uprobe on file at zero offset 2. mmap the file at zero offset: addr1 = mmap(NULL, 2 * 4096, PROT_NONE, MAP_PRIVATE, fd, 0); 3. mremap part of vma1 to new vma2: addr2 = mremap(addr1, 4096, 2 * 4096, MREMAP_MAYMOVE); 4. mremap back to orig addr1: mremap(addr2, 4096, 4096, MREMAP_MAYMOVE | MREMAP_FIXED, addr1); In step 3, the vma1 range [addr1, addr1 + 4096] will be remap to new vma2 with range [addr2, addr2 + 8192], and remap uprobe anon page from the vma1 to vma2, then unmap the vma1 range [addr1, addr1 + 4096]. In step 4, the vma2 range [addr2, addr2 + 4096] will be remap back to the addr range [addr1, addr1 + 4096]. Since the addr range [addr1 + 4096, addr1 + 8192] still maps the file, it will take vma_merge_new_range to expand the range, and then do uprobe_mmap in vma_complete. Since the merged vma pgoff is also zero offset, it will install uprobe anon page to the merged vma. However, the upcomming move_page_tables step, which use set_pte_at to remap the vma2 uprobe pte to the merged vma, will overwrite the newly uprobe pte in the merged vma, and lead that pte to be orphan. Since the uprobe pte will be remapped to the merged vma, we can remove the unnecessary uprobe_mmap upon merged vma. This problem was first found in linux-6.6.y and also exists in the community syzkaller: https://lore.kernel.org/all/000000000000ada39605a5e71711@google.com/T/
CVE-2024-27017	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:H/A:N	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: netfilter: nft_set_pipapo: walk over current view on netlink dump The generation mask can be updated while netlink dump is in progress. The pipapo set backend walk iterator cannot rely on it to infer what view of the datastructure is to be used. Add notation to specify if user wants to read/update the set. Based on patch from Florian Westphal.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-39763	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ACPI: APEI: send SIGBUS to current task if synchronous memory error not recovered If a synchronous error is detected as a result of user-space process triggering a 2-bit uncorrected error, the CPU will take a synchronous error exception such as Synchronous External Abort (SEA) on Arm64. The kernel will queue a memory_failure() work which poisons the related page, unmaps the page, and then sends a SIGBUS to the process, so that a system wide panic can be avoided. However, no memory_failure() work will be queued when abnormal synchronous errors occur. These errors can include situations like invalid PA, unexpected severity, no memory failure config support, invalid GUID section, etc. In such a case, the user-space process will trigger SEA again. This loop can potentially exceed the platform firmware threshold or even trigger a kernel hard lockup, leading to a system reboot. Fix it by performing a force kill if no memory_failure() work is queued for synchronous errors. [rjw: Changelog edits]
CVE-2025-38105	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ALSA: usb-audio: Kill timer properly at removal The USB-audio MIDI code initializes the timer, but in a rare case, the driver might be freed without the disconnect call. This leaves the timer in an active state while the assigned object is released via snd_usbmidi_free(), which ends up with a kernel warning when the debug configuration is enabled, as spotted by fuzzer. For avoiding the problem, put timer_shutdown_sync() at snd_usbmidi_free(), so that the timer can be killed properly. While we're at it, replace the existing timer_delete_sync() at the disconnect callback with timer_shutdown_sync(), too.
CVE-2025-39752	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ARM: rockchip: fix kernel hang during smp initialization In order to bring up secondary CPUs main CPU write trampoline code to SRAM. The trampoline code is written while secondary CPUs are powered on (at least that true for RK3188 CPU). Sometimes that leads to kernel hang. Probably because secondary CPU execute trampoline code while kernel doesn't expect. The patch moves SRAM initialization step to the point where all secondary CPUs are powered down. That fixes rarely hangs on RK3188: [0.091568] CPU0: thread -1, cpu 0, socket 0, mpidr 80000000 [0.091996] rockchip_smp_prepare_cpus: ncores 4
CVE-2025-39794	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ARM: tegra: Use I/O memcpy to write to IRAM Kasan crashes the kernel trying to check boundaries when using the normal memcpy.
CVE-2025-38099	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: Bluetooth: Disable SCO support if READ_VOICE_SETTING is unsupported/broken A SCO connection without the proper voice_setting can cause the controller to lock up.
CVE-2025-38303	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: Bluetooth: eir: Fix possible crashes on eir_create_adv_data eir_create_adv_data may attempt to add EIR_FLAGS and EIR_TX_POWER without checking if that would fit.
CVE-2024-56591	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: Bluetooth: hci_conn: Use disable_delayed_work_sync This makes use of disable_delayed_work_sync instead cancel_delayed_work_sync as it not only cancel the ongoing work but also disables new submit which is disarable since the object holding the work is about to be freed.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2024-58241	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: Bluetooth: hci_core: Disable works on hci_unregister_dev This make use of disable_work_* on hci_unregister_dev since the hci_dev is about to be freed new submissions are not disarable.
CVE-2025-38495	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: HID: core: ensure the allocated report buffer can contain the reserved report ID When the report ID is not used, the low level transport drivers expect the first byte to be 0. However, currently the allocated buffer not account for that extra byte, meaning that instead of having 8 guaranteed bytes for implement to be working, we only have 7.
CVE-2025-39808	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: HID: hid-ntrig: fix unable to handle page fault in ntrig_report_version() in ntrig_report_version(), hdev parameter passed from hid_probe(). sending descriptor to /dev/uhid can make hdev->dev.parent->parent to null if hdev->dev.parent->parent is null, usb_dev has invalid address(0xffffffffffffff58) that hid_to_usb_dev(hdev) returned when usb_rcvctrlpipe() use usb_dev,it trigger page fault error for address(0xffffffffffffff58) add null check logic to ntrig_report_version() before calling hid_to_usb_dev()
CVE-2025-38507	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: HID: nintendo: avoid bluetooth suspend/resume stalls Ensure we don't stall or panic the kernel when using bluetooth-connected controllers. This was reported as an issue on android devices using kernel 6.6 due to the resume hook which had been added for usb joycons. First, set a new state value to JOYCON_CTLR_STATE_SUSPENDED in a newly-added nintendo_hid_suspend. This makes sure we will not stall out the kernel waiting for input reports during led classdev suspend. The stalls could happen if connectivity is unreliable or lost to the controller prior to suspend. Second, since we lose connectivity during suspend, do not try joycon_init() for bluetooth controllers in the nintendo_hid_resume path. Tested via multiple suspend/resume flows when using the controller both in USB and bluetooth modes.
CVE-2025-38540	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: HID: quirks: Add quirk for 2 Chicony Electronics HP 5MP Cameras The Chicony Electronics HP 5MP Cameras (USB ID 04F2:B824 & 04F2:B82C) report a HID sensor interface that is not actually implemented. Attempting to access this non-functional sensor via iio_info causes system hangs as runtime PM tries to wake up an unresponsive sensor. Add these 2 devices to the HID ignore list since the sensor interface is non-functional by design and should not be exposed to userspace.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38351	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: KVM: x86/hyper-v: Skip non-canonical addresses during PV TLB flush In KVM guests with Hyper-V hypercalls enabled, the hypercalls HVCALL_FLUSH_VIRTUAL_ADDRESS_LIST and HVCALL_FLUSH_VIRTUAL_ADDRESS_LIST_EX allow a guest to request invalidation of portions of a virtual TLB. For this, the hypercall parameter includes a list of GVAs that are supposed to be invalidated. However, when non-canonical GVAs are passed, there is currently no filtering in place and they are eventually passed to checked invocations of INVVPID on Intel / INVLPGA on AMD. While AMD's INVLPGA silently ignores non-canonical addresses (effectively a no-op), Intel's INVVPID explicitly signals VM-Fail and ultimately triggers the WARN_ONCE in invvpid_error(): invvpid failed: ext=0x0 vpid=1 gva=0xaaaaaaaaaaaaa000 WARNING: CPU: 6 PID: 326 at arch/x86/kvm/vmx/vmx.c:482 invvpid_error+0x91/0xa0 [kvm_intel] Modules linked in: kvm_intel kvm 9pnet_virtio irqbypass fuse CPU: 6 UID: 0 PID: 326 Comm: kvm-vm Not tainted 6.15.0 #14 PREEMPT(voluntary) RIP: 0010:invvpid_error+0x91/0xa0 [kvm_intel] Call Trace: vmx_flush_tlb_gva+0x320/0x490 [kvm_intel] kvm_hv_vcpu_flush_tlb+0x24f/0x4f0 [kvm] kvm_arch_vcpu_ioctl_run+0x3013/0x5810 [kvm] Hyper-V documents that invalid GVAs (those that are beyond a partition's GVA space) are to be ignored. While not completely clear whether this ruling also applies to non-canonical GVAs, it is likely fine to make that assumption, and manual testing on Azure confirms "real" Hyper-V interprets the specification in the same way. Skip non-canonical GVAs when processing the list of address to avoid tripping the INVVPID failure. Alternatively, KVM could filter out "bad" GVAs before inserting into the FIFO, but practically speaking the only downside of pushing validation to the final processing is that doing so is suboptimal for the guest, and no well-behaved guest will request TLB flushes for non-canonical addresses.
CVE-2025-38469	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: KVM: x86/xen: Fix cleanup logic in emulation of Xen schedop poll hypercalls kvm_xen_schedop_poll does a kmalloc_array() when a VM polls the host for more than one event channel potr (nr_ports > 1). After the kmalloc_array(), the error paths need to go through the "out" label, but the call to kvm_read_guest_virt() does not. [Adjusted commit message. - Paolo]
CVE-2025-38723	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: LoongArch: BPF: Fix jump offset calculation in tailcall The extra pass of bpf_int_jit_compile() skips JIT context initialization which essentially skips offset calculation leaving out_offset = -1, so the jmp_offset in emit_bpf_tail_call is calculated by "#define jmp_offset (out_offset - (cur_offset))" is a negative number, which is wrong. The final generated assembly are as follow. 54: bgeu $a2, $t1, -8 # 0x0000004c 58: addi.d $a6, $s5, -1 5c: bltz $a6, -16 # 0x0000004c 60: alsl.d $t2, $a2, $a1, 0x3 64: ld.d $t2, $t2, 264 68: beq $t2, $zero, -28 # 0x0000004c Before apply this patch, the follow test case will reveal soft lock issues. cd tools/testing/selftests/bpf/ ./test_progs --allow=tailcalls/tailcall_bpf2bpf_1 dmesg: watchdog: BUG: soft lockup - CPU#2 stuck for 26s! [test_progs:25056]

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-71129	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: LoongArch: BPF: Sign extend kfunc call arguments The kfunc calls are native calls so they should follow LoongArch calling conventions. Sign extend its arguments properly to avoid kernel panic. This is done by adding a new emit_abi_ext() helper. The emit_abi_ext() helper performs extension in place meaning a value already store in the target register (Note: this is different from the existing sign_extend() helper and thus we can't reuse it).

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2024-53089	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: LoongArch: KVM: Mark hrtimer to expire in hard interrupt context Like commit 2c0d278f3293f ("KVM: LAPIC: Mark hrtimer to expire in hard interrupt context") and commit 9090825fa9974 ("KVM: arm/arm64: Let the timer expire in hardirq context on RT"), On PREEMPT_RT enabled kernels unmarked hrtimers are moved into soft interrupt expiry mode by default. Then the timers are canceled from an preempt-notifier which is invoked with disabled preemption which is not allowed on PREEMPT_RT. The timer callback is short so in could be invoked in hard-IRQ context. So let the timer expire on hard-IRQ context even on -RT. This fix a "scheduling while atomic" bug for PREEMPT_RT enabled kernels: BUG: scheduling while atomic: qemu-system-loo/1011/0x00000002 Modules linked in: amdgpu rfkill nft_fib_inet nft_fib_ipv4 nft_fib_ipv6 nft_fib nft_reject_inet nf_reject_ipv4 nf_reject_ipv6 nft_reject nft_ct nft_chain_nat ns CPU: 1 UID: 0 PID: 1011 Comm: qemu-system-loo Tainted: G W 6.12.0-rc2+ #1774 Tainted: [W]=WARN Hardware name: Loongson Loongson-3A5000-7A1000-1w-CRB/Loongson-LS3A5000-7A1000-1w-CRB, BIOS vUDK2018-LoongArch-V2.0.0-prebeta9 10/21/2022 Stack : ffffffffffffffff 0000000000000000 9000000004e3ea38 9000000116744000 90000001167475a0 0000000000000000 90000001167475a8 9000000005644830 90000000058dc000 90000000058dbff8 9000000116747420 0000000000000001 0000000000000001 6a613fc938313980 000000000790c000 90000001001c1140 00000000000003fe 0000000000000001 000000000000000d 0000000000000003 0000000000000030 00000000000003f3 000000000790c000 9000000116747830 90000000057ef000 0000000000000000 9000000005644830 0000000000000004 0000000000000000 90000000057f4b58 0000000000000001 9000000116747868 900000000451b600 9000000005644830 9000000003a13998 0000000010000020 00000000000000b0 0000000000000004 0000000000000000 0000000000071c1d ... Call Trace: [<9000000003a13998>] show_stack+0x38/0x180 [<9000000004e3ea34>] dump_stack_lvl+0x84/0xc0 [<9000000003a71708>] __schedule_bug+0x48/0x60 [<9000000004e45734>] __schedule+0x1114/0x1660 [<9000000004e46040>] schedule_rtlock+0x20/0x60 [<9000000004e4e330>] rtlock_slowlock_locked+0x3f0/0x10a0 [<9000000004e4f038>] rt_spin_lock+0x58/0x80 [<9000000003b02d68>] hrtimer_cancel_wait_running+0x68/0xc0 [<9000000003b02e30>] hrtimer_cancel+0x70/0x80 [] kvm_restore_timer+0x50/0x1a0 [kvm] [] kvm_arch_vcpu_load+0x68/0x2a0 [kvm] [] kvm_sched_in+0x34/0x60 [kvm] [<9000000003a749a0>] finish_task_switch.isra.0+0x140/0x2e0 [<9000000004e44a70>] __schedule+0x450/0x1660 [<9000000004e45cb0>] schedule+0x30/0x180 [] kvm_vcpu_block+0x70/0x120 [kvm] [] kvm_vcpu_halt+0x60/0x3e0 [kvm] [] kvm_handle_gspr+0x3f4/0x4e0 [kvm] [] kvm_handle_exit+0x1c8/0x260 [kvm]

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-21949	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: LoongArch: Set hugetlb mmap base address aligned with pmd size With ltp test case "testcases/bin/hugefork02", there is a dmesg error report message such as: kernel BUG at mm/hugetlb.c:5550! Oops - BUG[#1]: CPU: 0 UID: 0 PID: 1517 Comm: hugefork02 Not tainted 6.14.0-rc2+ #241 Hardware name: QEMU QEMU Virtual Machine, BIOS unknown 2/2/2022 pc 90000000004eaf1c ra 9000000000485538 tp 900000010edbc000 sp 900000010edbf940 a0 900000010edbfb00 a1 9000000108d20280 a2 00007fffe9474000 a3 00007ffff3474000 a4 0000000000000000 a5 0000000000000003 a6 00000000003cadd3 a7 0000000000000000 t0 0000000001ffffff t1 0000000001474000 t2 900000010ecd7900 t3 00007fffe9474000 t4 00007fffe9474000 t5 0000000000000040 t6 900000010edbfb00 t7 0000000000000001 t8 0000000000000005 u0 90000000004849d0 s9 900000010edbfa00 s0 9000000108d20280 s1 00007fffe9474000 s2 0000000002000000 s3 9000000108d20280 s4 9000000002b38b10 s5 900000010edbfb00 s6 00007ffff3474000 s7 0000000000000406 s8 900000010edbfa08 ra: 9000000000485538 unmap_vmas+0x130/0x218 ERA: 90000000004eaf1c __unmap_hugepage_range+0x6f4/0x7d0 PRMD: 00000004 (PPLV0 +PIE -PWE) EUEN: 00000007 (+FPE +SXE +ASXE -BTE) ECFG: 00071c1d (LIE=0,2-4,10-12 VS=7) ESTAT: 000c0000 [BRK] (IS= ECode=12 EsubCode=0) PRID: 0014c010 (Loongson-64bit, Loongson-3A5000) Process hugefork02 (pid: 1517, threadinfo=00000000a670eaf4, task=000000007a95fc64) Call Trace: [<90000000004eaf1c>] __unmap_hugepage_range+0x6f4/0x7d0 [<9000000000485534>] unmap_vmas+0x12c/0x218 [<9000000000494068>] exit_mmap+0xe0/0x308 [<900000000025fdc4>] mmput+0x74/0x180 [<900000000026a284>] do_exit+0x294/0x898 [<900000000026aa30>] do_group_exit+0x30/0x98 [<900000000027bed4>] get_signal+0x83c/0x868 [<90000000002457b4>] arch_do_signal_or_restart+0x54/0xfa0 [<90000000015795e8>] irqentry_exit_to_user_mode+0xb8/0x138 [<90000000002572d0>] tlb_do_page_fault_1+0x114/0x1b4 The problem is that base address allocated from hugetlbfs is not aligned with pmd size. Here add a checking for hugetlbfs and align base address with pmd size. After this patch the test case "testcases/bin/hugefork02" passes to run. This is similar to the commit 7f24cbc9c4d42db8a3c8484d1 ("mm/mmap: teach generic_get_unmapped_area{_topdown} to handle hugetlb mappings").
CVE-2025-39798	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: NFS: Fix the setting of capabilities when automounting a new filesystem Capabilities cannot be inherited when we cross into a new filesystem. They need to be reset to the minimal defaults, and then probed for again.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2024-58006	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: PCI: dwc: ep: Prevent changing BAR size/flags in pci_epc_set_bar() In commit 4284c88fff0e ("PCI: designware-ep: Allow pci_epc_set_bar() update inbound map address") set_bar() was modified to support dynamically changing the backing physical address of a BAR that was already configured. This means that set_bar() can be called twice, without ever calling clear_bar() (as calling clear_bar() would clear the BAR's PCI address assigned by the host). This can only be done if the new BAR size/flags does not differ from the existing BAR configuration. Add these missing checks. If we allow set_bar() to set e.g. a new BAR size that differs from the existing BAR size, the new address translation range will be smaller than the BAR size already determined by the host, which would mean that a read past the new BAR size would pass the iATU untranslated, which could allow the host to read memory not belonging to the new struct pci_epf_bar. While at it, add comments which clarifies the support for dynamically changing the physical address of a BAR. (Which was also missing.)
CVE-2024-57809	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: PCI: imx6: Fix suspend/resume support on i.MX6QDL The suspend/resume functionality is currently broken on the i.MX6QDL platform, as documented in the NXP errata (ERR005723): https://www.nxp.com/docs/en/errata/IMX6DQCE.pdf This patch addresses the issue by sharing most of the suspend/resume sequences used by other i.MX devices, while avoiding modifications to critical registers that disrupt the PCIe functionality. It targets the same problem as the following downstream commit: https://github.com/nxp-imx/linux-imx/commit/4e92355e1f79d225ea842511fcfd42b343b32995 Unlike the downstream commit, this patch also resets the connected PCIe device if possible. Without this reset, certain drivers, such as ath10k or iwlwifi, will crash on resume. The device reset is also done by the driver on other i.MX platforms, making this patch consistent with existing practices. Upon resuming, the kernel will hang and display an error. Here's an example of the error encountered with the ath10k driver: ath10k_pci 0000:01:00.0: Unable to change power state from D3hot to D0, device inaccessible Unhandled fault: imprecise external abort (0x1406) at 0x0106f944 Without this patch, suspend/resume will fail on i.MX6QDL devices if a PCIe device is connected. [kwilczynski: commit log, added tag for stable releases]

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38624	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: PCI: pnv_php: Clean up allocated IRQs on unplug When the root of a nested PCIe bridge configuration is unplugged, the pnv_php driver leaked the allocated IRQ resources for the child bridges' hotplug event notifications, resulting in a panic. Fix this by walking all child buses and deallocating all its IRQ resources before calling pci_hp_remove_devices(). Also modify the lifetime of the workqueue at struct pnv_php_slot::wq so that it is only destroyed in pnv_php_free_slot(), instead of pnv_php_disable_irq(). This is required since pnv_php_disable_irq() will now be called by workers triggered by hot unplug interrupts, so the workqueue needs to stay allocated. The abridged kernel panic that occurs without this patch is as follows: WARNING: CPU: 0 PID: 687 at kernel/irq/msi.c:292 msi_device_data_release+0x6c/0x9c CPU: 0 UID: 0 PID: 687 Comm: bash Not tainted 6.14.0-rc5+ #2 Call Trace: msi_device_data_release+0x34/0x9c (unreliable) release_nodes+0x64/0x13c devres_release_all+0xc0/0x140 device_del+0x2d4/0x46c pci_destroy_dev+0x5c/0x194 pci_hp_remove_devices+0x90/0x128 pci_hp_remove_devices+0x44/0x128 pnv_php_disable_slot+0x54/0xd4 power_write_file+0xf8/0x18c pci_slot_attr_store+0x40/0x5c sysfs_kf_write+0x64/0x78 kernfs_fop_write_iter+0x1b0/0x290 vfs_write+0x3bc/0x50c ksys_write+0x84/0x140 system_call_exception+0x124/0x230 system_call_vectored_common+0x15c/0x2ec [bhelgaas: tidy comments]
CVE-2025-38623	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: PCI: pnv_php: Fix surprise plug detection and recovery The existing PowerNV hotplug code did not handle surprise plug events correctly, leading to a complete failure of the hotplug system after device removal and a required reboot to detect new devices. This comes down to two issues: 1) When a device is surprise removed, often the bridge upstream port will cause a PE freeze on the PHB. If this freeze is not cleared, the MSI interrupts from the bridge hotplug notification logic will not be received by the kernel, stalling all plug events on all slots associated with the PE. 2) When a device is removed from a slot, regardless of surprise or programmatic removal, the associated PHB/PE Is left frozen. If this freeze is not cleared via a fundamental reset, skiboot is unable to clear the freeze and cannot retrain / rescan the slot. This also requires a reboot to clear the freeze and redetect the device in the slot. Issue the appropriate unfreeze and rescan commands on hotplug events, and don't oops on hotplug if pci_bus_to_OF_node() returns NULL. [bhelgaas: tidy comments]

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-21885	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: RDMA/bnxt_re: Fix the page details for the srq created by kernel consumers While using nvme target with use_srq on, below kernel panic is noticed. [549.698111] bnxt_en 0000:41:00.0 enp65s0np0: FEC autoneg off encoding: Clause 91 RS(544,514) [566.393619] Oops: divide error: 0000 [#1] PREEMPT SMP NOPTI .. [566.393799] [566.393807] ? __die_body+0x1a/0x60 [566.393823] ? die+0x38/0x60 [566.393835] ? do_trap+0xe4/0x110 [566.393847] ? bnxt_qplib_alloc_init_hwq+0x1d4/0x580 [bnxt_re] [566.393867] ? bnxt_qplib_alloc_init_hwq+0x1d4/0x580 [bnxt_re] [566.393881] ? do_error_trap+0x7c/0x120 [566.393890] ? bnxt_qplib_alloc_init_hwq+0x1d4/0x580 [bnxt_re] [566.393911] ? exc_divide_error+0x34/0x50 [566.393923] ? bnxt_qplib_alloc_init_hwq+0x1d4/0x580 [bnxt_re] [566.393939] ? asm_exc_divide_error+0x16/0x20 [566.393966] ? bnxt_qplib_alloc_init_hwq+0x1d4/0x580 [bnxt_re] [566.393997] bnxt_qplib_create_srq+0xc9/0x340 [bnxt_re] [566.394040] bnxt_re_create_srq+0x335/0x3b0 [bnxt_re] [566.394057] ? srso_return_thunk+0x5/0x5f [566.394068] ? __init_swait_queue_head+0x4a/0x60 [566.394090] ib_create_srq_user+0xa7/0x150 [ib_core] [566.394147] nvmet_rdma_queue_connect+0x7d0/0xbe0 [nvmet_rdma] [566.394174] ? lock_release+0x22c/0x3f0 [566.394187] ? srso_return_thunk+0x5/0x5f Page size and shift info is set only for the user space SRQs. Set page size and page shift for kernel space SRQs also.
CVE-2025-71084	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: RDMA/cm: Fix leaking the multicast GID table reference If the CM ID is destroyed while the CM event for multicast creating is still queued the cancel_work_sync() will prevent the work from running which also prevents destroying the ah_attr. This leaks a refcount and triggers a WARN: GID entry ref leak for dev syz1 index 2 ref=573 WARNING: CPU: 1 PID: 655 at drivers/infiniband/core/cache.c:809 release_gid_table drivers/infiniband/core/cache.c:806 [inline] WARNING: CPU: 1 PID: 655 at drivers/infiniband/core/cache.c:809 gid_table_release_one+0x284/0x3cc drivers/infiniband/core/cache.c:886 Destroy the ah_attr after canceling the work, it is safe to call this twice.
CVE-2025-21888	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: RDMA/mlx5: Fix a WARN during dereg_mr for DM type Memory regions (MR) of type DM (device memory) do not have an associated umem. In the __mlx5_ib_dereg_mr() -> mlx5_free_priv_descs() flow, the code incorrectly takes the wrong branch, attempting to call dma_unmap_single() on a DMA address that is not mapped. This results in a WARN [1], as shown below. The issue is resolved by properly accounting for the DM type and ensuring the correct branch is selected in mlx5_free_priv_descs(). [1] WARNING: CPU: 12 PID: 1346 at drivers/iommu/dma-iommu.c:1230 iommu_dma_unmap_page+0x79/0x90 Modules linked in: ip6table_mangle ip6table_nat ip6table_filter ip6_tables iptable_mangle xt_conntrack xt_MASQUERADE nf_conntrack_netlink nfnetlink xt_addrtype iptable_nat nf_nat br_netfilter rpcsec_gss_krb5 auth_rpcgss oid_registry ovelay rpcrdma rdma_ucm ib_iser libiscsi scsi_transport_iscsi ib_umad rdma_cm ib_ipoib iw_cm ib_cm mlx5_ib ib_uverbs ib_core fuse mlx5_core CPU: 12 UID: 0 PID: 1346 Comm: ibv_rc_pingpong Not tainted 6.12.0-rc7+ #1631 Hardware name:

CVE	Severity	CVSS	CVSS Vector	Component	Description
					QEMU Standard PC (Q35 + ICH9, 2009), BIOS rel-1.13.0-0-gf21b5a4aeb02-prebuilt.qemu.org 04/01/2014 RIP: 0010:iommu_dma_unmap_page+0x79/0x90 Code: 2b 49 3b 29 72 26 49 3b 69 08 73 20 4d 89 f0 44 89 e9 4c 89 e2 48 89 ee 48 89 df 5b 5d 41 5c 41 5d 41 5e 41 5f e9 07 b8 88 ff <0f> 0b 5b 5d 41 5c 41 5d 41 5e 41 5f c3 cc cc cc cc 66 0f 1f 44 00 RSP: 0018:ffffc90001913a10 EFLAGS: 00010246 RAX: 0000000000000000 RBX: ffff88810194b0a8 RCX: 0000000000000000 RDX: 0000000000000000 RSI: 0000000000000000 RDI: 0000000000000001 RBP: ffff88810194b0a8 R08: 0000000000000000 R09: 0000000000000000 R10: 0000000000000001 R11: 0000000000000000 R12: 0000000000000000 R13: 0000000000000001 R14: 0000000000000000 R15: 0000000000000000 FS: 00007f537abdd740(0000) GS:ffff88885fb00000(0000) knlGS:0000000000000000 CS: 0010 DS: 0000 ES: 0000 CR0: 0000000080050033 CR2: 00007f537aeb8000 CR3: 000000010c248001 CR4: 0000000000372eb0 DR0: 0000000000000000 DR1: 0000000000000000 DR2: 0000000000000000 DR3: 0000000000000000 DR6: 00000000fffe0ff0 DR7: 0000000000000400 Call Trace: ? __warn+0x84/0x190 ? iommu_dma_unmap_page+0x79/0x90 ? report_bug+0xf8/0x1c0 ? handle_bug+0x55/0x90 ? exc_invalid_op+0x13/0x60 ? asm_exc_invalid_op+0x16/0x20 ? iommu_dma_unmap_page+0x79/0x90 dma_unmap_page_attrs+0xe6/0x290 mlx5_free_priv_descs+0xb0/0xe0 [mlx5_ib] __mlx5_ib_dereg_mr+0x37e/0x520 [mlx5_ib] ? _raw_spin_unlock_irq+0x24/0x40 ? wait_for_completion+0xfe/0x130 ? rdma_restrack_put+0x63/0xe0 [ib_core] ib_dereg_mr_user+0x5f/0x120 [ib_core] ? lock_release+0xc6/0x280 destroy_hw_idr_uobject+0x1d/0x60 [ib_uverbs] uverbs_destroy_uobject+0x58/0x1d0 [ib_uverbs] uobj_destroy+0x3f/0x70 [ib_uverbs] ib_uverbs_cmd_verbs+0x3e4/0xbb0 [ib_uverbs] ? __pfx_uverbs_destroy_def_handler+0x10/0x10 [ib_uverbs] ? lock_acquire+0xc1/0x2f0 ? ib_uverbs_ioctl+0xcb/0x170 [ib_uverbs] ? ib_uverbs_ioctl+0x116/0x170 [ib_uverbs] ? lock_release+0xc6/0x280 ib_uverbs_ioctl+0xe7/0x170 [ib_uverbs] ? ib_uverbs_ioctl+0xcb/0x170 [ib_uverbs] __x64_sys_ioctl+0x1b0/0xa70 do_syscall_64+0x6b/0x140 entry_SYSCALL_64_after_hwframe+0x76/0x7e RIP: 0033:0x7f537adaf17b Code: 0f 1e fa 48 8b 05 1d ad 0c 00 64 c7 00 26 00 00 00 48 c7 c0 ff ff ff ff c3 66 0f 1f 44 00 00 f3 0f 1e fa b8 10 00 00 00 0f 05 <48> 3d 01 f0 ff ff 73 01 c3 48 8b 0d ed ac 0c 00 f7 d8 64 89 01 48 RSP: 002b:00007ffff218f0b8 EFLAGS: 00000246 ORIG_RAX: 0000000000000010 RAX: ffffffffffffffda RBX: 00007ffff218f1d8 RCX: 00007f537adaf17b RDX: 00007ffff218f1c0 RSI: 00000000c0181b01 RDI: 0000000000000003 RBP: 00007ffff218f1a0 R08: 00007f537aa8d010 R09: 0000561ee2e4f270 R10: 00007f537aace3a8 R11: 0000000000000246 R12: 00007ffff218f190 R13: 000000000000001c R14: 0000561ee2e4d7c0 R15: 00007ffff218f450

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-39758	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: RDMA/siw: Fix the sendmsg byte count in siw_tcp_sendpages Ever since commit c2ff29e99a76 ("siw: Inline do_tcp_sendpages()"), we have been doing this: static int siw_tcp_sendpages(struct socket *s, struct page **page, int offset, size_t size) [...] /* Calculate the number of bytes we need to push, for this page * specifically */ size_t bytes = min_t(size_t, PAGE_SIZE - offset, size); /* If we can't splice it, then copy it in, as normal */ if (!sendpage_ok(page[i])) msg.msg_flags &= ~MSG_SPLICE_PAGES; /* Set the bvec pointing to the page, with len $bytes */ bvec_set_page(&bvec, page[i], bytes, offset); /* Set the iter to $size, aka the size of the whole sendpages (!!!) */ iov_iter_bvec(&msg.msg_iter, ITER_SOURCE, &bvec, 1, size); try_page_again: lock_sock(sk); /* Sendmsg with $size size (!!!) */ rv = tcp_sendmsg_locked(sk, &msg, size); This means we've been sending oversized iov_iters and tcp_sendmsg calls for a while. This has a been a benign bug because sendpage_ok() always returned true. With the recent slab allocator changes being slowly introduced into next (that disallow sendpage on large kmalloc allocations), we have recently hit out-of-bounds crashes, due to slight differences in iov_iter behavior between the MSG_SPLICE_PAGES and "regular" copy paths: (MSG_SPLICE_PAGES) skb_splice_from_iter iov_iter_extract_pages iov_iter_extract_bvec_pages uses i->nr_segs to correctly stop in its tracks before OoB'ing everywhere skb_splice_from_iter gets a "short" read (!MSG_SPLICE_PAGES) skb_copy_to_page_nocache copy=iov_iter_count [...] copy_from_iter /* this doesn't help */ if (unlikely(iter->count < len)) len = iter->count; iterate_bvec ... and we run off the bvecs Fix this by properly setting the iov_iter's byte count, plus sending the correct byte count to tcp_sendmsg_locked.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-23135	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: RISC-V: KVM: Teardown riscv specific bits after kvm_exit During a module removal, kvm_exit invokes arch specific disable call which disables AIA. However, we invoke aia_exit before kvm_exit resulting in the following warning. KVM kernel module can't be inserted afterwards due to inconsistent state of IRQ. [25469.031389] percpu IRQ 31 still enabled on CPU0! [25469.031732] WARNING: CPU: 3 PID: 943 at kernel/irq/manage.c:2476 __free_percpu_irq+0xa2/0x150 [25469.031804] Modules linked in: kvm(-) [25469.031848] CPU: 3 UID: 0 PID: 943 Comm: rmmod Not tainted 6.14.0-rc5-06947-g91c763118f47-dirty #2 [25469.031905] Hardware name: riscv-virtio,qemu (DT) [25469.031928] epc : __free_percpu_irq+0xa2/0x150 [25469.031976] ra : __free_percpu_irq+0xa2/0x150 [25469.032197] epc : ffffffff8007db1e ra : ffffffff8007db1e sp : ff2000000088bd50 [25469.032241] gp : ffffffff8131cef8 tp : ff60000080b96400 t0 : ff2000000088baf8 [25469.032285] t1 : fffffffffffffffc t2 : 5249207570637265 s0 : ff2000000088bd90 [25469.032329] s1 : ff60000098b21080 a0 : 037d527a15eb4f00 a1 : 037d527a15eb4f00 [25469.032372] a2 : 0000000000000023 a3 : 0000000000000001 a4 : ffffffff8122dbf8 [25469.032410] a5 : 0000000000000fff a6 : 0000000000000000 a7 : ffffffff8122dc10 [25469.032448] s2 : ff60000080c22eb0 s3 : 0000000200000022 s4 : 000000000000001f [25469.032488] s5 : ff60000080c22e00 s6 : ffffffff80c351c0 s7 : 0000000000000000 [25469.032582] s8 : 0000000000000003 s9 : 000055556b7fb490 s10: 00007ffff0e12fa0 [25469.032621] s11: 00007ffff0e13e9a t3 : ffffffff81354ac7 t4 : ffffffff81354ac7 [25469.032664] t5 : ffffffff81354ac8 t6 : ffffffff81354ac7 [25469.032698] status: 0000000200000100 badaddr: ffffffff8007db1e cause: 0000000000000003 [25469.032738] [] __free_percpu_irq+0xa2/0x150 [25469.032797] [] free_percpu_irq+0x30/0x5e [25469.032856] [] kvm_riscv_aia_exit+0x40/0x42 [kvm] [25469.033947] [] cleanup_module+0x10/0x32 [kvm] [25469.035300] [] __riscv_sys_delete_module+0x18e/0x1fc [25469.035374] [] syscall_handler+0x3a/0x46 [25469.035456] [] do_trap_ecall_u+0x72/0x134 [25469.035536] [] handle_exception+0x148/0x156 Invoke aia_exit and other arch specific cleanup functions after kvm_exit so that disable gets a chance to be called first before exit.
CVE-2025-39734	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: Revert "fs/ntfs3: Replace inode_trylock with inode_lock" This reverts commit 69505fe98f198ee813898cbcaf6770949636430b. Initially, conditional lock acquisition was removed to fix an xfstest bug that was observed during internal testing. The deadlock reported by syzbot is resolved by reintroducing conditional acquisition. The xfstest bug no longer occurs on kernel version 6.16-rc1 during internal testing. I assume that changes in other modules may have contributed to this.
CVE-2025-38660	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: [ceph] parse_longname(): strrchr() expects NUL-terminated string ... and parse_longname() is not guaranteed that. That's the reason why it uses kmemdup_nul() to build the argument for kstrtou64(); the problem is, kstrtou64() is not the only thing that need it. Just get a NUL-terminated copy of the entire thing and be done with that...
CVE-2026-23128	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: arm64: Set __nocfi on swsusp_arch_resume() A DABT is reported[1] on an android based system when resume from

CVE	Severity	CVSS	CVSS Vector	Component	Description
					hiberate. This happens because swsusp_arch_suspend_exit() is marked with SYM_CODE_*() and does not have a CFI hash, but swsusp_arch_resume() will attempt to verify the CFI hash when calling a copy of swsusp_arch_suspend_exit(). Given that there's an existing requirement that the entrypoint to swsusp_arch_suspend_exit() is the first byte of the .hibernate_exit.text section, we cannot fix this by marking swsusp_arch_suspend_exit() with SYM_FUNC_*(). The simplest fix for now is to disable the CFI check in swsusp_arch_resume(). Mark swsusp_arch_resume() as __nocfi to disable the CFI check. [1] [22.991934][T1] Unable to handle kernel paging request at virtual address 0000000109170ffc [22.991934][T1] Mem abort info: [22.991934][T1] ESR = 0x0000000096000007 [22.991934][T1] EC = 0x25: DABT (current EL), IL = 32 bits [22.991934] [T1] SET = 0, FnV = 0 [22.991934][T1] EA = 0, S1PTW = 0 [22.991934][T1] FSC = 0x07: level 3 translation fault [22.991934][T1] Data abort info: [22.991934][T1] ISV = 0, ISS = 0x00000007, ISS2 = 0x00000000 [22.991934][T1] CM = 0, WnR = 0, TnD = 0, TagAccess = 0 [22.991934][T1] GCS = 0, Overlay = 0, DirtyBit = 0, Xs = 0 [22.991934][T1] [0000000109170ffc] user address but active_mm is swapper [22.991934][T1] Internal error: Oops: 0000000096000007 [#1] PREEMPT SMP [22.991934][T1] Dumping ftrace buffer: [22.991934][T1] (ftrace buffer empty) [22.991934][T1] Modules linked in: [22.991934][T1] CPU: 0 PID: 1 Comm: swapper/0 Not tainted 6.6.98-android15-8-g0b1d2aee7fc3-dirty-4k #1 688c7060a825a3ac418fe53881730b355915a41 9 [22.991934][T1] Hardware name: Unisoc UMS9360-base Board (DT) [22.991934][T1] pstate: 804000c5 (Nzcv daIF +PAN -UAO -TCO -DIT -SSBS BTYPE=--) [22.991934][T1] pc : swsusp_arch_resume+0x2ac/0x344 [22.991934][T1] lr : swsusp_arch_resume+0x294/0x344 [22.991934][T1] sp : ffffffc08006b960 [22.991934] [T1] x29: ffffffc08006b9c0 x28: 0000000000000000 x27: 0000000000000000 [22.991934][T1] x26: 0000000000000000 x25: 0000000000000000 x24: 0000000000000820 [22.991934][T1] x23: ffffffd0817e3000 x22: ffffffd0817e3000 x21: 0000000000000000 [22.991934][T1] x20: fffffff8089171000 x19: ffffffd08252c8c8 x18: ffffffc080061058 [22.991934][T1] x17: 00000000529c6ef0 x16: 00000000529c6ef0 x15: 0000000000000004 [22.991934][T1] x14: fffffff8178c88000 x13: 0000000000000006 x12: 0000000000000000 [22.991934][T1] x11: 0000000000000015 x10: 0000000000000001 x9 : ffffffd082533000 [22.991934][T1] x8 : 0000000109171000 x7 : 205b5d3433393139 x6 : 392e32322020205b [22.991934][T1] x5 : 000000010916f000 x4 : 000000008164b000 x3 : ffffff808a4e0530 [22.991934][T1] x2 : ffffffd08058e784 x1 : 0000000082326000 x0 : 000000010a283000 [22.991934][T1] Call trace: [22.991934][T1] swsusp_arch_resume+0x2ac/0x344 [22.991934][T1] hibernation_restore+0x158/0x18c [22.991934][T1] load_image_and_restore+0xb0/0xec [22.991934][T1] software_resume+0xf4/0x19c [22.991934][T1] software_resume_initcall+0x34/0x78 [22.991934][T1] do_one_initcall+0xe8/0x370 [22.991934][T1] do_initcall_level+0xc8/0x19c [22.991934][T1] do_initcalls+0x70/0xc0 [22.991934][T1] do_basic_setup+0x1c/0x28 [22.991934][T1] kernel_init_freeable+0xe0/0x148 [22.991934][T1] kernel_init+0x20/0x1a8 [22.991934][T1] ret_from_fork+0x10/0x20 [22.991934][T1] Code: a9400a61 f94013e0 f9438923 f9400a64 (b85fc110) [catalin.marinas@arm.com: commit log updated by Mark Rutland]
CVE-2025-38246	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: bnxt: properly flush XDP redirect lists We encountered following crash when

CVE	Severity	CVSS	CVSS Vector	Component	Description
					testing a XDP_REDIRECT feature in production: [56251.579676] list_add corruption. next->prev should be prev (ffff93120dd40f30), but was ffffb301ef3a6740. (next=ffff93120dd 40f30). [56251.601413] ------------[cut here]------------ [56251.611357] kernel BUG at lib/list_debug.c:29! [56251.621082] Oops: invalid opcode: 0000 [#1] PREEMPT SMP NOPTI [56251.632073] CPU: 111 UID: 0 PID: 0 Comm: swapper/111 Kdump: loaded Tainted: P O 6.12.33-cloudflare-2025.6. 3 #1 [56251.653155] Tainted: [P]=PROPRIETARY_MODULE, [O]=OOT_MODULE [56251.663877] Hardware name: MiTAC GC68B-B8032-G11P6-GPU/S8032GM-HE-CFR, BIOS V7.020.B10-sig 01/22/2025 [56251.682626] RIP: 0010:__list_add_valid_or_report+0x4b/0xa0 [56251.693203] Code: 0e 48 c7 c7 68 e7 d9 97 e8 42 16 fe ff 0f 0b 48 8b 52 08 48 39 c2 74 14 48 89 f1 48 c7 c7 90 e7 d9 97 48 89 c6 e8 25 16 fe ff <0f> 0b 4c 8b 02 49 39 f0 74 14 48 89 d1 48 c7 c7 e8 e7 d9 97 4c 89 [56251.725811] RSP: 0018:ffff93120dd40b80 EFLAGS: 00010246 [56251.736094] RAX: 0000000000000075 RBX: ffffb301e6bba9d8 RCX: 0000000000000000 [56251.748260] RDX: 0000000000000000 RSI: ffff9149afda0b80 RDI: ffff9149afda0b80 [56251.760349] RBP: ffff9131e49c8000 R08: 0000000000000000 R09: ffff93120dd40a18 [56251.772382] R10: ffff9159cf2ce1a8 R11: 0000000000000003 R12: ffff911a80850000 [56251.784364] R13: ffff93120fbc7000 R14: 0000000000000010 R15: ffff9139e7510e40 [56251.796278] FS: 0000000000000000(0000) GS:ffff9149afd80000(0000) knlGS:0000000000000000 [56251.809133] CS: 0010 DS: 0000 ES: 0000 CR0: 0000000080050033 [56251.819561] CR2: 00007f5e85e6f300 CR3: 00000038b85e2006 CR4: 0000000000770ef0 [56251.831365] PKRU: 55555554 [56251.838653] Call Trace: [56251.845560] [56251.851943] cpu_map_enqueue.cold+0x5/0xa [56251.860243] xdp_do_redirect+0x2d9/0x480 [56251.868388] bnxt_rx_xdp+0x1d8/0x4c0 [bnxt_en] [56251.877028] bnxt_rx_pkt+0x5f7/0x19b0 [bnxt_en] [56251.885665] ? cpu_max_write+0x1e/0x100 [56251.893510] ? srso_alias_return_thunk+0x5/0xfbef5 [56251.902276] __bnxt_poll_work+0x190/0x340 [bnxt_en] [56251.911058] bnxt_poll+0xab/0x1b0 [bnxt_en] [56251.919041] ? srso_alias_return_thunk+0x5/0xfbef5 [56251.927568] ? srso_alias_return_thunk+0x5/0xfbef5 [56251.935958] ? srso_alias_return_thunk+0x5/0xfbef5 [56251.944250] __napi_poll+0x2b/0x160 [56251.951155] bpf_trampoline_6442548651+0x79/0x123 [56251.959262] __napi_poll+0x5/0x160 [56251.966037] net_rx_action+0x3d2/0x880 [56251.973133] ? srso_alias_return_thunk+0x5/0xfbef5 [56251.981265] ? srso_alias_return_thunk+0x5/0xfbef5 [56251.989262] ? __hrtimer_run_queues+0x162/0x2a0 [56251.996967] ? srso_alias_return_thunk+0x5/0xfbef5 [56252.004875] ? srso_alias_return_thunk+0x5/0xfbef5 [56252.012673] ? bnxt_msix+0x62/0x70 [bnxt_en] [56252.019903] handle_softirqs+0xcf/0x270 [56252.026650] irq_exit_rcu+0x67/0x90 [56252.032933] common_interrupt+0x85/0xa0 [56252.039498] [56252.044246] [56252.048935] asm_common_interrupt+0x26/0x40 [56252.055727] RIP: 0010:cpuidle_enter_state+0xb8/0x420 [56252.063305] Code: dc 01 00 00 e8 f9 79 3b ff e8 64 f7 ff ff 49 89 c5 0f 1f 44 00 00 31 ff e8 a5 32 3a ff 45 84 ff 0f 85 ae 01 00 00 fb 45 85 f6 <0f> 88 88 01 00 00 48 8b 04 24 49 63 ce 4c 89 ea 48 6b f1 68 48 29 [56252.088911] RSP: 0018:ffff93120c97fe98 EFLAGS: 00000202

CVE	Severity	CVSS	CVSS Vector	Component	Description
					[56252.096912] RAX: ffff9149afd80000 RBX: ffff9141d3a72800 RCX: 0000000000000000 [56252.106844] RDX: 00003329176c6b98 RSI: ffffffe36db3fdc7 RDI: 0000000000000000 [56252.116733] RBP: 0000000000000002 R08: 0000000000000002 R09: 000000000000004e [56252.126652] R10: ffff9149afdb30c4 R11: 071c71c71c71c71c R12: ffffffff985ff860 [56252.136637] R13: 00003329176c6b98 R14: 0000000000000002 R15: 0000000000000000 [56252.146667] ? cpuidle_enter_state+0xab/0x420 [56252.153909] cpuidle_enter+0x2d/0x40 [56252.160360] do_idle+0x176/0x1c0 [56252.166456 ---truncated---
CVE-2025-22108	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: bnxt_en: Mask the bd_cnt field in the TX BD properly The bd_cnt field in the TX BD specifies the total number of BDs for the TX packet. The bd_cnt field has 5 bits and the maximum number supported is 32 with the value 0. CONFIG_MAX_SKB_FRAGS can be modified and the total number of SKB fragments can approach or exceed the maximum supported by the chip. Add a macro to properly mask the bd_cnt field so that the value 32 will be properly masked and set to 0 in the bd_cnd field. Without this patch, the out-of-range bd_cnt value will corrupt the TX BD and may cause TX timeout. The next patch will check for values exceeding 32.
CVE-2025-38439	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: bnxt_en: Set DMA unmap len correctly for XDP_REDIRECT When transmitting an XDP_REDIRECT packet, call dma_unmap_len_set() with the proper length instead of 0. This bug triggers this warning on a system with IOMMU enabled: WARNING: CPU: 36 PID: 0 at drivers/iommu/dma-iommu.c:842 __iommu_dma_unmap+0x159/0x170 RIP: 0010:__iommu_dma_unmap+0x159/0x170 Code: a8 00 00 00 00 48 c7 45 b0 00 00 00 00 48 c7 45 c8 00 00 00 00 48 c7 45 a0 ff ff ff ff 4c 89 45 b8 4c 89 45 c0 e9 77 ff ff ff <0f> 0b e9 60 ff ff ff e8 8b bf 6a 00 66 66 2e 0f 1f 84 00 00 00 00 RSP: 0018:ff22d31181150c88 EFLAGS: 00010206 RAX: 0000000000002000 RBX: 00000000e13a0000 RCX: 0000000000000000 RDX: 0000000000000000 RSI: 0000000000000000 RDI: 0000000000000000 RBP: ff22d31181150cf0 R08: ff22d31181150ca8 R09: 0000000000000000 R10: 0000000000000000 R11: ff22d311d36c9d80 R12: 0000000000001000 R13: ff13544d10645010 R14: ff22d31181150c90 R15: ff13544d0b2bac00 FS: 0000000000000000(0000) GS:ff13550908a00000(0000) knlGS:0000000000000000 CS: 0010 DS: 0000 ES: 0000 CR0: 0000000080050033 CR2: 00005be909dacff8 CR3: 0008000173408003 CR4: 0000000000f71ef0 PKRU: 55555554 Call Trace: ? show_regs+0x6d/0x80 ? __warn+0x89/0x160 ? __iommu_dma_unmap+0x159/0x170 ? report_bug+0x17e/0x1b0 ? handle_bug+0x46/0x90 ? exc_invalid_op+0x18/0x80 ? asm_exc_invalid_op+0x1b/0x20 ? __iommu_dma_unmap+0x159/0x170 ? __iommu_dma_unmap+0xb3/0x170 iommu_dma_unmap_page+0x4f/0x100 dma_unmap_page_attrs+0x52/0x220 ? srso_alias_return_thunk+0x5/0xfbef5 ? xdp_return_frame+0x2e/0xd0 bnxt_tx_int_xdp+0xdf/0x440 [bnxt_en] __bnxt_poll_work_done+0x81/0x1e0 [bnxt_en] bnxt_poll+0xd3/0x1e0 [bnxt_en]

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-22105	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: bonding: check xdp prog when set bond mode Following operations can trigger a warning[1]: ip netns add ns1 ip netns exec ns1 ip link add bond0 type bond mode balance-rr ip netns exec ns1 ip link set dev bond0 xdp obj af_xdp_kern.o sec xdp ip netns exec ns1 ip link set bond0 type bond mode broadcast ip netns del ns1 When delete the namespace, dev_xdp_uninstall() is called to remove xdp program on bond dev, and bond_xdp_set() will check the bond mode. If bond mode is changed after attaching xdp program, the warning may occur. Some bond modes (broadcast, etc.) do not support native xdp. Set bond mode with xdp program attached is not good. Add check for xdp program when set bond mode. [1] ------------[cut here]------------ WARNING: CPU: 0 PID: 11 at net/core/dev.c:9912 unregister_netdevice_many_notify+0x8d9/0x930 Modules linked in: CPU: 0 UID: 0 PID: 11 Comm: kworker/u4:0 Not tainted 6.14.0-rc4 #107 Hardware name: QEMU Standard PC (i440FX + PIIX, 1996), BIOS rel-1.15.0-0-g2dd4b9b3f840-prebuilt.qemu.org 04/01/2014 Workqueue: netns cleanup_net RIP: 0010:unregister_netdevice_many_notify+0x8d9/0x930 Code: 00 00 48 c7 c6 6f e3 a2 82 48 c7 c7 d0 b3 96 82 e8 9c 10 3e ... RSP: 0018:ffffc90000063d80 EFLAGS: 00000282 RAX: 00000000ffffffa1 RBX: ffff888004959000 RCX: 00000000ffffdfff RDX: 0000000000000000 RSI: 00000000ffffffea RDI: ffffc90000063b48 RBP: ffffc90000063e28 R08: ffffffff82d39b28 R09: 00000000000009ffb R10: 0000000000000175 R11: ffffffff82d09b40 R12: ffff8880049598e8 R13: 0000000000000001 R14: dead000000000100 R15: ffffc90000045000 FS: 0000000000000000(0000) GS:ffff888007a00000(0000) knlGS:0000000000000000 CS: 0010 DS: 0000 ES: 0000 CR0: 0000000080050033 CR2: 000000000d406b60 CR3: 000000000483e000 CR4: 00000000000006f0 Call Trace: ? __warn+0x83/0x130 ? unregister_netdevice_many_notify+0x8d9/0x930 ? report_bug+0x18e/0x1a0 ? handle_bug+0x54/0x90 ? exc_invalid_op+0x18/0x70 ? asm_exc_invalid_op+0x1a/0x20 ? unregister_netdevice_many_notify+0x8d9/0x930 ? bond_net_exit_batch_rtnl+0x5c/0x90 cleanup_net+0x237/0x3d0 process_one_work+0x163/0x390 worker_thread+0x293/0x3b0 ? __pfx_worker_thread+0x10/0x10 kthread+0xec/0x1e0 ? __pfx_kthread+0x10/0x10 ? __pfx_kthread+0x10/0x10 ret_from_fork+0x2f/0x50 ? __pfx_kthread+0x10/0x10 ret_from_fork_asm+0x1a/0x30 ---[end trace 0000000000000000]---

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2026-23119	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: bonding: provide a net pointer to __skb_flow_dissect() After 3cbf4ffba5ee ("net: plumb network namespace into __skb_flow_dissect") we have to provide a net pointer to __skb_flow_dissect(), either via skb->dev, skb->sk, or a user provided pointer. In the following case, syzbot was able to cook a bare skb. WARNING: net/core/flow_dissector.c:1131 at __skb_flow_dissect+0xb57/0x68b0 net/core/flow_dissector.c:1131, CPU#1: syz.2.1418/11053 Call Trace: bond_flow_dissect drivers/net/bonding/bond_main.c:4093 [inline] __bond_xmit_hash+0x2d7/0xba0 drivers/net/bonding/bond_main.c:4157 bond_xmit_hash_xdp drivers/net/bonding/bond_main.c:4208 [inline] bond_xdp_xmit_3ad_xor_slave_get drivers/net/bonding/bond_main.c:5139 [inline] bond_xdp_get_xmit_slave+0x1fd/0x710 drivers/net/bonding/bond_main.c:5515 xdp_master_redirect+0x13f/0x2c0 net/core/filter.c:4388 bpf_prog_run_xdp include/net/xdp.h:700 [inline] bpf_test_run+0x6b2/0x7d0 net/bpf/test_run.c:421 bpf_prog_test_run_xdp+0x795/0x10e0 net/bpf/test_run.c:1390 bpf_prog_test_run+0x2c7/0x340 kernel/bpf/syscall.c:4703 __sys_bpf+0x562/0x860 kernel/bpf/syscall.c:6182 __do_sys_bpf kernel/bpf/syscall.c:6274 [inline] __se_sys_bpf kernel/bpf/syscall.c:6272 [inline] __x64_sys_bpf+0x7c/0x90 kernel/bpf/syscall.c:6272 do_syscall_x64 arch/x86/entry/syscall_64.c:63 [inline] do_syscall_64+0xec/0xf80 arch/x86/entry/syscall_64.c:94
CVE-2024-47703	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: bpf, lsm: Add check for BPF LSM return value A bpf prog returning a positive number attached to file_alloc_security hook makes kernel panic. This happens because file system can not filter out the positive number returned by the LSM prog using IS_ERR, and misinterprets this positive number as a file pointer. Given that hook file_alloc_security never returned positive number before the introduction of BPF LSM, and other BPF LSM hooks may encounter similar issues, this patch adds LSM return value check in verifier, to ensure no unexpected value is returned.
CVE-2026-23140	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: bpf, test_run: Subtract size of xdp_frame from allowed metadata size The xdp_frame structure takes up part of the XDP frame headroom, limiting the size of the metadata. However, in bpf_test_run, we don't take this into account, which makes it possible for userspace to supply a metadata size that is too large (taking up the entire headroom). If userspace supplies such a large metadata size in live packet mode, the xdp_update_frame_from_buff() call in xdp_test_run_init_page() call will fail, after which packet transmission proceeds with an uninitialised frame structure, leading to the usual Bad Stuff. The commit in the Fixes tag fixed a related bug where the second check in xdp_update_frame_from_buff() could fail, but did not add any additional constraints on the metadata size. Complete the fix by adding an additional check on the metadata size. Reorder the checks slightly to make the logic clearer and add a comment.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38640	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: bpf: Disable migration in nf_hook_run_bpf(). syzbot reported that the netfilter bpf prog can be called without migration disabled in xmit path. Then the assertion in __bpf_prog_run() fails, triggering the splat below. [0] Let's use bpf_prog_run_pin_on_cpu() in nf_hook_run_bpf(). [0]: BUG: assuming non migratable context at ./include/linux/filter.h:703 in_atomic(): 0, irqs_disabled(): 0, migration_disabled() 0 pid: 5829, name: sshd-session 3 locks held by sshd-session/5829: #0: ffff88807b4e4218 (sk_lock-AF_INET){+.+.}-{0:0}, at: lock_sock include/net/sock.h:1667 [inline] #0: ffff88807b4e4218 (sk_lock-AF_INET){+.+.}-{0:0}, at: tcp_sendmsg+0x20/0x50 net/ipv4/tcp.c:1395 #1: ffffffff8e5c4e00 (rcu_read_lock){....}-{1:3}, at: rcu_lock_acquire include/linux/rcupdate.h:331 [inline] #1: ffffffff8e5c4e00 (rcu_read_lock){....}-{1:3}, at: rcu_read_lock include/linux/rcupdate.h:841 [inline] #1: ffffffff8e5c4e00 (rcu_read_lock){....}-{1:3}, at: __ip_queue_xmit+0x69/0x26c0 net/ipv4/ip_output.c:470 #2: ffffffff8e5c4e00 (rcu_read_lock){....}-{1:3}, at: rcu_lock_acquire include/linux/rcupdate.h:331 [inline] #2: ffffffff8e5c4e00 (rcu_read_lock){....}-{1:3}, at: rcu_read_lock include/linux/rcupdate.h:841 [inline] #2: ffffffff8e5c4e00 (rcu_read_lock){....}-{1:3}, at: nf_hook+0xb2/0x680 include/linux/netfilter.h:241 CPU: 0 UID: 0 PID: 5829 Comm: sshd-session Not tainted 6.16.0-rc6-syzkaller-00002-g155a3c003e55 #0 PREEMPT(full) Hardware name: Google Google Compute Engine/Google Compute Engine, BIOS Google 05/07/2025 Call Trace: __dump_stack lib/dump_stack.c:94 [inline] dump_stack_lvl+0x16c/0x1f0 lib/dump_stack.c:120 __cant_migrate kernel/sched/core.c:8860 [inline] __cant_migrate+0x1c7/0x250 kernel/sched/core.c:8834 __bpf_prog_run include/linux/filter.h:703 [inline] bpf_prog_run include/linux/filter.h:725 [inline] nf_hook_run_bpf+0x83/0x1e0 net/netfilter/nf_bpf_link.c:20 nf_hook_entry_hookfn include/linux/netfilter.h:157 [inline] nf_hook_slow+0xbb/0x200 net/netfilter/core.c:623 nf_hook+0x370/0x680 include/linux/netfilter.h:272 NF_HOOK_COND include/linux/netfilter.h:305 [inline] ip_output+0x1bc/0x2a0 net/ipv4/ip_output.c:433 dst_output include/net/dst.h:459 [inline] ip_local_out net/ipv4/ip_output.c:129 [inline] __ip_queue_xmit+0x1d7d/0x26c0 net/ipv4/ip_output.c:527 __tcp_transmit_skb+0x2686/0x3e90 net/ipv4/tcp_output.c:1479 tcp_transmit_skb net/ipv4/tcp_output.c:1497 [inline] tcp_write_xmit+0x1274/0x84e0 net/ipv4/tcp_output.c:2838 __tcp_push_pending_frames+0xaf/0x390 net/ipv4/tcp_output.c:3021 tcp_push+0x225/0x700 net/ipv4/tcp.c:759 tcp_sendmsg_locked+0x1870/0x42b0 net/ipv4/tcp.c:1359 tcp_sendmsg+0x2e/0x50 net/ipv4/tcp.c:1396 inet_sendmsg+0xb9/0x140 net/ipv4/af_inet.c:851 sock_sendmsg_nosec net/socket.c:712 [inline] __sock_sendmsg net/socket.c:727 [inline] sock_write_iter+0x4aa/0x5b0 net/socket.c:1131 new_sync_write fs/read_write.c:593 [inline] vfs_write+0x6c7/0x1150 fs/read_write.c:686 ksys_write+0x1f8/0x250 fs/read_write.c:738 do_syscall_x64 arch/x86/entry/syscall_64.c:63 [inline] do_syscall_64+0xcd/0x4c0 arch/x86/entry/syscall_64.c:94 entry_SYSCALL_64_after_hwframe+0x77/0x7f RIP: 0033:0x7fe7d365d407 Code: 48 89 fa 4c 89 df e8 38 aa 00 00 8b 93 08 03 00 00 59 5e 48 83 f8 fc 74 1a 5b c3 0f 1f 84 00 00 00 00 00 48 8b 44 24 10 0f 05 <5b> c3 0f 1f 80 00 00 00 00 83 e2 39 83 fa 08 75 de e8 23 ff ff ff RSP:

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-68725	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: bpf: Do not let BPF test infra emit invalid GSO types to stack Yinhao et al. reported that their fuzzer tool was able to trigger a skb_warn_bad_offload() from netif_skb_features() -> gso_features_check(). When a BPF program - triggered via BPF test infra - pushes the packet to the loopback device via bpf_clone_redirect() then mentioned offload warning can be seen. GSO-related features are then rightfully disabled. We get into this situation due to convert___skb_to_skb() setting gso_segs and gso_size but not gso_type. Technically, it makes sense that this warning triggers since the GSO properties are malformed due to the gso_type. Potentially, the gso_type could be marked non-trustworthy through setting it at least to SKB_GSO_DODGY without any other specific assumptions, but that also feels wrong given we should not go further into the GSO engine in the first place. The checks were added in 121d57af308d ("gso: validate gso_type in GSO handlers") because there were malicious (syzbot) senders that combine a protocol with a non-matching gso_type. If we would want to drop such packets, gso_features_check() currently only returns feature flags via netif_skb_features(), so one location for potentially dropping such skbs could be validate_xmit_unreadable_skb(), but then otoh it would be an additional check in the fast-path for a very corner case. Given bpf_clone_redirect() is the only place where BPF test infra could emit such packets, lets reject them right there.
CVE-2024-47702	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: bpf: Fail verification for sign-extension of packet data/data_end/data_meta syzbot reported a kernel crash due to commit 1f1e864b6555 ("bpf: Handle sign-extenstin ctx member accesses"). The reason is due to sign-extension of 32-bit load for packet data/data_end/data_meta uapi field. The original code looks like: r2 = *(s32 *)(r1 + 76) /* load __sk_buff->data */ r3 = *(u32 *)(r1 + 80) /* load __sk_buff->data_end */ r0 = r2 r0 += 8 if r3 > r0 goto +1 ... Note that __sk_buff->data load has 32-bit sign extension. After verification and convert_ctx_accesses(), the final asm code looks like: r2 = *(u64 *)(r1 +208) r2 = (s32)r2 r3 = *(u64 *)(r1 +80) r0 = r2 r0 += 8 if r3 > r0 goto pc+1 ... Note that 'r2 = (s32)r2' may make the kernel __sk_buff->data address invalid which may cause runtime failure. Currently, in C code, typically we have void *data = (void *)(long)skb->data; void *data_end = (void *)(long)skb->data_end; ... and it will generate r2 = *(u64 *)(r1 +208) r3 = *(u64 *)(r1 +80) r0 = r2 r0 += 8 if r3 > r0 goto pc+1 If we allow sign-extension, void *data = (void *)(long)(int)skb->data; void *data_end = (void *)(long)skb->data_end; ... the generated code looks like r2 = *(u64 *)(r1 +208) r2 <<= 32 r2 s>>= 32 r3 = *(u64 *)(r1 +80) r0 = r2 r0 += 8 if r3 > r0 goto pc+1 and this will cause verification failure since "r2 <<= 32" is not allowed as "r2" is a packet pointer. To fix this issue for case r2 = *(s32 *)(r1 + 76) /* load __sk_buff->data */ this patch added additional checking in is_valid_access() callback function for packet data/data_end/data_meta access. If those accesses are with sign-extenstion, the verification will fail. [1] https://lore.kernel.org/bpf/000000000000c90eee061d236d37@google.com/

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2026-22994	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: bpf: Fix reference count leak in bpf_prog_test_run_xdp() syzbot is reporting unregister_netdevice: waiting for sit0 to become free. Usage count = 2 problem. A debug printk() patch found that a refcount is obtained at xdp_convert_md_to_buff() from bpf_prog_test_run_xdp(). According to commit ec94670fcb3b ("bpf: Support specifying ingress via xdp_md context in BPF_PROG_TEST_RUN"), the refcount obtained by xdp_convert_md_to_buff() will be released by xdp_convert_buff_to_md(). Therefore, we can consider that the error handling path introduced by commit 1c1949982524 ("bpf: introduce frags support to bpf_prog_test_run_xdp()") forgot to call xdp_convert_buff_to_md().
CVE-2025-39748	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: bpf: Forget ranges when refining tnum after JSET Syzbot reported a kernel warning due to a range invariant violation on the following BPF program. 0: call bpf_get_netns_cookie 1: if r0 == 0 goto 2: if r0 & 0xffffffff goto The issue is on the path where we fall through both jumps. That path is unreachable at runtime: after insn 1, we know r0 != 0, but with the sign extension on the jset, we would only fallthrough insn 2 if r0 == 0. Unfortunately, is_branch_taken() isn't currently able to figure this out, so the verifier walks all branches. The verifier then refines the register bounds using the second condition and we end up with inconsistent bounds on this unreachable path: 1: if r0 == 0 goto r0: u64=[0x1, 0xffffffffffffffff] var_off=(0, 0xffffffffffffffff) 2: if r0 & 0xffffffff goto r0 before reg_bounds_sync: u64=[0x1, 0xffffffffffffffff] var_off=(0, 0) r0 after reg_bounds_sync: u64=[0x1, 0] var_off=(0, 0) Improving the range refinement for JSET to cover all cases is tricky. We also don't expect many users to rely on JSET given LLVM doesn't generate those instructions. So instead of improving the range refinement for JSETs, Eduard suggested we forget the ranges whenever we're narrowing tnums after a JSET. This patch implements that approach.
CVE-2025-38528	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: bpf: Reject %p% format string in bprintf-like helpers static const char fmt[] = "%p%"; bpf_trace_printk(fmt, sizeof(fmt)); The above BPF program isn't rejected and causes a kernel warning at runtime: Please remove unsupported %\x00 in format string WARNING: CPU: 1 PID: 7244 at lib/vsprintf.c:2680 format_decode+0x49c/0x5d0 This happens because bpf_bprintf_prepare skips over the second %, detected as punctuation, while processing %p. This patch fixes it by not skipping over punctuation. %\x00 is then processed in the next iteration and rejected.
CVE-2024-46754	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: bpf: Remove tst_run from lwt_seg6local_prog_ops. The syzbot reported that the lwt_seg6 related BPF ops can be invoked via bpf_test_run() without without entering input_action_end_bpf() first. Martin KaFai Lau said that self test for BPF_PROG_TYPE_LWT_SEG6LOCAL probably didn't work since it was introduced in commit 04d4b274e2a ("ipv6: sr: Add seg6local action End.BPF"). The reason is that the per-CPU variable seg6_bpf_srh_states::srh is never assigned in the self test case but each BPF function expects it. Remove test_run for BPF_PROG_TYPE_LWT_SEG6LOCAL.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-39886	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: bpf: Tell memcg to use allow_spinning=false path in bpf_timer_init() Currently, calling bpf_map_kmalloc_node() from __bpf_async_init() can cause various locking issues; see the following stack trace (edited for style) as one example: ... [10.011566] do_raw_spin_lock.cold [10.011570] try_to_wake_up (5) double-acquiring the same [10.011575] kick_pool rq_lock, causing a hardlockup [10.011579] __queue_work [10.011582] queue_work_on [10.011585] kernfs_notify [10.011589] cgroup_file_notify [10.011593] try_charge_memcg (4) memcg accounting raises an [10.011597] obj_cgroup_charge_pages MEMCG_MAX event [10.011599] obj_cgroup_charge_account [10.011600] __memcg_slab_post_alloc_hook [10.011603] __kmalloc_node_noprof ... [10.011611] bpf_map_kmalloc_node [10.011612] __bpf_async_init [10.011615] bpf_timer_init (3) BPF calls bpf_timer_init() [10.011617] bpf_prog_xxxxxxxxxxxxxxxx_fcg_runnable [10.011619] bpf__sched_ext_ops_runnable [10.011620] enqueue_task_scx (2) BPF runs with rq_lock held [10.011622] enqueue_task [10.011626] ttwu_do_activate [10.011629] sched_ttwu_pending (1) grabs rq_lock ... The above was reproduced on bpf-next (b338cf849ec8) by modifying ./tools/sched_ext/scx_flatcg.bpf.c to call bpf_timer_init() during ops.runnable(), and hacking the memcg accounting code a bit to make a bpf_timer_init() call more likely to raise an MEMCG_MAX event. We have also run into other similar variants (both internally and on bpf-next), including double-acquiring cgroup_file_kn_lock, the same worker_pool::lock, etc. As suggested by Shakeel, fix this by using __GFP_HIGH instead of GFP_ATOMIC in __bpf_async_init(), so that e.g. if try_charge_memcg() raises an MEMCG_MAX event, we call __memcg_memory_event() with @allow_spinning=false and avoid calling cgroup_file_notify() there. Depends on mm patch "memcg: skip cgroup_file_notify if spinning is not allowed": https://lore.kernel.org/bpf/20250905201606.66198-1-shakeel.butt@linux.dev/ v0 approach s/bpf_map_kmalloc_node/bpf_mem_alloc/ https://lore.kernel.org/bpf/20250905061919.439648-1-yepeilin@google.com/ v1 approach: https://lore.kernel.org/bpf/20250905234547.862249-1-yepeilin@google.com/

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2024-50138	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: bpf: Use raw_spinlock_t in ringbuf The function __bpf_ringbuf_reserve is invoked from a tracepoint, which disables preemption. Using spinlock_t in this context can lead to a "sleep in atomic" warning in the RT variant. This issue is illustrated in the example below: BUG: sleeping function called from invalid context at kernel/locking/spinlock_rt.c:48 in_atomic(): 1, irqs_disabled(): 0, non_block: 0, pid: 556208, name: test_progs preempt_count: 1, expected: 0 RCU nest depth: 1, expected: 1 INFO: lockdep is turned off. Preemption disabled at: [] migrate_enable+0xc0/0x39c CPU: 7 PID: 556208 Comm: test_progs Tainted: G Hardware name: Qualcomm SA8775P Ride (DT) Call trace: dump_backtrace+0xac/0x130 show_stack+0x1c/0x30 dump_stack_lvl+0xac/0xe8 dump_stack+0x18/0x30 __might_resched+0x3bc/0x4fc rt_spin_lock+0x8c/0x1a4 __bpf_ringbuf_reserve+0xc4/0x254 bpf_ringbuf_reserve_dynptr+0x5c/0xdc bpf_prog_ac3d15160d62622a_test_read_write+0x104/0x238 trace_call_bpf+0x238/0x774 perf_call_bpf_enter.isra.0+0x104/0x194 perf_syscall_enter+0x2f8/0x510 trace_sys_enter+0x39c/0x564 syscall_trace_enter+0x220/0x3c0 do_el0_svc+0x138/0x1dc el0_svc+0x54/0x130 el0t_64_sync_handler+0x134/0x150 el0t_64_sync+0x17c/0x180 Switch the spinlock to raw_spinlock_t to avoid this error.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38060	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: bpf: copy_verifier_state() should copy 'loop_entry' field The bpf_verifier_state.loop_entry state should be copied by copy_verifier_state(). Otherwise, .loop_entry values from unrelated states would poison env->cur_state. Additionally, env->stack should not contain any states with .loop_entry != NULL. The states in env->stack are yet to be verified, while .loop_entry is set for states that reached an equivalent state. This means that env->cur_state->loop_entry should always be NULL after pop_stack(). See the selftest in the next commit for an example of the program that is not safe yet is accepted by verifier w/o this fix. This change has some verification performance impact for selftests: File Program Insns (A) Insns (B) Insns (DIFF) States (A) States (B) States (DIFF) ---------- ---------- ---------- ---------- ---------- ---------- arena_htab.bpf.o arena_htab_llvm 717 426 -291 (-40.59%) 57 37 -20 (-35.09%) arena_htab_asm.bpf.o arena_htab_asm 597 445 -152 (-25.46%) 47 37 -10 (-21.28%) arena_list.bpf.o arena_list_del 309 279 -30 (-9.71%) 23 14 -9 (-39.13%) iters.bpf.o iter_subprog_check_stacksafe 155 141 -14 (-9.03%) 15 14 -1 (-6.67%) iters.bpf.o iter_subprog_iters 1094 1003 -91 (-8.32%) 88 83 -5 (-5.68%) iters.bpf.o loop_state_deps2 479 725 +246 (+51.36%) 46 63 +17 (+36.96%) kmem_cache_iter.bpf.o open_coded_iter 63 59 -4 (-6.35%) 7 6 -1 (-14.29%) verifier_bits_iter.bpf.o max_words 92 84 -8 (-8.70%) 8 7 -1 (-12.50%) verifier_iterating_callbacks.bpf.o cond_break2 113 107 -6 (-5.31%) 12 12 +0 (+0.00%) And significant negative impact for sched_ext: File Program Insns (A) Insns (B) Insns (DIFF) States (A) States (B) States (DIFF) ---------- ---------- ---------- ---------- ---------- ---------- bpf.bpf.o lavd_init 7039 14723 +7684 (+109.16%) 490 1139 +649 (+132.45%) bpf.bpf.o layered_dispatch 11485 10548 -937 (-8.16%) 848 762 -86 (-10.14%) bpf.bpf.o layered_dump 7422 1000001 +992579 (+13373.47%) 681 31178 +30497 (+4478.27%) bpf.bpf.o layered_enqueue 16854 71127 +54273 (+322.02%) 1611 6450 +4839 (+300.37%) bpf.bpf.o p2dq_dispatch 665 791 +126 (+18.95%) 68 78 +10 (+14.71%) bpf.bpf.o p2dq_init 2343 2980 +637 (+27.19%) 201 237 +36 (+17.91%) bpf.bpf.o refresh_layer_cpumasks 16487 674760 +658273 (+3992.68%) 1770 65370 +63600 (+3593.22%) bpf.bpf.o rusty_select_cpu 1937 40872 +38935 (+2010.07%) 177 3210 +3033 (+1713.56%) scx_central.bpf.o central_dispatch 636 2687 +2051 (+322.48%) 63 227 +164 (+260.32%) scx_nest.bpf.o nest_init 636 815 +179 (+28.14%) 60 73 +13 (+21.67%) scx_qmap.bpf.o qmap_dispatch ---truncated---
CVE-2025-39800	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: btrfs: abort transaction on unexpected eb generation at btrfs_copy_root() If we find an unexpected generation for the extent buffer we are cloning at btrfs_copy_root(), we just WARN_ON() and don't error out and abort the transaction, meaning we allow to persist metadata with an unexpected generation. Instead of warning only, abort the transaction and return -EUCLEAN.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-37931	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: btrfs: adjust subpage bit start based on sectorsize When running machines with 64k page size and a 16k nodesize we started seeing tree log corruption in production. This turned out to be because we were not writing out dirty blocks sometimes, so this in fact affects all metadata writes. When writing out a subpage EB we scan the subpage bitmap for a dirty range. If the range isn't dirty we do bit_start++; to move onto the next bit. The problem is the bitmap is based on the number of sectors that an EB has. So in this case, we have a 64k pagesize, 16k nodesize, but a 4k sectorsize. This means our bitmap is 4 bits for every node. With a 64k page size we end up with 4 nodes per page. To make this easier this is how everything looks [0 16k 32k 48k] logical address [0 4 8 12] radix tree offset [64k page] folio [16k eb][16k eb][16k eb][16k eb] extent buffers [| | | | | | | | | | | | | | | |] bitmap Now we use all of our addressing based on fs_info->sectorsize_bits, so as you can see the above our 16k eb->start turns into radix entry 4. When we find a dirty range for our eb, we correctly do bit_start += sectors_per_node, because if we start at bit 0, the next bit for the next eb is 4, to correspond to eb->start 16k. However if our range is clean, we will do bit_start++, which will now put us offset from our radix tree entries. In our case, assume that the first time we check the bitmap the block is not dirty, we increment bit_start so now it == 1, and then we loop around and check again. This time it is dirty, and we go to find that start using the following equation start = folio_start + bit_start * fs_info->sectorsize; so in the case above, eb->start 0 is now dirty, and we calculate start as 0 + 1 * fs_info->sectorsize = 4096 4096 >> 12 = 1 Now we're looking up the radix tree for 1, and we won't find an eb. What's worse is now we're using bit_start == 1, so we do bit_start += sectors_per_node, which is now 5. If that eb is dirty we will run into the same thing, we will look at an offset that is not populated in the radix tree, and now we're skipping the writeout of dirty extent buffers. The best fix for this is to not use sectorsize_bits to address nodes, but that's a larger change. Since this is a fs corruption problem fix it simply by always using sectors_per_node to increment the start bit.
CVE-2025-71183	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: btrfs: always detect conflicting inodes when logging inode refs After rename exchanging (either with the rename exchange operation or regular renames in multiple non-atomic steps) two inodes and at least one of them is a directory, we can end up with a log tree that contains only of the inodes and after a power failure that can result in an attempt to delete the other inode when it should not because it was not deleted before the power failure. In some case that delete attempt fails when the target inode is a directory that contains a subvolume inside it, since the log replay code is not prepared to deal with directory entries that point to root items (only inode items). 1) We have directories "dir1" (inode A) and "dir2" (inode B) under the same parent directory; 2) We have a file (inode C) under directory "dir1" (inode A); 3) We have a subvolume inside directory "dir2" (inode B); 4) All these inodes were persisted in a past transaction and we are currently at transaction N; 5) We rename the file (inode C), so at btrfs_log_new_name() we update inode C's last_unlink_trans to N; 6) We get a rename exchange for "dir1" (inode A) and "dir2" (inode B), so after the exchange "dir1" is inode B and "dir2" is inode A. During the rename exchange we call btrfs_log_new_name() for inodes A and B, but because they are directories, we don't update their last_unlink_trans to N; 7) An fsync against the file (inode C) is done, and because its inode has a last_unlink_trans with a value of N we log its parent directory (inode A) (through

CVE	Severity	CVSS	CVSS Vector	Component	Description
					btrfs_log_all_parents(), called from btrfs_log_inode_parent()). 8) So we end up with inode B not logged, which now has the old name of inode A. At copy_inode_items_to_log(), when logging inode A, we did not check if we had any conflicting inode to log because inode A has a generation lower than the current transaction (created in a past transaction); 9) After a power failure, when replaying the log tree, since we find that inode A has a new name that conflicts with the name of inode B in the fs tree, we attempt to delete inode B... this is wrong since that directory was never deleted before the power failure, and because there is a subvolume inside that directory, attempting to delete it will fail since replay_dir_deletes() and btrfs_unlink_inode() are not prepared to deal with dir items that point to roots instead of inodes. When that happens the mount fails and we get a stack trace like the following: [87.2314] BTRFS info (device dm-0): start tree-log replay [87.2318] BTRFS critical (device dm-0): failed to delete reference to subvol, root 5 inode 256 parent 259 [87.2332] ----------[cut here]---------- [87.2338] BTRFS: Transaction aborted (error -2) [87.2346] WARNING: CPU: 1 PID: 638968 at fs/btrfs/inode.c:4345 __btrfs_unlink_inode+0x416/0x440 [btrfs] [87.2368] Modules linked in: btrfs loop dm_thin_pool (...) [87.2470] CPU: 1 UID: 0 PID: 638968 Comm: mount Tainted: G W 6.18.0-rc7-btrfs-next-218+ #2 PREEMPT(full) [87.2489] Tainted: [W]=WARN [87.2494] Hardware name: QEMU Standard PC (i440FX + PIIX, 1996), BIOS rel-1.16.2-0-gea1b7a073390-prebuilt.qemu.org 04/01/2014 [87.2514] RIP: 0010:__btrfs_unlink_inode+0x416/0x440 [btrfs] [87.2538] Code: c0 89 04 24 (...) [87.2568] RSP: 0018:ffffc0e741f4b9b8 EFLAGS: 00010286 [87.2574] RAX: 0000000000000000 RBX: ffff9d3ec8a6cf60 RCX: 0000000000000000 [87.2582] RDX: 0000000000000002 RSI: ffffffff84ab45a1 RDI: 00000000ffffffff [87.2591] RBP: ffff9d3ec8a6ef20 R08: 0000000000000000 R09: ffffc0e741f4b840 [87.2599] R10: ffff9d45dc1fffa8 R11: 0000000000000003 R12: ffff9d3ee26d77e0 [87.2608] R13: ffffc0e741f4ba98 R14: ffff9d4458040800 R15: ffff9d44b6b7ca10 [87.2618] FS: 00007f7b9603a840(0000) GS:ffff9d4658982000(0000) knlGS:0000000000000000 [87. ---truncated---

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-68358	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: btrfs: fix racy bitfield write in btrfs_clear_space_info_full() From the memory-barriers.txt document regarding memory barrier ordering guarantees: (*) These guarantees do not apply to bitfields, because compilers often generate code to modify these using non-atomic read-modify-write sequences. Do not attempt to use bitfields to synchronize parallel algorithms. (*) Even in cases where bitfields are protected by locks, all fields in a given bitfield must be protected by one lock. If two fields in a given bitfield are protected by different locks, the compiler's non-atomic read-modify-write sequences can cause an update to one field to corrupt the value of an adjacent field. btrfs_space_info has a bitfield sharing an underlying word consisting of the fields full, chunk_alloc, and flush: struct btrfs_space_info { struct btrfs_fs_info * fs_info; /* 0 8 */ struct btrfs_space_info * parent; /* 8 8 */ ... int clamp; /* 172 4 */ unsigned int full:1; /* 176: 0 4 */ unsigned int chunk_alloc:1; /* 176: 1 4 */ unsigned int flush:1; /* 176: 2 4 */ ... Therefore, to be safe from parallel read-modify-writes losing a write to one of the bitfield members protected by a lock, all writes to all the bitfields must use the lock. They almost universally do, except for btrfs_clear_space_info_full() which iterates over the space_infos and writes out found->full = 0 without a lock. Imagine that we have one thread completing a transaction in which we finished deleting a block_group and are thus calling btrfs_clear_space_info_full() while simultaneously the data reclaim ticket infrastructure is running do_async_reclaim_data_space(): T1 T2 btrfs_commit_transaction btrfs_clear_space_info_full data_sinfo->full = 0 READ: full:0, chunk_alloc:0, flush:1 do_async_reclaim_data_space(data_sinfo) spin_lock(&space_info->lock); if(list_empty(tickets)) space_info->flush = 0; READ: full: 0, chunk_alloc:0, flush:1 MOD/WRITE: full: 0, chunk_alloc:0, flush:0 spin_unlock(&space_info->lock); return; MOD/WRITE: full:0, chunk_alloc:0, flush:1 and now data_sinfo->flush is 1 but the reclaim worker has exited. This breaks the invariant that flush is 0 iff there is no work queued or running. Once this invariant is violated, future allocations that go into __reserve_bytes() will add tickets to space_info->tickets but will see space_info->flush is set to 1 and not queue the work. After this, they will block forever on the resulting ticket, as it is now impossible to kick the worker again. I also confirmed by looking at the assembly of the affected kernel that it is doing RMW operations. For example, to set the flush (3rd) bit to 0, the assembly is: andb $0xfb,0x60(%rbx) and similarly for setting the full (1st) bit to 0: andb $0xfe,-0x20(%rax) So I think this is really a bug on practical systems. I have observed a number of systems in this exact state, but am currently unable to reproduce it. Rather than leaving this footgun lying around for the future, take advantage of the fact that there is room in the struct anyway, and that it is already quite large and simply change the three bitfield members to bools. This avoids writes to space_info->full having any effect on ---truncated---

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-37856	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: btrfs: harden block_group::bg_list against list_del() races As far as I can tell, these calls of list_del_init() on bg_list cannot run concurrently with btrfs_mark_bg_unused() or btrfs_mark_bg_to_reclaim(), as they are in transaction error paths and situations where the block group is readonly. However, if there is any chance at all of racing with mark_bg_unused(), or a different future user of bg_list, better to be safe than sorry. Otherwise we risk the following interleaving (bg_list refcount in parens) T1 (some random op) T2 (btrfs_mark_bg_unused) !list_empty(&bg->bg_list); (1) list_del_init(&bg->bg_list); (1) list_move_tail (1) btrfs_put_block_group (0) btrfs_delete_unused_bgs bg = list_first_entry list_del_init(&bg->bg_list); btrfs_put_block_group(bg); (-1) Ultimately, this results in a broken ref count that hits zero one deref early and the real final deref underflows the refcount, resulting in a WARNING.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2026-23214	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: btrfs: reject new transactions if the fs is fully read-only [BUG] There is a bug report where a heavily fuzzed fs is mounted with all rescue mount options, which leads to the following warnings during unmount: BTRFS: Transaction aborted (error -22) Modules linked in: CPU: 0 UID: 0 PID: 9758 Comm: repro.out Not tainted 6.19.0-rc5-00002-gb71e635feefc #7 PREEMPT(full) Hardware name: QEMU Standard PC (i440FX + PIIX, 1996), BIOS 1.15.0-1 04/01/2014 RIP: 0010:find_free_extent_update_loop fs/btrfs/extent-tree.c:4208 [inline] RIP: 0010:find_free_extent+0x52f0/0x5d20 fs/btrfs/extent-tree.c:4611 Call Trace: btrfs_reserve_extent+0x2cd/0x790 fs/btrfs/extent-tree.c:4705 btrfs_alloc_tree_block+0x1e1/0x10e0 fs/btrfs/extent-tree.c:5157 btrfs_force_cow_block+0x578/0x2410 fs/btrfs/ctree.c:517 btrfs_cow_block+0x3c4/0xa80 fs/btrfs/ctree.c:708 btrfs_search_slot+0xcad/0x2b50 fs/btrfs/ctree.c:2130 btrfs_truncate_inode_items+0x45d/0x2350 fs/btrfs/inode-item.c:499 btrfs_evict_inode+0x923/0xe70 fs/btrfs/inode.c:5628 evict+0x5f4/0xae0 fs/inode.c:837 __dentry_kill+0x209/0x660 fs/dcache.c:670 finish_dput+0xc9/0x480 fs/dcache.c:879 shrink_dcache_for_umount+0xa0/0x170 fs/dcache.c:1661 generic_shutdown_super+0x67/0x2c0 fs/super.c:621 kill_anon_super+0x3b/0x70 fs/super.c:1289 btrfs_kill_super+0x41/0x50 fs/btrfs/super.c:2127 deactivate_locked_super+0xbc/0x130 fs/super.c:474 cleanup_mnt+0x425/0x4c0 fs/namespace.c:1318 task_work_run+0x1d4/0x260 kernel/task_work.c:233 exit_task_work include/linux/task_work.h:40 [inline] do_exit+0x694/0x22f0 kernel/exit.c:971 do_group_exit+0x21c/0x2d0 kernel/exit.c:1112 __do_sys_exit_group kernel/exit.c:1123 [inline] __se_sys_exit_group kernel/exit.c:1121 [inline] __x64_sys_exit_group+0x3f/0x40 kernel/exit.c:1121 x64_sys_call+0x2210/0x2210 arch/x86/include/generated/asm/syscalls_64.h:232 do_syscall_x64 arch/x86/entry/syscall_64.c:63 [inline] do_syscall_64+0xe8/0xf80 arch/x86/entry/syscall_64.c:94 entry_SYSCALL_64_after_hwframe+0x77/0x7f RIP: 0033:0x44f639 Code: Unable to access opcode bytes at 0x44f60f. RSP: 002b:00007ffc15c4e088 EFLAGS: 00000246 ORIG_RAX: 00000000000000e7 RAX: ffffffffffffffda RBX: 00000000004c32f0 RCX: 000000000044f639 RDX: 000000000000003c RSI: 00000000000000e7 RDI: 0000000000000001 RBP: 0000000000000001 R08: ffffffffffffffc0 R09: 0000000000000000 R10: 0000000000000000 R11: 0000000000000246 R12: 00000000004c32f0 R13: 0000000000000001 R14: 0000000000000000 R15: 0000000000000001 Since rescue mount options will mark the full fs read-only, there should be no new transaction triggered. But during unmount we will evict all inodes, which can trigger a new transaction, and triggers warnings on a heavily corrupted fs. [CAUSE] Btrfs allows new transaction even on a read-only fs, this is to allow log replay happen even on read-only mounts, just like what ext4/xfs do. However with rescue mount options, the fs is fully read-only and cannot be remounted read-write, thus in that case we should also reject any new transactions. [FIX] If we find the fs has rescue mount options, we should treat the fs as error, so that no new transaction can be started.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2024-41067	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: btrfs: scrub: handle RST lookup error correctly [BUG] When running btrfs/060 with forced RST feature, it would crash the following ASSERT() inside scrub_read_endio(): ASSERT(sector_nr < stripe->nr_sectors); Before that, we would have tree dump from btrfs_get_raid_extent_offset(), as we failed to find the RST entry for the range. [CAUSE] Inside scrub_submit_extent_sector_read() every time we allocated a new bbio we immediately called btrfs_map_block() to make sure there was some RST range covering the scrub target. But if btrfs_map_block() fails, we immediately call endio for the bbio, while the bbio is newly allocated, it's completely empty. Then inside scrub_read_endio(), we go through the bvecs to find the sector number (as bi_sector is no longer reliable if the bio is submitted to lower layers). And since the bio is empty, such bvecs iteration would not find any sector matching the sector, and return sector_nr == stripe->nr_sectors, triggering the ASSERT(). [FIX] Instead of calling btrfs_map_block() after allocating a new bbio, call btrfs_map_block() first. Since our only objective of calling btrfs_map_block() is only to update stripe_len, there is really no need to do that after btrfs_alloc_bio(). This new timing would avoid the problem of handling empty bbio completely, and in fact fixes a possible race window for the old code, where if the submission thread is the only owner of the pending_io, the scrub would never finish (since we didn't decrease the pending_io counter). Although the root cause of RST lookup failure still needs to be addressed.
CVE-2026-23141	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: btrfs: send: check for inline extents in range_is_hole_in_parent() Before accessing the disk_bytenr field of a file extent item we need to check if we are dealing with an inline extent. This is because for inline extents their data starts at the offset of the disk_bytenr field. So accessing the disk_bytenr means we are accessing inline data or in case the inline data is less than 8 bytes we can actually cause an invalid memory access if this inline extent item is the first item in the leaf or access metadata from other items.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-39779	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: btrfs: subpage: keep TOWRITE tag until folio is cleaned btrfs_subpage_set_writeback() calls folio_start_writeback() the first time a folio is written back, and it also clears the PAGECACHE_TAG_TOWRITE tag even if there are still dirty blocks in the folio. This can break ordering guarantees, such as those required by btrfs_wait_ordered_extents(). That ordering breakage leads to a real failure. For example, running generic/464 on a zoned setup will hit the following ASSERT. This happens because the broken ordering fails to flush existing dirty pages before the file size is truncated. assertion failed: !list_empty(&ordered->list) :: 0, in fs/btrfs/zoned.c:1899 ------------[cut here]------------ kernel BUG at fs/btrfs/zoned.c:1899! Oops: invalid opcode: 0000 [#1] SMP NOPTI CPU: 2 UID: 0 PID: 1906169 Comm: kworker/u130:2 Kdump: loaded Not tainted 6.16.0-rc6-BTRFS-ZNS+ #554 PREEMPT(voluntary) Hardware name: Supermicro Super Server/H12SSL-NT, BIOS 2.0 02/22/2021 Workqueue: btrfs-endio-write btrfs_work_helper [btrfs] RIP: 0010:btrfs_finish_ordered_zoned.cold+0x50/0x52 [btrfs] RSP: 0018:ffffc9002efdbd60 EFLAGS: 00010246 RAX: 000000000000004c RBX: ffff88811923c4e0 RCX: 0000000000000000 RDX: 0000000000000000 RSI: ffffffff827e38b1 RDI: 00000000ffffffff RBP: ffff88810005d000 R08: 00000000ffffdfff R09: ffffffff831051c8 R10: ffffffff83055220 R11: 0000000000000000 R12: ffff8881c2458c00 R13: ffff88811923c540 R14: ffff88811923c5e8 R15: ffff8881c1bd9680 FS: 0000000000000000(0000) GS:ffff88a04acd0000(0000) knlGS:0000000000000000 CS: 0010 DS: 0000 ES: 0000 CR0: 0000000080050033 CR2: 00007f907c7a918c CR3: 0000000004024000 CR4: 0000000000350ef0 Call Trace: ? srso_return_thunk+0x5/0x5f btrfs_finish_ordered_io+0x4a/0x60 [btrfs] btrfs_work_helper+0xf9/0x490 [btrfs] process_one_work+0x204/0x590 ? srso_return_thunk+0x5/0x5f worker_thread+0x1d6/0x3d0 ? __pfx_worker_thread+0x10/0x10 kthread+0x118/0x230 ? __pfx_kthread+0x10/0x10 ret_from_fork+0x205/0x260 ? __pfx_kthread+0x10/0x10 ret_from_fork_asm+0x1a/0x30 Consider process A calling writepages() with WB_SYNC_NONE. In zoned mode or for compressed writes, it locks several folios for delalloc and starts writing them out. Let's call the last locked folio folio X. Suppose the write range only partially covers folio X, leaving some pages dirty. Process A calls btrfs_subpage_set_writeback() when building a bio. This function call clears the TOWRITE tag of folio X, whose size = 8K and the block size = 4K. It is following state. 0 4K 8K |/////|/////| (flag: DIRTY, tag: DIRTY) <------> Process A will write this range. Now suppose process B concurrently calls writepages() with WB_SYNC_ALL. It calls tag_pages_for_writeback() to tag dirty folios with PAGECACHE_TAG_TOWRITE. Since folio X is still dirty, it gets tagged. Then, B collects tagged folios using filemap_get_folios_tag() and must wait for folio X to be written before returning from writepages(). 0 4K 8K |/////|/////| (flag: DIRTY, tag: DIRTY|TOWRITE) However, between tagging and collecting, process A may call btrfs_subpage_set_writeback() and clear folio X's TOWRITE tag. 0 4K 8K	|/////| (flag: DIRTY|WRITEBACK, tag: DIRTY) As a result, process B won't see folio X in its batch, and returns without waiting for it. This breaks the WB_SYNC_ALL ordering requirement. Fix this by using btrfs_subpage_set_writeback_keepwrite(), which retains the TOWRITE tag. We now manually clear the tag only after the folio becomes clean, via the xas operation.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-39925	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: can: j1939: implement NETDEV_UNREGISTER notification handler syzbot is reporting unregister_netdevice: waiting for vcan0 to become free. Usage count = 2 problem, for j1939 protocol did not have NETDEV_UNREGISTER notification handler for undoing changes made by j1939_sk_bind(). Commit 25fe97cb7620 ("can: j1939: move j1939_priv_put() into sk_destruct callback") expects that a call to j1939_priv_put() can be unconditionally delayed until j1939_sk_sock_destruct() is called. But we need to call j1939_priv_put() against an extra ref held by j1939_sk_bind() call (as a part of undoing changes made by j1939_sk_bind()) as soon as NETDEV_UNREGISTER notification fires (i.e. before j1939_sk_sock_destruct() is called via j1939_sk_release()). Otherwise, the extra ref on "struct j1939_priv" held by j1939_sk_bind() call prevents "struct net_device" from dropping the usage count to 1; making it impossible for unregister_netdevice() to continue. [mkl: remove space in front of label]
CVE-2025-71182	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: can: j1939: make j1939_session_activate() fail if device is no longer registered syzbot is still reporting unregister_netdevice: waiting for vcan0 to become free. Usage count = 2 even after commit 93a27b5891b8 ("can: j1939: add missing calls in NETDEV_UNREGISTER notification handler") was added. A debug printk() patch found that j1939_session_activate() can succeed even after j1939_cancel_active_session() from j1939_netdev_notify(NETDEV_UNREGISTER) has completed. Since j1939_cancel_active_session() is processed with the session list lock held, checking ndev->reg_state in j1939_session_activate() with the session list lock held can reliably close the race window.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-39953	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: cgroup: split cgroup_destroy_wq into 3 workqueues A hung task can occur during [1] LTP cgroup testing when repeatedly mounting/unmounting perf_event and net_prio controllers with systemd.unified_cgroup_hierarchy=1. The hang manifests in cgroup_lock_and_drain_offline() during root destruction. Related case: cgroup_fj_function_perf_event cgroup_fj_function.sh perf_event cgroup_fj_function_net_prio cgroup_fj_function.sh net_prio Call Trace: cgroup_lock_and_drain_offline+0x14c/0x1e8 cgroup_destroy_root+0x3c/0x2c0 css_free_rwork_fn+0x248/0x338 process_one_work+0x16c/0x3b8 worker_thread+0x22c/0x3b0 kthread+0xec/0x100 ret_from_fork+0x10/0x20 Root Cause: CPU0 CPU1 mount perf_event umount net_prio cgroup1_get_tree cgroup_kill_sb rebind_subsystems // root destruction enqueues // cgroup_destroy_wq // kill all perf_event css // one perf_event css A is dying // css A offline enqueues cgroup_destroy_wq // root destruction will be executed first css_free_rwork_fn cgroup_destroy_root cgroup_lock_and_drain_offline // some perf descendants are dying // cgroup_destroy_wq max_active = 1 // waiting for css A to die Problem scenario: 1. CPU0 mounts perf_event (rebind_subsystems) 2. CPU1 unmounts net_prio (cgroup_kill_sb), queuing root destruction work 3. A dying perf_event CSS gets queued for offline after root destruction 4. Root destruction waits for offline completion, but offline work is blocked behind root destruction in cgroup_destroy_wq (max_active=1) Solution: Split cgroup_destroy_wq into three dedicated workqueues: cgroup_offline_wq – Handles CSS offline operations cgroup_release_wq – Manages resource release cgroup_free_wq – Performs final memory deallocation This separation eliminates blocking in the CSS free path while waiting for offline operations to complete. [1] https://github.com/linux-test-project/ltp/blob/master/runtest/controllers
CVE-2025-38041	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: clk: sunxi-ng: h616: Reparent GPU clock during frequency changes The H616 manual does not state that the GPU PLL supports dynamic frequency configuration, so we must take extra care when changing the frequency. Currently any attempt to do device DVFS on the GPU lead to panfrost various ooops, and GPU hangs. The manual describes the algorithm for changing the PLL frequency, which the CPU PLL notifier code already support, so we reuse that to reparent the GPU clock to GPU1 clock during frequency changes.
CVE-2025-38499	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: clone_private_mnt(): make sure that caller has CAP_SYS_ADMIN in the right userns What we want is to verify there is that clone won't expose something hidden by a mount we wouldn't be able to undo. "Wouldn't be able to undo" may be a result of MNT_LOCKED on a child, but it may also come from lacking admin rights in the userns of the namespace mount belongs to. clone_private_mnt() checks the former, but not the latter. There's a number of rather confusing CAP_SYS_ADMIN checks in various userns during the mount, especially with the new mount API; they serve different purposes and in case of clone_private_mnt() they usually, but not always end up covering the missing check mentioned above.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38481	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: comedi: Fail COMEDI_INSNLIST ioctl if n_insns is too large The handling of the `COMEDI_INSNLIST` ioctl allocates a kernel buffer to hold the array of `struct comedi_insn`, getting the length from the `n_insns` member of the `struct comedi_insnlist` supplied by the user. The allocation will fail with a WARNING and a stack dump if it is too large. Avoid that by failing with an `-EINVAL` error if the supplied `n_insns` value is unreasonable. Define the limit on the `n_insns` value in the `MAX_INSNS` macro. Set this to the same value as `MAX_SAMPLES` (65536), which is the maximum allowed sum of the values of the member `n` in the array of `struct comedi_insn`, and sensible comedi instructions will have an `n` of at least 1.
CVE-2025-38132	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: coresight: holding cscfg_csdev_lock while removing cscfg from csdev There'll be possible race scenario for coresight config: CPU0 CPU1 (perf enable) load module cscfg_load_config_sets() activate config. // sysfs (sys_active_cnt == 1) ... cscfg_csdev_enable_active_config() lock(csdev->cscfg_csdev_lock) deactivate config // sysfs (sys_activec_cnt == 0) cscfg_unload_config_sets() cscfg_remove_owned_csdev_configs() // here load config activate by CPU1 unlock(csdev->cscfg_csdev_lock) iterating config_csdev_list could be raced with config_csdev_list's entry delete. To resolve this race , hold csdev->cscfg_csdev_lock() while cscfg_remove_owned_csdev_configs()
CVE-2025-71180	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: counter: interrupt-cnt: Drop IRQF_NO_THREAD flag An IRQ handler can either be IRQF_NO_THREAD or acquire spinlock_t, as CONFIG_PROVE_RAW_LOCK_NESTING warns: ============================= [BUG: Invalid wait context] 6.18.0-rc1+git... #1 ----------------------------- some-user-space-process/1251 is trying to lock: (&counter->events_list_lock){....}-{3:3}, at: counter_push_event [counter] other info that might help us debug this: context-{2:2} no locks held by some-user-space-process/.... stack backtrace: CPU: 0 UID: 0 PID: 1251 Comm: some-user-space-process 6.18.0-rc1+git... #1 PREEMPT Call trace: show_stack (C) dump_stack_lvl dump_stack __lock_acquire lock_acquire _raw_spin_lock_irqsave counter_push_event [counter] interrupt_cnt_isr [interrupt_cnt] __handle_irq_event_percpu handle_irq_event handle_simple_irq handle_irq_desc generic_handle_domain_irq gpio_irq_handler handle_irq_desc generic_handle_domain_irq gic_handle_irq call_on_irq_stack do_interrupt_handler el0_interrupt __el0_irq_handler_common el0t_64_irq_handler el0t_64_irq ... and Sebastian correctly points out. Remove IRQF_NO_THREAD as an alternative to switching to raw_spinlock_t, because the latter would limit all potential nested locks to raw_spinlock_t only.
CVE-2025-38038	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: cpufreq: amd-pstate: Remove unnecessary driver_lock in set_boost set_boost is a per-policy function call, hence a driver wide lock is unnecessary. Also this mutex_acquire can collide with the mutex_acquire from the mode-switch path in status_store(), which can lead to a deadlock. So, remove it.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2024-50178	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: cpufreq: loongson3: Use raw_smp_processor_id() in do_service_request() Use raw_smp_processor_id() instead of plain smp_processor_id() in do_service_request(), otherwise we may get some errors with the driver enabled: BUG: using smp_processor_id() in preemptible [00000000] code: (udev-worker)/208 caller is loongson3_cpufreq_probe+0x5c/0x250 [loongson3_cpufreq]
CVE-2026-23222	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: crypto: omap - Allocate OMAP_CRYPTO_FORCE_COPY scatterlists correctly The existing allocation of scatterlists in omap_crypto_copy_sg_lists() was allocating an array of scatterlist pointers, not scatterlist objects, resulting in a 4x too small allocation. Use sizeof(*new_sg) to get the correct object size.
CVE-2025-71131	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: crypto: seqiv - Do not use req->iv after crypto_aead_encrypt As soon as crypto_aead_encrypt is called, the underlying request may be freed by an asynchronous completion. Thus dereferencing req->iv after it returns is invalid. Instead of checking req->iv against info, create a new variable unaligned_info and use it for that purpose instead.
CVE-2026-23229	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: crypto: virtio - Add spinlock protection with virtqueue notification When VM boots with one virtio-crypto PCI device and builtin backend, run openssl benchmark command with multiple processes, such as openssl speed -evp aes-128-cbc -engine afalg -seconds 10 -multi 32 openssl processes will hangup and there is error reported like this: virtio_crypto virtio0: dataq.0:id 3 is not a head! It seems that the data virtqueue need protection when it is handled for virtio done notification. If the spinlock protection is added in virtcrypto_done_task(), openssl benchmark with multiple processes works well.
CVE-2025-39789	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: crypto: x86/aegis - Add missing error checks The skcipher_walk functions can allocate memory and can fail, so checking for errors is necessary.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38496	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: dm-bufio: fix sched in atomic context If "try_verify_in_tasklet" is set for dm-verity, DM_BUFIO_CLIENT_NO_SLEEP is enabled for dm-bufio. However, when bufio tries to evict buffers, there is a chance to trigger scheduling in spin_lock_bh, the following warning is hit: BUG: sleeping function called from invalid context at drivers/md/dm-bufio.c:2745 in_atomic(): 1, irqs_disabled(): 0, non_block: 0, pid: 123, name: kworker/2:2 preempt_count: 201, expected: 0 RCU nest depth: 0, expected: 0 4 locks held by kworker/2:2/123: #0: ffff88800a2d1548 ((wq_completion)dm_bufio_cache){....}-{0:0}, at: process_one_work+0xe46/0x1970 #1: ffffc90000d97d20 ((work_completion)(&dm_bufio_replacement_work)){....}-{0:0}, at: process_one_work+0x763/0x1970 #2: ffffffff8555b528 (dm_bufio_clients_lock){....}-{3:3}, at: do_global_cleanup+0x1ce/0x710 #3: ffff88801d5820b8 (&c->spinlock){....}-{2:2}, at: do_global_cleanup+0x2a5/0x710 Preemption disabled at: [<0000000000000000>] 0x0 CPU: 2 UID: 0 PID: 123 Comm: kworker/2:2 Not tainted 6.16.0-rc3-g90548c634bd0 #305 PREEMPT(voluntary) Hardware name: QEMU Standard PC (i440FX + PIIX, 1996), BIOS rel-1.16.3-0-ga6ed6b701f0a-prebuilt.qemu.org 04/01/2014 Workqueue: dm_bufio_cache do_global_cleanup Call Trace: dump_stack_lvl+0x53/0x70 __might_resched+0x360/0x4e0 do_global_cleanup+0x2f5/0x710 process_one_work+0x7db/0x1970 worker_thread+0x518/0xea0 kthread+0x359/0x690 ret_from_fork+0xf3/0x1b0 ret_from_fork_asm+0x1a/0x30 That can be reproduced by: veritysetup format --data-block-size=4096 --hash-block-size=4096 /dev/vda /dev/vdb SIZE=$(blockdev --getsz /dev/vda) dmsetup create myverity -r --table "0 $SIZE verity 1 /dev/vda /dev/vdb 4096 4096 1 sha256 1 try_verify_in_tasklet" mount /dev/dm-0 /mnt -o ro echo 102400 > /sys/module/dm_bufio/parameters/max_cache_size_bytes [read files in /mnt]

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2024-35794	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: dm-raid: really frozen sync_thread during suspend 1) commit f52f5c71f3d4 ("md: fix stopping sync thread") remove MD_RECOVERY_FROZEN from __md_stop_writes() and doesn't realize that dm-raid relies on __md_stop_writes() to frozen sync_thread indirectly. Fix this problem by adding MD_RECOVERY_FROZEN in md_stop_writes(), and since stop_sync_thread() is only used for dm-raid in this case, also move stop_sync_thread() to md_stop_writes(). 2) The flag MD_RECOVERY_FROZEN doesn't mean that sync thread is frozen, it only prevent new sync_thread to start, and it can't stop the running sync thread; In order to frozen sync_thread, after seting the flag, stop_sync_thread() should be used. 3) The flag MD_RECOVERY_FROZEN doesn't mean that writes are stopped, use it as condition for md_stop_writes() in raid_postsuspend() doesn't look correct. Consider that reentrant stop_sync_thread() do nothing, always call md_stop_writes() in raid_postsuspend(). 4) raid_message can set/clear the flag MD_RECOVERY_FROZEN at anytime, and if MD_RECOVERY_FROZEN is cleared while the array is suspended, new sync_thread can start unexpected. Fix this by disallow raid_message() to change sync_thread status during suspend. Note that after commit f52f5c71f3d4 ("md: fix stopping sync thread"), the test shell/lvconvert-raid-reshape.sh start to hang in stop_sync_thread(), and with previous fixes, the test won't hang there anymore, however, the test will still fail and complain that ext4 is corrupted. And with this patch, the test won't hang due to stop_sync_thread() or fail due to ext4 is corrupted anymore. However, there is still a deadlock related to dm-raid456 that will be fixed in following patches.
CVE-2025-38140	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: dm: limit swapping tables for devices with zone write plugs dm_revalidate_zones() only allowed new or previously unzoned devices to call blk_revalidate_disk_zones(). If the device was already zoned, disk->nr_zones would always equal md->nr_zones, so dm_revalidate_zones() returned without doing any work. This would make the zoned settings for the device not match the new table. If the device had zone write plug resources, it could run into errors like bdev_zone_is_seq() reading invalid memory because disk->conv_zones_bitmap was the wrong size. If the device doesn't have any zone write plug resources, calling blk_revalidate_disk_zones() will always correctly update device. If blk_revalidate_disk_zones() fails, it can still overwrite or clear the current disk->nr_zones value. In this case, DM must restore the previous value of disk->nr_zones, so that the zoned settings will continue to match the previous value that it fell back to. If the device already has zone write plug resources, blk_revalidate_disk_zones() will not correctly update them, if it is called for arbitrary zoned device changes. Since there is not much need for this ability, the easiest solution is to disallow any table reloads that change the zoned settings, for devices that already have zone plug resources. Specifically, if a device already has zone plug resources allocated, it can only switch to another zoned table that also emulates zone append. Also, it cannot change the device size or the zone size. A device can switch to an error target.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-71191	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: dmaengine: at_hdmac: fix device leak on of_dma_xlate() Make sure to drop the reference taken when looking up the DMA platform device during of_dma_xlate() when releasing channel resources. Note that commit 3832b78b3ec2 ("dmaengine: at_hdmac: add missing put_device() call in at_dma_xlate()") fixed the leak in a couple of error paths but the reference is still leaking on successful allocation.
CVE-2025-71190	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: dmaengine: bcm-sba-raid: fix device leak on probe Make sure to drop the reference taken when looking up the mailbox device during probe on probe failures and on driver unbind.
CVE-2025-39923	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: dmaengine: qcom: bam_dma: Fix DT error handling for num-channels/ees When we don't have a clock specified in the device tree, we have no way to ensure the BAM is on. This is often the case for remotely-controlled or remotely-powered BAM instances. In this case, we need to read num-channels from the DT to have all the necessary information to complete probing. However, at the moment invalid device trees without clock and without num-channels still continue probing, because the error handling is missing return statements. The driver will then later try to read the number of channels from the registers. This is unsafe, because it relies on boot firmware and lucky timing to succeed. Unfortunately, the lack of proper error handling here has been abused for several Qualcomm SoCs upstream, causing early boot crashes in several situations [1, 2]. Avoid these early crashes by erroring out when any of the required DT properties are missing. Note that this will break some of the existing DTs upstream (mainly BAM instances related to the crypto engine). However, clearly these DTs have never been tested properly, since the error in the kernel log was just ignored. It's safer to disable the crypto engine for these broken DTBs. [1]: https://lore.kernel.org/r/CY01EKQVWE36.B9X5TDXAREPF@fairphone.com/ [2]: https://lore.kernel.org/r/20230626145959.646747-1-krzysztof.kozlowski@linaro.org/
CVE-2025-38042	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: dmaengine: ti: k3-udma-glue: Drop skip_fdq argument from k3_udma_glue_reset_rx_chn The user of k3_udma_glue_reset_rx_chn() e.g. ti_am65_cpsw_nuss can run on multiple platforms having different DMA architectures. On some platforms there can be one FDQ for all flows in the RX channel while for others there is a separate FDQ for each flow in the RX channel. So far we have been relying on the skip_fdq argument of k3_udma_glue_reset_rx_chn(). Instead of relying on the user to provide this information, infer it based on DMA architecture during k3_udma_glue_request_rx_chn() and save it in an internal flag 'single_fdq'. Use that flag at k3_udma_glue_reset_rx_chn() to deicide if the FDQ needs to be cleared for every flow or just for flow 0. Fixes the below issue on ti_am65_cpsw_nuss driver on AM62-SK. > ip link set eth1 down > ip link set eth0 down > ethtool -L eth0 rx 8 > ip link set eth0 up > modprobe -r ti_am65_cpsw_nuss [103.045726] ------------[cut here]------------ [103.050505] k3_knav_desc_pool size 512000 != avail 64000 [103.050703] WARNING: CPU: 1 PID: 450 at drivers/net/ethernet/ti/k3-cppi-desc-pool.c:33 k3_cppi_desc_pool_destroy+0xa0/0xa8 [k3_cppi_desc_pool] [103.068810] Modules linked in: ti_am65_cpsw_nuss(-) k3_cppi_desc_pool snd_soc_hdmi_codec crct10dif_ce snd_soc_simple_card snd_soc_simple_card_utils display_connector rtc_ti_k3 k3_j72xx_bandgap tidss drm_client_lib

CVE	Severity	CVSS	CVSS Vector	Component	Description
					snd_soc_davinci_mcasp drm_dma_helper tps6598x phylink snd_soc_ti_udma rti_wdt drm_display_helper snd_soc_tlv320aic3x_i2c typec at24 phy_gmii_sel snd_soc_ti_edma snd_soc_tlv320aic3x sii902x snd_soc_ti_sdma sa2ul omap_mailbox drm_kms_helper authenc cfg80211 rfkill fuse drm drm_panel_orientation_quirks backlight ip_tables x_tables ipv6 [last unloaded: k3_cppi_desc_pool] [103.119950] CPU: 1 UID: 0 PID: 450 Comm: modprobe Not tainted 6.13.0-rc7-00001-g9c5e3435fa66 #1011 [103.119968] Hardware name: Texas Instruments AM625 SK (DT) [103.119974] pstate: 80000005 (Nzcv daif -PAN -UAO -TCO -DIT -SSBS BTYPE=--) [103.119983] pc : k3_cppi_desc_pool_destroy+0xa0/0xa8 [k3_cppi_desc_pool] [103.148007] lr : k3_cppi_desc_pool_destroy+0xa0/0xa8 [k3_cppi_desc_pool] [103.154709] sp : ffff8000826ebbc0 [103.158015] x29: ffff8000826ebbc0 x28: ffff0000090b6300 x27: 0000000000000000 [103.165145] x26: 0000000000000000 x25: 0000000000000000 x24: ffff0000019df6b0 [103.172271] x23: ffff0000019df6b8 x22: ffff0000019df410 x21: ffff8000826ebc88 [103.179397] x20: 000000000007d000 x19: ffff00000a3b3000 x18: 0000000000000000 [103.186522] x17: 0000000000000000 x16: 0000000000000000 x15: 000001e8c35e1cde [103.193647] x14: 0000000000000396 x13: 000000000000035c x12: 0000000000000000 [103.200772] x11: 000000000000003a x10: 00000000000009c0 x9 : ffff8000826eba20 [103.207897] x8 : ffff0000090b6d20 x7 : ffff00007728c180 x6 : ffff00007728c100 [103.215022] x5 : 0000000000000001 x4 : ffff000000508a50 x3 : ffff7ffff6146000 [103.222147] x2 : 0000000000000000 x1 : e300b4173ee6b200 x0 : 0000000000000000 [103.229274] Call trace: [103.231714] k3_cppi_desc_pool_destroy+0xa0/0xa8 [k3_cppi_desc_pool] (P) [103.238408] am65_cpsw_nuss_free_rx_chns+0x28/0x4c [ti_am65_cpsw_nuss] [103.244942] devm_action_release+0x14/0x20 [103.249040] release_nodes+0x3c/0x68 [103.252610] devres_release_all+0x8c/0xdc [103.256614] device_unbind_cleanup+0x18/0x60 [103.260876] device_release_driver_internal+0xf8/0x178 [103.266004] driver_detach+0x50/0x9c [103.269571] bus_remove_driver+0x6c/0xbc [103.273485] driver_unregister+0x30/0x60 [103.277401] platform_driver_unregister+0x14/0x20 [103.282096] am65_cpsw_nuss_driver_exit+0x18/0xff4 [ti_am65_cpsw_nuss] [103.288620] __arm64_sys_delete_module+0x17c/0x25c [103.293404] invoke_syscall+0x44/0x100 [103.297149] el0_svc_common.constprop.0+0xc0/0xe0 [103.301845] do_el0_svc+0x1c/0x28 [103.305155] el0_svc+0x28/0x98 ---truncated---
CVE-2025-37870	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: drm/amd/display: prevent hang on link training fail [Why] When link training fails, the phy clock will be disabled. However, in enable_streams, it is assumed that link training succeeded and the mux selects the phy clock, causing a hang when a register write is made. [How] When enable_stream is hit, check if link training failed. If it did, fall back to the ref clock to avoid a hang and keep the system in a recoverable state.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2026-23213	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: drm/amd/pm: Disable MMIO access during SMU Mode 1 reset During Mode 1 reset, the ASIC undergoes a reset cycle and becomes temporarily inaccessible via PCIe. Any attempt to access MMIO registers during this window (e.g., from interrupt handlers or other driver threads) can result in uncompleted PCIe transactions, leading to NMI panics or system hangs. To prevent this, set the `no_hw_access` flag to true immediately after triggering the reset. This signals other driver components to skip register accesses while the device is offline. A memory barrier `smp_mb()` is added to ensure the flag update is globally visible to all cores before the driver enters the sleep/wait state. (cherry picked from commit 7edb503fe4b6d67f47d8bb0dfafb8e699bb0f8a4)
CVE-2025-38426	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: drm/amdgpu: Add basic validation for RAS header If RAS header read from EEPROM is corrupted, it could result in trying to allocate huge memory for reading the records. Add some validation to header fields.
CVE-2024-35931	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: drm/amdgpu: Skip do PCI error slot reset during RAS recovery Why: The PCI error slot reset maybe triggered after inject ue to UMC multi times, this caused system hang. [557.371857] amdgpu 0000:af:00.0: amdgpu: GPU reset succeeded, trying to resume [557.373718] [drm] PCIE GART of 512M enabled. [557.373722] [drm] PTB located at 0x0000031FED700000 [557.373788] [drm] VRAM is lost due to GPU reset! [557.373789] [drm] PSP is resuming... [557.547012] mlx5_core 0000:55:00.0: mlx5_pci_err_detected Device state = 1 pci_status: 0. Exit, result = 3, need reset [557.547067] [drm] PCI error: detected callback, state(1)!! [557.547069] [drm] No support for XGMI hive yet... [557.548125] mlx5_core 0000:55:00.0: mlx5_pci_slot_reset Device state = 1 pci_status: 0. Enter [557.607763] mlx5_core 0000:55:00.0: wait vital counter value 0x16b5b after 1 iterations [557.607777] mlx5_core 0000:55:00.0: mlx5_pci_slot_reset Device state = 1 pci_status: 1. Exit, err = 0, result = 5, recovered [557.610492] [drm] PCI error: slot reset callback!! ... [560.689382] amdgpu 0000:3f:00.0: amdgpu: GPU reset(2) succeeded! [560.689546] amdgpu 0000:5a:00.0: amdgpu: GPU reset(2) succeeded! [560.689562] general protection fault, probably for non-canonical address 0x5f080b54534f611f: 0000 [#1] SMP NOPTI [560.701008] CPU: 16 PID: 2361 Comm: kworker/u448:9 Tainted: G OE 5.15.0-91-generic #101-Ubuntu [560.712057] Hardware name: Microsoft C278A/C278A, BIOS C2789.5.BS.1C11.AG.1 11/08/2023 [560.720959] Workqueue: amdgpu-reset-hive amdgpu_ras_do_recovery [amdgpu] [560.728887] RIP: 0010:amdgpu_device_gpu_recover.cold+0xbf1/0xcf5 [amdgpu] [560.736891] Code: ff 41 89 c6 e9 1b ff ff ff 44 0f b6 45 b0 e9 4f ff ff ff be 01 00 00 00 4c 89 e7 e8 76 c9 8b ff 44 0f b6 45 b0 e9 3c fd ff ff <48> 83 ba 18 02 00 00 00 0f 84 6a f8 ff ff 48 8d 7a 78 be 01 00 00 [560.757967] RSP: 0018:ffa0000032e53d80 EFLAGS: 00010202 [560.763848] RAX: ffa00000001dfd10 RBX: ffa0000000197090 RCX: ffa0000032e53db0 [560.771856] RDX: 5f080b54534f5f07 RSI: 0000000000000000 RDI: ff11000128100010 [560.779867] RBP: ffa0000032e53df0 R08: 0000000000000000 R09: ffffffffffe77f08 [560.787879] R10: 000000000ffff0a R11: 0000000000000001 R12: 0000000000000000 [560.795889] R13: ffa0000032e53e00 R14: 0000000000000000 R15: 0000000000000000 [560.803889] FS: 0000000000000000(0000) GS:ff11007e7e800000(0000) knlGS:0000000000000000 [560.812973] CS: 0010 DS: 0000 ES: 0000 CR0:

CVE	Severity	CVSS	CVSS Vector	Component	Description
					0000000080050033 [560.819422] CR2: 000055a04c118e68 CR3: 0000000007410005 CR4: 0000000000771ee0 [560.827433] DR0: 0000000000000000 DR1: 0000000000000000 DR2: 0000000000000000 [560.835433] DR3: 0000000000000000 DR6: 00000000fffe07f0 DR7: 0000000000000400 [560.843444] PKRU: 55555554 [560.846480] Call Trace: [560.849225] [560.851580] ? show_trace_log_lvl+0x1d6/0x2ea [560.856488] ? show_trace_log_lvl+0x1d6/0x2ea [560.861379] ? amdgpu_ras_do_recovery+0x1b2/0x210 [amdgpu] [560.867778] ? show_regs.part.0+0x23/0x29 [560.872293] ? __die_body.cold+0x8/0xd [560.876502] ? die_addr+0x3e/0x60 [560.880238] ? exc_general_protection+0x1c5/0x410 [560.885532] ? asm_exc_general_protection+0x27/0x30 [560.891025] ? amdgpu_device_gpu_recover.cold+0xbf1/0xcf5 [amdgpu] [560.898323] amdgpu_ras_do_recovery+0x1b2/0x210 [amdgpu] [560.904520] process_one_work+0x228/0x3d0 How: In RAS recovery, mode-1 reset is issued from RAS fatal error handling and expected all the nodes in a hive to be reset. no need to issue another mode-1 during this procedure.
CVE-2024-41008	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: drm/amdgpu: change vm->task_info handling This patch changes the handling and lifecycle of vm->task_info object. The major changes are: - vm->task_info is a dynamically allocated ptr now, and its uasge is reference counted. - introducing two new helper funcs for task_info lifecycle management - amdgpu_vm_get_task_info: reference counts up task_info before returning this info - amdgpu_vm_put_task_info: reference counts down task_info - last put to task_info() frees task_info from the vm. This patch also does logistical changes required for existing usage of vm->task_info. V2: Do not block all the prints when task_info not found (Felix) V3: Fixed review comments from Felix - Fix wrong indentation - No debug message for -ENOMEM - Add NULL check for task_info - Do not duplicate the debug messages (ti vs no ti) - Get first reference of task_info in vm_init(), put last in vm_fini() V4: Fixed review comments from Felix - fix double reference increment in create_task_info - change amdgpu_vm_get_task_info_pasid - additional changes in amdgpu_gem.c while porting

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-21961	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: eth: bnxt: fix truesize for mb-xdp-pass case When mb-xdp is set and return is XDP_PASS, packet is converted from xdp_buff to sk_buff with xdp_update_skb_shared_info() in bnxt_xdp_build_skb(). bnxt_xdp_build_skb() passes incorrect truesize argument to xdp_update_skb_shared_info(). The truesize is calculated as BNXT_RX_PAGE_SIZE * sinfo->nr_frags but the skb_shared_info was wiped by napi_build_skb() before. So it stores sinfo->nr_frags before bnxt_xdp_build_skb() and use it instead of getting skb_shared_info from xdp_get_shared_info_from_buff(). Splat looks like: ------------[cut here]------------ WARNING: CPU: 2 PID: 0 at net/core/skbuff.c:6072 skb_try_coalesce+0x504/0x590 Modules linked in: xt_nat xt_tcpudp veth af_packet xt_conntrack nft_chain_nat xt_MASQUERADE nf_conntrack_netlink xfrm_user xt_addrtype nft_coms CPU: 2 UID: 0 PID: 0 Comm: swapper/2 Not tainted 6.14.0-rc2+ #3 RIP: 0010:skb_try_coalesce+0x504/0x590 Code: 4b fd ff ff 49 8b 34 24 40 80 e6 40 0f 84 3d fd ff ff 49 8b 74 24 48 40 f6 c6 01 0f 84 2e fd ff ff 48 8d 4e ff e9 25 fd ff ff <0f> 0b e99 RSP: 0018:ffffb62c4120caa8 EFLAGS: 00010287 RAX: 0000000000000003 RBX: ffffb62c4120cb14 RCX: 0000000000000ec0 RDX: 0000000000001000 RSI: ffffa06e5d7dc000 RDI: 0000000000000003 RBP: ffffa06e5d7ddec0 R08: ffffa06e6120a800 R09: ffffa06e7a119900 R10: 0000000000002310 R11: ffffa06e5d7dcec0 R12: ffffe4360575f740 R13: ffffe43600000000 R14: 0000000000000002 R15: 0000000000000002 FS: 0000000000000000(0000) GS:ffffa0755f700000(0000) knlGS:0000000000000000 CS: 0010 DS: 0000 ES: 0000 CR0: 0000000080050033 CR2: 00007f147b76b0f8 CR3: 00000001615d4000 CR4: 00000000007506f0 PKRU: 55555554 Call Trace: ? __warn+0x84/0x130 ? skb_try_coalesce+0x504/0x590 ? report_bug+0x18a/0x1a0 ? handle_bug+0x53/0x90 ? exc_invalid_op+0x14/0x70 ? asm_exc_invalid_op+0x16/0x20 ? skb_try_coalesce+0x504/0x590 inet_frag_reasm_finish+0x11f/0x2e0 ip_defrag+0x37a/0x900 ip_local_deliver+0x51/0x120 ip_sublist_rcv_finish+0x64/0x70 ip_sublist_rcv+0x179/0x210 ip_list_rcv+0xf9/0x130 How to reproduce: ip link set $interface1 xdp obj xdp_pass.o ip link set $interface1 mtu 9000 up ip a a 10.0.0.1/24 dev $interface1 ip link set $interfac2 mtu 9000 up ip a a 10.0.0.2/24 dev $interface2 ping 10.0.0.1 -s 65000 Following ping.py patch adds xdp-mb-pass case. so ping.py is going to be able to reproduce this issue.
CVE-2024-46834	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ethtool: fail closed if we can't get max channel used in indirection tables Commit 0d1b7d6c9274 ("bnxt: fix crashes when reducing ring count with active RSS contexts") proves that allowing indirection table to contain channels with out of bounds IDs may lead to crashes. Currently the max channel check in the core gets skipped if driver can't fetch the indirection table or when we can't allocate memory. Both of those conditions should be extremely rare but if they do happen we should try to be safe and fail the channel change.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-22113	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ext4: avoid journaling sb update on error if journal is destroying Presently we always BUG_ON if trying to start a transaction on a journal marked with JBD2_UNMOUNT, since this should never happen. However, while ltp running stress tests, it was observed that in case of some error handling paths, it is possible for update_super_work to start a transaction after the journal is destroyed eg: (umount) ext4_kill_sb kill_block_super generic_shutdown_super sync_filesystem /* commits all txns */ evict_inodes /* might start a new txn */ ext4_put_super flush_work(&sbi->s_sb_upd_work) /* flush the workqueue */ jbd2_journal_destroy journal_kill_thread journal->j_flags |= JBD2_UNMOUNT; jbd2_journal_commit_transaction jbd2_journal_get_descriptor_buffer jbd2_journal_bmap ext4_journal_bmap ext4_map_blocks ... ext4_inode_error ext4_handle_error schedule_work(&sbi->s_sb_upd_work) /* work queue kicks in */ update_super_work jbd2_journal_start start_this_handle BUG_ON(journal->j_flags & JBD2_UNMOUNT) Hence, introduce a new mount flag to indicate journal is destroying and only do a journaled (and deferred) update of sb if this flag is not set. Otherwise, just fallback to an un-journaled commit. Further, in the journal destroy path, we have the following sequence: 1. Set mount flag indicating journal is destroying 2. force a commit and wait for it 3. flush pending sb updates This sequence is important as it ensures that, after this point, there is no sb update that might be journaled so it is safe to update the sb outside the journal. (To avoid race discussed in 2d01ddc86606) Also, we don't need a similar check in ext4_grp_locked_error since it is only called from mballoc and AFAICT it would be always valid to schedule work here.
CVE-2024-49968	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ext4: filesystems without casefold feature cannot be mounted with siphash When mounting the ext4 filesystem, if the default hash version is set to DX_HASH_SIPHASH but the casefold feature is not set, exit the mounting.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-23130	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: f2fs: fix to avoid panic once fallocation fails for pinfile syzbot reports a f2fs bug as below: ------------[cut here]------------ kernel BUG at fs/f2fs/segment.c:2746! CPU: 0 UID: 0 PID: 5323 Comm: syz.0.0 Not tainted 6.13.0-rc2-syzkaller-00018-g7cb1b4663150 #0 RIP: 0010:get_new_segment fs/f2fs/segment.c:2746 [inline] RIP: 0010:new_curseg+0x1f52/0x1f70 fs/f2fs/segment.c:2876 Call Trace: __allocate_new_segment+0x1ce/0x940 fs/f2fs/segment.c:3210 f2fs_allocate_new_section fs/f2fs/segment.c:3224 [inline] f2fs_allocate_pinning_section+0xfa/0x4e0 fs/f2fs/segment.c:3238 f2fs_expand_inode_data+0x696/0xca0 fs/f2fs/file.c:1830 f2fs_fallocate+0x537/0xa10 fs/f2fs/file.c:1940 vfs_fallocate+0x569/0x6e0 fs/open.c:327 do_vfs_ioctl+0x258c/0x2e40 fs/ioctl.c:885 __do_sys_ioctl fs/ioctl.c:904 [inline] __se_sys_ioctl+0x80/0x170 fs/ioctl.c:892 do_syscall_x64 arch/x86/entry/common.c:52 [inline] do_syscall_64+0xf3/0x230 arch/x86/entry/common.c:83 entry_SYSCALL_64_after_hwframe+0x77/0x7f Concurrent pinfile allocation may run out of free section, result in panic in get_new_segment(), let's expand pin_sem lock coverage to include f2fs_gc(), so that we can make sure to reclaim enough free space for following allocation. In addition, do below changes to enhance error path handling: - call f2fs_bug_on() only in non-pinfile allocation path in get_new_segment(). - call reset_curseg_fields() to reset all fields of curseg in new_curseg()
CVE-2025-38333	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: f2fs: fix to bail out in get_new_segment() ------------[cut here]------------ WARNING: CPU: 3 PID: 579 at fs/f2fs/segment.c:2832 new_curseg+0x5e8/0x6dc pc : new_curseg+0x5e8/0x6dc Call trace: new_curseg+0x5e8/0x6dc f2fs_allocate_data_block+0xa54/0xe28 do_write_page+0x6c/0x194 f2fs_do_write_node_page+0x38/0x78 __write_node_page+0x248/0x6d4 f2fs_sync_node_pages+0x524/0x72c f2fs_write_checkpoint+0x4bc/0x9b0 __checkpoint_and_complete_reqs+0x80/0x244 issue_checkpoint_thread+0x8c/0xec kthread+0x114/0x1bc ret_from_fork+0x10/0x20 get_new_segment() detects inconsistent status in between free_segmap and free_secmap, let's record such error into super block, and bail out get_new_segment() instead of continue using the segment.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38626	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: f2fs: fix to trigger foreground gc during f2fs_map_blocks() in lfs mode w/ "mode=lfs" mount option, generic/299 will cause system panic as below: ------------[cut here]------------ kernel BUG at fs/f2fs/segment.c:2835! Call Trace: f2fs_allocate_data_block+0x6f4/0xc50 f2fs_map_blocks+0x970/0x1550 f2fs_iomap_begin+0xb2/0x1e0 iomap_iter+0x1d6/0x430 __iomap_dio_rw+0x208/0x9a0 f2fs_file_write_iter+0x6b3/0xfa0 aio_write+0x15d/0x2e0 io_submit_one+0x55e/0xab0 __x64_sys_io_submit+0xa5/0x230 do_syscall_64+0x84/0x2f0 entry_SYSCALL_64_after_hwframe+0x76/0x7e RIP: 0010:new_curseg+0x70f/0x720 The root cause of we run out-of-space is: in f2fs_map_blocks(), f2fs may trigger foreground gc only if it allocates any physical block, it will be a little bit later when there is multiple threads writing data w/ aio/dio/bufio method in parallel, since we always use OPU in lfs mode, so f2fs_map_blocks() does block allocations aggressively. In order to fix this issue, let's give a chance to trigger foreground gc in prior to block allocation in f2fs_map_blocks().
CVE-2025-23132	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: f2fs: quota: fix to avoid warning in dquot_writeback_dquots() F2FS-fs (dm-59): checkpoint=enable has some unwritten data. ------------[cut here]------------ WARNING: CPU: 6 PID: 8013 at fs/quota/dquot.c:691 dquot_writeback_dquots+0x2fc/0x308 pc : dquot_writeback_dquots+0x2fc/0x308 lr : f2fs_quota_sync+0xcc/0x1c4 Call trace: dquot_writeback_dquots+0x2fc/0x308 f2fs_quota_sync+0xcc/0x1c4 f2fs_write_checkpoint+0x3d4/0x9b0 f2fs_issue_checkpoint+0x1bc/0x2c0 f2fs_sync_fs+0x54/0x150 f2fs_do_sync_file+0x2f8/0x814 __f2fs_ioctl+0x1960/0x3244 f2fs_ioctl+0x54/0xe0 __arm64_sys_ioctl+0xa8/0xe4 invoke_syscall+0x58/0x114 checkpoint and f2fs_remount may race as below, resulting triggering warning in dquot_writeback_dquots(). atomic write remount - do_remount - down_write(&sb->s_umount); - f2fs_remount - ioctl - f2fs_do_sync_file - f2fs_sync_fs - f2fs_write_checkpoint - block_operations - locked = down_read_trylock(&sbi->sb->s_umount) : fail to lock due to the write lock was held by remount - up_write(&sb->s_umount); - f2fs_quota_sync - dquot_writeback_dquots - WARN_ON_ONCE(!rwsem_is_locked(&sb->s_umount)) : trigger warning because s_umount lock was unlocked by remount If checkpoint comes from mount/umount/remount/freeze/quotactl, caller of checkpoint has already held s_umount lock, calling dquot_writeback_dquots() in the context should be safe. So let's record task to sbi->umount_lock_holder, so that checkpoint can know whether the lock has held in the context or not by checking current w/ it. In addition, in order to not misrepresent caller of checkpoint, we should not allow to trigger async checkpoint for those callers: mount/umount/remount/ freeze/quotactl.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-71105	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: f2fs: use global inline_xattr_slab instead of per-sb slab cache As Hong Yun reported in mailing list: loop7: detected capacity change from 0 to 131072 ------------[cut here]------------ kmem_cache of name 'f2fs_xattr_entry-7:7' already exists WARNING: CPU: 0 PID: 24426 at mm/slab_common.c:110 kmem_cache_sanity_check mm/slab_common.c:109 [inline] WARNING: CPU: 0 PID: 24426 at mm/slab_common.c:110 __kmem_cache_create_args+0xa6/0x320 mm/slab_common.c:307 CPU: 0 UID: 0 PID: 24426 Comm: syz.7.1370 Not tainted 6.17.0-rc4 #1 PREEMPT(full) Hardware name: QEMU Standard PC (i440FX + PIIX, 1996), BIOS 1.13.0-1ubuntu1.1 04/01/2014 RIP: 0010:kmem_cache_sanity_check mm/slab_common.c:109 [inline] RIP: 0010:__kmem_cache_create_args+0xa6/0x320 mm/slab_common.c:307 Call Trace: __kmem_cache_create include/linux/slab.h:353 [inline] f2fs_kmem_cache_create fs/f2fs/f2fs.h:2943 [inline] f2fs_init_xattr_caches+0xa5/0xe0 fs/f2fs/xattr.c:843 f2fs_fill_super+0x1645/0x2620 fs/f2fs/super.c:4918 get_tree_bdev_flags+0x1fb/0x260 fs/super.c:1692 vfs_get_tree+0x43/0x140 fs/super.c:1815 do_new_mount+0x201/0x550 fs/namespace.c:3808 do_mount fs/namespace.c:4136 [inline] __do_sys_mount fs/namespace.c:4347 [inline] __se_sys_mount+0x298/0x2f0 fs/namespace.c:4324 do_syscall_x64 arch/x86/entry/syscall_64.c:63 [inline] do_syscall_64+0x8e/0x3a0 arch/x86/entry/syscall_64.c:94 entry_SYSCALL_64_after_hwframe+0x76/0x7e The bug can be reproduced w/ below scripts: - mount /dev/vdb /mnt1 - mount /dev/vdc /mnt2 - umount /mnt1 - mounnt /dev/vdb /mnt1 The reason is if we created two slab caches, named f2fs_xattr_entry-7:3 and f2fs_xattr_entry-7:7, and they have the same slab size. Actually, slab system will only create one slab cache core structure which has slab name of "f2fs_xattr_entry-7:3", and two slab caches share the same structure and cache address. So, if we destroy f2fs_xattr_entry-7:3 cache w/ cache address, it will decrease reference count of slab cache, rather than release slab cache entirely, since there is one more user has referenced the cache. Then, if we try to create slab cache w/ name "f2fs_xattr_entry-7:3" again, slab system will find that there is existed cache which has the same name and trigger the warning. Let's changes to use global inline_xattr_slab instead of per-sb slab cache for fixing.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-39731	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: f2fs: vm_unmap_ram() may be called from an invalid context When testing F2FS with xfstests using UFS backed virtual disks the kernel complains sometimes that f2fs_release_decomp_mem() calls vm_unmap_ram() from an invalid context. Example trace from f2fs/007 test: f2fs/007 5s ... [12:59:38][8.902525] run fstests f2fs/007 [11.468026] BUG: sleeping function called from invalid context at mm/vmalloc.c:2978 [11.471849] in_atomic(): 1, irqs_disabled(): 1, non_block: 0, pid: 68, name: irq/22-ufshcd [11.475357] preempt_count: 1, expected: 0 [11.476970] RCU nest depth: 0, expected: 0 [11.478531] CPU: 0 UID: 0 PID: 68 Comm: irq/22-ufshcd Tainted: G W 6.16.0-rc5-xfstests-ufs-g40f92e79b0aa #9 PREEMPT(none) [11.478535] Tainted: [W]=WARN [11.478536] Hardware name: QEMU Standard PC (Q35 + ICH9, 2009), BIOS 1.16.3-debian-1.16.3-2 04/01/2014 [11.478537] Call Trace: [11.478543] [11.478545] dump_stack_lvl+0x4e/0x70 [11.478554] __might_resched.cold+0xaf/0xbe [11.478557] vm_unmap_ram+0x21/0xb0 [11.478560] f2fs_release_decomp_mem+0x59/0x80 [11.478563] f2fs_free_dic+0x18/0x1a0 [11.478565] f2fs_finish_read_bio+0xd7/0x290 [11.478570] blk_update_request+0xec/0x3b0 [11.478574] ? sbitmap_queue_clear+0x3b/0x60 [11.478576] scsi_end_request+0x27/0x1a0 [11.478582] scsi_io_completion+0x40/0x300 [11.478583] ufshcd_mcq_poll_cqe_lock+0xa3/0xe0 [11.478588] ufshcd_sl_intr+0x194/0x1f0 [11.478592] ufshcd_threaded_intr+0x68/0xb0 [11.478594] ? __pfx_irq_thread_fn+0x10/0x10 [11.478599] irq_thread_fn+0x20/0x60 [11.478602] ? __pfx_irq_thread_fn+0x10/0x10 [11.478603] irq_thread+0xb9/0x180 [11.478605] ? __pfx_irq_thread_dtor+0x10/0x10 [11.478607] ? __pfx_irq_thread+0x10/0x10 [11.478609] kthread+0x10a/0x230 [11.478614] ? __pfx_kthread+0x10/0x10 [11.478615] ret_from_fork+0x7e/0xd0 [11.478619] ? __pfx_kthread+0x10/0x10 [11.478621] ret_from_fork_asm+0x1a/0x30 [11.478623] This patch modifies in_task() check inside f2fs_read_end_io() to also check if interrupts are disabled. This ensures that pages are unmapped asynchronously in an interrupt handler.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38164	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: f2fs: zone: fix to avoid inconsistence in between SIT and SSA w/ below testcase, it will cause inconsistence in between SIT and SSA. create_null_blk 512 2 1024 1024 mkfs.f2fs -m /dev/nullb0 mount /dev/nullb0 /mnt/f2fs/ touch /mnt/f2fs/file f2fs_io pinfile set /mnt/f2fs/file fallocate -l 4GiB /mnt/f2fs/file F2FS-fs (nullb0): Inconsistent segment (0) type [1, 0] in SSA and SIT CPU: 5 UID: 0 PID: 2398 Comm: fallocate Tainted: G O 6.13.0-rc1 #84 Tainted: [O]=OOT_MODULE Hardware name: innotek GmbH VirtualBox/VirtualBox, BIOS VirtualBox 12/01/2006 Call Trace: dump_stack_lvl+0xb3/0xd0 dump_stack+0x14/0x20 f2fs_handle_critical_error+0x18c/0x220 [f2fs] f2fs_stop_checkpoint+0x38/0x50 [f2fs] do_garbage_collect+0x674/0x6e0 [f2fs] f2fs_gc_range+0x12b/0x230 [f2fs] f2fs_allocate_pinning_section+0x5c/0x150 [f2fs] f2fs_expand_inode_data+0x1cc/0x3c0 [f2fs] f2fs_fallocate+0x3c3/0x410 [f2fs] vfs_fallocate+0x15f/0x4b0 __x64_sys_fallocate+0x4a/0x80 x64_sys_call+0x15e8/0x1b80 do_syscall_64+0x68/0x130 entry_SYSCALL_64_after_hwframe+0x67/0x6f RIP: 0033:0x7f9dba5197ca F2FS-fs (nullb0): Stopped filesystem due to reason: 4 The reason is f2fs_gc_range() may try to migrate block in curseg, however, its SSA block is not uptodate due to the last summary block data is still in cache of curseg. In this patch, we add a condition in f2fs_gc_range() to check whether section is opened or not, and skip block migration for opened section.
CVE-2025-21976	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: fbdev: hyperv_fb: Allow graceful removal of framebuffer When a Hyper-V framebuffer device is unbind, hyperv_fb driver tries to release the framebuffer forcefully. If this framebuffer is in use it produce the following WARN and hence this framebuffer is never released. [44.111220] WARNING: CPU: 35 PID: 1882 at drivers/video/fbdev/core/fb_info.c:70 framebuffer_release+0x2c/0x40 < snip > [44.111289] Call Trace: [44.111290] [44.111291] ? show_regs+0x6c/0x80 [44.111295] ? __warn+0x8d/0x150 [44.111298] ? framebuffer_release+0x2c/0x40 [44.111300] ? report_bug+0x182/0x1b0 [44.111303] ? handle_bug+0x6e/0xb0 [44.111306] ? exc_invalid_op+0x18/0x80 [44.111308] ? asm_exc_invalid_op+0x1b/0x20 [44.111311] ? framebuffer_release+0x2c/0x40 [44.111313] ? hvfb_remove+0x86/0xa0 [hyperv_fb] [44.111315] vmbus_remove+0x24/0x40 [hv_vmbus] [44.111323] device_remove+0x40/0x80 [44.111325] device_release_driver_internal+0x20b/0x270 [44.111327] ? bus_find_device+0xb3/0xf0 Fix this by moving the release of framebuffer and assosiated memory to fb_ops.fb_destroy function, so that framebuffer framework handles it gracefully. While we fix this, also replace manual registrations/unregistration of framebuffer with devm_register_framebuffer.
CVE-2026-23236	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: fbdev: smscufx: properly copy ioctl memory to kernelspace The UFX_IOCTL_REPORT_DAMAGE ioctl does not properly copy data from userspace to kernelspace, and instead directly references the memory, which can cause problems if invalid data is passed from userspace. Fix this all up by correctly copying the memory before accessing it within the kernel.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2026-23168	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: flex_proportions: make fprop_new_period() hardirq safe Bernd has reported a lockdep splat from flexible proportions code that is essentially complaining about the following race: run_timer_softirq - we are in softirq context call_timer_fn writeout_period fprop_new_period write_seqcount_begin(&p->sequence); ... blk_mq_end_request() blk_update_request() ext4_end_bio() folio_end_writeback() __wb_writeout_add() __fprop_add_percpu_max() if (unlikely(max_frac < FPROP_FRAC_BASE)) { fprop_fraction_percpu() seq = read_seqcount_begin(&p->sequence); - sees odd sequence so loops indefinitely Note that a deadlock like this is only possible if the bdi has configured maximum fraction of writeout throughput which is very rare in general but frequent for example for FUSE bdis. To fix this problem we have to make sure write section of the sequence counter is irqsafe.
CVE-2025-38615	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: fs/ntfs3: cancle set bad inode after removing name fails The reproducer uses a file0 on a ntfs3 file system with a corrupted i_link. When renaming, the file0's inode is marked as a bad inode because the file name cannot be deleted. The underlying bug is that make_bad_inode() is called on a live inode. In some cases it's "icache lookup finds a normal inode, d_splice_alias() is called to attach it to dentry, while another thread decides to call make_bad_inode() on it - that would evict it from icache, but we'd already found it there earlier". In some it's outright "we have an inode attached to dentry - that's how we got it in the first place; let's call make_bad_inode() on it just for shits and giggles".
CVE-2025-39819	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: fs/smb: Fix inconsistent refcnt update A possible inconsistent update of refcount was identified in `smb2_compound_op`. Such inconsistent update could lead to possible resource leaks. Why it is a possible bug: 1. In the comment section of the function, it clearly states that the reference to `cfile` should be dropped after calling this function. 2. Every control flow path would check and drop the reference to `cfile`, except the patched one. 3. Existing callers would not handle refcount update of `cfile` if -ENOMEM is returned. To fix the bug, an extra goto label "out" is added, to make sure that the cleanup logic would always be respected. As the problem is caused by the allocation failure of `vars`, the cleanup logic between label "finished" and "out" can be safely ignored. According to the definition of function `is_replayable_error`, the error code of "-ENOMEM" is not recoverable. Therefore, the replay logic also gets ignored.
CVE-2025-39753	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: gfs2: Set .migrate_folio in gfs2_{rgrp,meta}_aops Clears up the warning added in 7ee3647243e5 ("migrate: Remove call to ->writepage") that occurs in various xfstests, causing "something found in dmesg" failures. [341.136573] gfs2_meta_aops does not implement migrate_folio [341.136953] WARNING: CPU: 1 PID: 36 at mm/migrate.c:944 move_to_new_folio+0x2f8/0x300

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38710	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: gfs2: Validate i_depth for exhash directories A fuzzer test introduced corruption that ends up with a depth of 0 in dir_e_read(), causing an undefined shift by 32 at: index = hash >> (32 - dip->i_depth); As calculated in an open-coded way in dir_make_exhash(), the minimum depth for an exhash directory is ilog2(sdp->sd_hash_ptrs) and 0 is invalid as sdp->sd_hash_ptrs is fixed as sdp->bsize / 16 at mount time. So we can avoid the undefined behaviour by checking for depth values lower than the minimum in gfs2_dinode_in(). Values greater than the maximum are already being checked for there. Also switch the calculation in dir_make_exhash() to use ilog2() to clarify how the depth is calculated. Tested with the syzkaller repro.c and xfstests '-g quick'.
CVE-2025-38671	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: i2c: qup: jump out of the loop in case of timeout Original logic only sets the return value but doesn't jump out of the loop if the bus is kept active by a client. This is not expected. A malicious or buggy i2c client can hang the kernel in this case and should be avoided. This is observed during a long time test with a PCA953x GPIO extender. Fix it by changing the logic to not only sets the return value, but also jumps out of the loop and return to the caller with -ETIMEDOUT.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2026-23113	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: io_uring/io-wq: check IO_WQ_BIT_EXIT inside work run loop Currently this is checked before running the pending work. Normally this is quite fine, as work items either end up blocking (which will create a new worker for other items), or they complete fairly quickly. But syzbot reports an issue where io-wq takes seemingly forever to exit, and with a bit of debugging, this turns out to be because it queues a bunch of big (2GB - 4096b) reads with a /dev/msr* file. Since this file type doesn't support ->read_iter(), loop_rw_iter() ends up handling them. Each read returns 16MB of data read, which takes 20 (!!) seconds. With a bunch of these pending, processing the whole chain can take a long time. Easily longer than the syzbot uninterruptible sleep timeout of 140 seconds. This then triggers a complaint off the io-wq exit path: INFO: task syz.4.135:6326 blocked for more than 143 seconds. Not tainted syzkaller #0 Blocked by coredump. "echo 0 > /proc/sys/kernel/hung_task_timeout_secs" disables this message. task:syz.4.135 state:D stack:26824 pid:6326 tgid:6324 ppid:5957 task_flags:0x400548 flags:0x00080000 Call Trace: context_switch kernel/sched/core.c:5256 [inline] __schedule+0x1139/0x6150 kernel/sched/core.c:6863 __schedule_loop kernel/sched/core.c:6945 [inline] schedule+0xe7/0x3a0 kernel/sched/core.c:6960 schedule_timeout+0x257/0x290 kernel/time/sleep_timeout.c:75 do_wait_for_common kernel/sched/completion.c:100 [inline] __wait_for_common+0x2fc/0x4e0 kernel/sched/completion.c:121 io_wq_exit_workers io_uring/io-wq.c:1328 [inline] io_wq_put_and_exit+0x271/0x8a0 io_uring/io-wq.c:1356 io_uring_clean_tctx+0x10d/0x190 io_uring/tctx.c:203 io_uring_cancel_generic+0x69c/0x9a0 io_uring/cancel.c:651 io_uring_files_cancel include/linux/io_uring.h:19 [inline] do_exit+0x2ce/0x2bd0 kernel/exit.c:911 do_group_exit+0xd3/0x2a0 kernel/exit.c:1112 get_signal+0x2671/0x26d0 kernel/signal.c:3034 arch_do_signal_or_restart+0x8f/0x7e0 arch/x86/kernel/signal.c:337 __exit_to_user_mode_loop kernel/entry/common.c:41 [inline] exit_to_user_mode_loop+0x8c/0x540 kernel/entry/common.c:75 __exit_to_user_mode_prepare include/linux/irq-entry-common.h:226 [inline] syscall_exit_to_user_mode_prepare include/linux/irq-entry-common.h:256 [inline] syscall_exit_to_user_mode_work include/linux/entry-common.h:159 [inline] syscall_exit_to_user_mode include/linux/entry-common.h:194 [inline] do_syscall_64+0x4ee/0xf80 arch/x86/entry/syscall_64.c:100 entry_SYSCALL_64_after_hwframe+0x77/0x7f RIP: 0033:0x7fa02738f749 RSP: 002b:00007fa0281ae0e8 EFLAGS: 00000246 ORIG_RAX: 00000000000000ca RAX: ffffffffffffe00 RBX: 00007fa0275e6098 RCX: 00007fa02738f749 RDX: 0000000000000000 RSI: 0000000000000080 RDI: 00007fa0275e6098 RBP: 00007fa0275e6090 R08: 0000000000000000 R09: 0000000000000000 R10: 0000000000000000 R11: 0000000000000246 R12: 0000000000000000 R13: 00007fa0275e6128 R14: 00007fff14e4fcb0 R15: 00007fff14e4fd98 There's really nothing wrong here, outside of processing these reads will take a LONG time. However, we can speed up the exit by checking the IO_WQ_BIT_EXIT inside the io_worker_handle_work() loop, as syzbot will exit the ring after queueing up all of these reads. Then once the first item is processed, io-wq will simply cancel the rest. That should avoid syzbot

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-39739	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	running into this complaint again. In the Linux kernel, the following vulnerability has been resolved: iommu/arm-smmu-qcom: Add SM6115 MDSS compatible Add the SM6115 MDSS compatible to clients compatible list, as it also needs that workaround. Without this workaround, for example, QRB4210 RB2 which is based on SM4250/SM6115 generates a lot of smmu unhandled context faults during boot: arm_smmu_context_fault: 116854 callbacks suppressed arm-smmu c600000.iommu: Unhandled context fault: fsr=0x402, iova=0x5c0ec600, fsynr=0x320021, cbfrsynra=0x420, cb=5 arm-smmu c600000.iommu: FSR = 00000402 [Format=2 TF], SID=0x420 arm-smmu c600000.iommu: FSYNR0 = 00320021 [S1CBNDX=50 PNU PLVL=1] arm-smmu c600000.iommu: Unhandled context fault: fsr=0x402, iova=0x5c0d7800, fsynr=0x320021, cbfrsynra=0x420, cb=5 arm-smmu c600000.iommu: FSR = 00000402 [Format=2 TF], SID=0x420 and also failed initialisation of Iontium It9611uxc, gpu and dpu is observed: (binding MDSS components triggered by It9611uxc have failed) ------------[cut here]------------ !aspace WARNING: CPU: 6 PID: 324 at drivers/gpu/drm/msm/msm_gem_vma.c:130 msm_gem_vma_init+0x150/0x18c [msm] Modules linked in: ... (long list of modules) CPU: 6 UID: 0 PID: 324 Comm: (udev-worker) Not tainted 6.15.0-03037-gaacc73ceeb8b #4 PREEMPT Hardware name: Qualcomm Technologies, Inc. QRB4210 RB2 (DT) pstate: 80000005 (Nzcv daif -PAN -UAO -TCO -DIT -SSBS BTYPE=--) pc : msm_gem_vma_init+0x150/0x18c [msm] lr : msm_gem_vma_init+0x150/0x18c [msm] sp : ffff80008144b280 ... Call trace: msm_gem_vma_init+0x150/0x18c [msm] (P) get_vma_locked+0xc0/0x194 [msm] msm_gem_get_and_pin_iova_range+0x4c/0xdc [msm] msm_gem_kernel_new+0x48/0x160 [msm] msm_gpu_init+0x34c/0x53c [msm] adreno_gpu_init+0x1b0/0x2d8 [msm] a6xx_gpu_init+0x1e8/0x9e0 [msm] adreno_bind+0x2b8/0x348 [msm] component_bind_all+0x100/0x230 msm_drm_bind+0x13c/0x3d0 [msm] try_to_bring_up_aggregate_device+0x164/0x1d0 __component_add+0xa4/0x174 component_add+0x14/0x20 dsi_dev_attach+0x20/0x34 [msm] dsi_host_attach+0x58/0x98 [msm] devm_mipi_dsi_attach+0x34/0x90 It9611uxc_attach_dsi.isra.0+0x94/0x124 [Iontium_It9611uxc] It9611uxc_probe+0x540/0x5fc [Iontium_It9611uxc] i2c_device_probe+0x148/0x2a8 really_probe+0xbc/0x2c0 __driver_probe_device+0x78/0x120 driver_probe_device+0x3c/0x154 __driver_attach+0x90/0x1a0 bus_for_each_dev+0x68/0xb8 driver_attach+0x24/0x30 bus_add_driver+0xe4/0x208 driver_register+0x68/0x124 i2c_register_driver+0x48/0xcc It9611uxc_driver_init+0x20/0x1000 [Iontium_It9611uxc] do_one_initcall+0x60/0x1d4 do_init_module+0x54/0x1fc load_module+0x1748/0x1c8c init_module_from_file+0x74/0xa0 __arm64_sys_finit_module+0x130/0x2f8 invoke_syscall+0x48/0x104 el0_svc_common.constprop.0+0xc0/0xe0 do_el0_svc+0x1c/0x28 el0_svc+0x2c/0x80 el0t_64_sync_handler+0x10c/0x138 el0t_64_sync+0x198/0x19c ---[end trace 0000000000000000]--- msm_dpu 5e01000.display-controller: [drm:msm_gpu_init [msm]] *ERROR* could not allocate memptrs: -22 msm_dpu 5e01000.display-controller: failed to load adreno gpu platform a400000.remoteproc:glink-

CVE	Severity	CVSS	CVSS Vector	Component	Description
					edge:apr:service@Z:dais: Adding to iommu group 19 msm_dpu 5e01000.display-controller: failed to bind 5900000.gpu (ops a3xx_ops [msm]): -22 msm_dpu 5e01000.display-controller: adev bind failed: -22 lt9611uxc 0-002b: failed to attach dsi to host lt9611uxc 0-002b: probe with driver lt9611uxc failed with error -22
CVE-2025-37877	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: iommu: Clear iommu-dma ops on cleanup If iommu_device_register() encounters an error, it can end up tearing down already-configured groups and default domains, however this currently still leaves devices hooked up to iommu-dma (and even historically the behaviour in this area was at best inconsistent across architectures/drivers...) Although in the case that an IOMMU is present whose driver has failed to probe, users cannot necessarily expect DMA to work anyway, it's still arguable that we should do our best to put things back as if the IOMMU driver was never there at all, and certainly the potential for crashing in iommu-dma itself is undesirable. Make sure we clean up the dev->dma_iommu flag along with everything else.
CVE-2025-71097	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ipv4: Fix reference count leak when using error routes with nexthop objects When a nexthop object is deleted, it is marked as dead and then fib_table_flush() is called to flush all the routes that are using the dead nexthop. The current logic in fib_table_flush() is to only flush error routes (e.g., blackhole) when it is called as part of network namespace dismantle (i.e., with flush_all=true). Therefore, error routes are not flushed when their nexthop object is deleted: # ip link add name dummy1 up type dummy # ip nexthop add id 1 dev dummy1 # ip route add 198.51.100.1/32 nhid 1 # ip route add blackhole 198.51.100.2/32 nhid 1 # ip nexthop del id 1 # ip route show blackhole 198.51.100.2 nhid 1 dev dummy1 As such, they keep holding a reference on the nexthop object which in turn holds a reference on the nexthop device, resulting in a reference count leak: # ip link del dev dummy1 [70.516258] unregister_netdevice: waiting for dummy1 to become free. Usage count = 2 Fix by flushing error routes when their nexthop is marked as dead. IPv6 does not suffer from this problem.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2026-23011	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ipv4: ip_gre: make ipgre_header() robust Analog to commit db5b4e39c4e6 ("ip6_gre: make ip6gre_header() robust") Over the years, syzbot found many ways to crash the kernel in ipgre_header() [1]. This involves team or bonding drivers ability to dynamically change their dev->needed_headroom and/or dev->hard_header_len In this particular crash mld_newpack() allocated an skb with a too small reserve/headroom, and by the time mld_sendpack() was called, syzbot managed to attach an ipgre device. [1] skbuff: skb_under_panic: text:ffffffff89ea3cb7 len:2030915468 put:2030915372 head:ffff888058b43000 data:ffff887fdfa6e194 tail:0x120 end:0x6c0 dev:team0 kernel BUG at net/core/skbuff.c:213 ! Oops: invalid opcode: 0000 [#1] SMP KASAN PTI CPU: 1 UID: 0 PID: 1322 Comm: kworker/1:9 Not tainted syzkaller #0 PREEMPT(full) Hardware name: Google Google Compute Engine/Google Compute Engine, BIOS Google 10/25/2025 Workqueue: mld mld_ifc_work RIP: 0010:skb_panic+0x157/0x160 net/core/skbuff.c:213 Call Trace: skb_under_panic net/core/skbuff.c:223 [inline] skb_push+0xc3/0xe0 net/core/skbuff.c:2641 ipgre_header+0x67/0x290 net/ipv4/ip_gre.c:897 dev_hard_header include/linux/netdevice.h:3436 [inline] neigh_connected_output+0x286/0x460 net/core/neighbour.c:1618 NF_HOOK_COND include/linux/netfilter.h:307 [inline] ip6_output+0x340/0x550 net/ipv6/ip6_output.c:247 NF_HOOK+0x9e/0x380 include/linux/netfilter.h:318 mld_sendpack+0x8d4/0xe60 net/ipv6/mcast.c:1855 mld_send_cr net/ipv6/mcast.c:2154 [inline] mld_ifc_work+0x83e/0xd60 net/ipv6/mcast.c:2693 process_one_work kernel/workqueue.c:3257 [inline] process_scheduled_works+0xad1/0x1770 kernel/workqueue.c:3340 worker_thread+0x8a0/0xda0 kernel/workqueue.c:3421 kthread+0x711/0x8a0 kernel/kthread.c:463 ret_from_fork+0x510/0xa50 arch/x86/kernel/process.c:158 ret_from_fork_asm+0x1a/0x30 arch/x86/entry/entry_64.S:246

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2024-50304	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ipv4: ip_tunnel: Fix suspicious RCU usage warning in ip_tunnel_find() The per-netns IP tunnel hash table is protected by the RTNL mutex and ip_tunnel_find() is only called from the control path where the mutex is taken. Add a lockdep expression to hlist_for_each_entry_rcu() in ip_tunnel_find() in order to validate that the mutex is held and to silence the suspicious RCU usage warning [1]. [1] WARNING: suspicious RCU usage 6.12.0-rc3-custom-gd95d9a31aceb #139 Not tainted ----------------------------- net/ipv4/ip_tunnel.c:221 RCU-list traversed in non-reader section!! other info that might help us debug this: rcu_scheduler_active = 2, debug_locks = 1 1 lock held by ip/362: #0: ffffffff86fc7cb0 (rtnl_mutex) {+.+.}-{3:3}, at: rtnetlink_rcv_msg+0x377/0xf60 stack backtrace: CPU: 12 UID: 0 PID: 362 Comm: ip Not tainted 6.12.0-rc3-custom-gd95d9a31aceb #139 Hardware name: Bochs Bochs, BIOS Bochs 01/01/2011 Call Trace: dump_stack_lvl+0xba/0x110 lockdep_rcu_suspicious.cold+0x4f/0xd6 ip_tunnel_find+0x435/0x4d0 ip_tunnel_newlink+0x517/0x7a0 ipgre_newlink+0x14c/0x170 __rtnl_newlink+0x1173/0x19c0 rtnl_newlink+0x6c/0xa0 rtnetlink_rcv_msg+0x3cc/0xf60 netlink_rcv_skb+0x171/0x450 netlink_unicast+0x539/0x7f0 netlink_sendmsg+0x8c1/0xd80 ____sys_sendmsg+0x8f9/0xc20 ___sys_sendmsg+0x197/0x1e0 __sys_sendmsg+0x122/0x1f0 do_syscall_64+0xbb/0x1d0 entry_SYSCALL_64_after_hwframe+0x77/0x7f
CVE-2026-23124	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ipv6: annotate data-race in ndisc_router_discovery() syzbot found that ndisc_router_discovery() could read and write in6_dev->ra_mtu without holding a lock [1] This looks fine, IFLA_INET6_RA_MTU is best effort. Add READ_ONCE()/WRITE_ONCE() to document the race. Note that we might also reject illegal MTU values (mtu < IPV6_MIN_MTU || mtu > skb->dev->mtu) in a future patch. [1] BUG: KCSAN: data-race in ndisc_router_discovery / ndisc_router_discovery read to 0xffff888119809c20 of 4 bytes by task 25817 on cpu 1: ndisc_router_discovery+0x151d/0x1c90 net/ipv6/ndisc.c:1558 ndisc_rcv+0x2ad/0x3d0 net/ipv6/ndisc.c:1841 icmpv6_rcv+0xe5a/0x12f0 net/ipv6/icmp.c:989 ip6_protocol_deliver_rcu+0xb2a/0x10d0 net/ipv6/ip6_input.c:438 ip6_input_finish+0xf0/0x1d0 net/ipv6/ip6_input.c:489 NF_HOOK include/linux/netfilter.h:318 [inline] ip6_input+0x5e/0x140 net/ipv6/ip6_input.c:500 ip6_mc_input+0x27c/0x470 net/ipv6/ip6_input.c:590 dst_input include/net/dst.h:474 [inline] ip6_rcv_finish+0x336/0x340 net/ipv6/ip6_input.c:79 ... write to 0xffff888119809c20 of 4 bytes by task 25816 on cpu 0: ndisc_router_discovery+0x155a/0x1c90 net/ipv6/ndisc.c:1559 ndisc_rcv+0x2ad/0x3d0 net/ipv6/ndisc.c:1841 icmpv6_rcv+0xe5a/0x12f0 net/ipv6/icmp.c:989 ip6_protocol_deliver_rcu+0xb2a/0x10d0 net/ipv6/ip6_input.c:438 ip6_input_finish+0xf0/0x1d0 net/ipv6/ip6_input.c:489 NF_HOOK include/linux/netfilter.h:318 [inline] ip6_input+0x5e/0x140 net/ipv6/ip6_input.c:500 ip6_mc_input+0x27c/0x470 net/ipv6/ip6_input.c:590 dst_input include/net/dst.h:474 [inline] ip6_rcv_finish+0x336/0x340 net/ipv6/ip6_input.c:79 ... value changed: 0x00000000 -> 0xe5400659

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2026-23085	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: irqchip/gic-v3-its: Avoid truncating memory addresses On 32-bit machines with CONFIG_ARM_LPAE, it is possible for lowmem allocations to be backed by addresses physical memory above the 32-bit address limit, as found while experimenting with larger VMSPLIT configurations. This caused the qemu virt model to crash in the GICv3 driver, which allocates the 'itt' object using GFP_KERNEL. Since all memory below the 4GB physical address limit is in ZONE_DMA in this configuration, kmalloc() defaults to higher addresses for ZONE_NORMAL, and the ITS driver stores the physical address in a 32-bit 'unsigned long' variable. Change the itt_addr variable to the correct phys_addr_t type instead, along with all other variables in this driver that hold a physical address. The gicv5 driver correctly uses u64 variables, while all other irqchip drivers don't call virt_to_phys or similar interfaces. It's expected that other device drivers have similar issues, but fixing this one is sufficient for booting a virtio based guest.
CVE-2025-38698	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: jfs: Regular file corruption check The reproducer builds a corrupted file on disk with a negative i_size value. Add a check when opening this file to avoid subsequent operation failures.
CVE-2024-58094	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: jfs: add check read-only before truncation in jfs_truncate_nolock() Added a check for "read-only" mode in the `jfs_truncate_nolock` function to avoid errors related to writing to a read-only filesystem. Call stack: block_write_begin() { jfs_write_failed() { jfs_truncate() { jfs_truncate_nolock() { txEnd() { ... log = JFS_SBI(tblk->sb)->log; // (log == NULL) If the `isReadOnly(ip)` condition is triggered in `jfs_truncate_nolock`, the function execution will stop, and no further data modification will occur. Instead, the `xtTruncate` function will be called with the "COMMIT_WMAP" flag, preventing modifications in "read-only" mode.
CVE-2024-58095	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: jfs: add check read-only before txBeginAnon() call Added a read-only check before calling `txBeginAnon` in `extAlloc` and `extRecord`. This prevents modification attempts on a read-only mounted filesystem, avoiding potential errors or crashes. Call trace: txBeginAnon+0xac/0x154 extAlloc+0xe8/0xdec fs/jfs/jfs_extent.c:78 jfs_get_block+0x340/0xb98 fs/jfs/inode.c:248 __block_write_begin_int+0x580/0x166c fs/buffer.c:2128 __block_write_begin fs/buffer.c:2177 [inline] block_write_begin+0x98/0x11c fs/buffer.c:2236 jfs_write_begin+0x44/0x88 fs/jfs/inode.c:299

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-37925	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: jfs: reject on-disk inodes of an unsupported type Syzbot has reported the following BUG: kernel BUG at fs/inode.c:668! Oops: invalid opcode: 0000 [#1] PREEMPT SMP KASAN PTI CPU: 3 UID: 0 PID: 139 Comm: jfsCommit Not tainted 6.12.0-rc4-syzkaller-00085-g4e46774408d9 #0 Hardware name: QEMU Standard PC (i440FX + PIIX, 1996), BIOS 1.16.3-3.fc41 04/01/2014 RIP: 0010:clear_inode+0x168/0x190 Code: 4c 89 f7 e8 ba fe e5 ff e9 61 ff ff ff 44 89 f1 80 e1 07 80 c1 03 38 c1 7c c1 4c 89 f7 e8 90 ff e5 ff eb b7 0b e8 01 5d 7f ff 90 0f 0b e8 f9 5c 7f ff 90 0f 0b e8 f1 5c 7f RSP: 0018:ffffc900027dfae8 EFLAGS: 00010093 RAX: ffffffff82157a87 RBX: 0000000000000001 RCX: ffff888104d4b980 RDX: 0000000000000000 RSI: 0000000000000001 RDI: 0000000000000000 RBP: ffffc900027dfc90 R08: ffffffff82157977 R09: fffff520004fbf38 R10: dfffffc0000000000 R11: fffff520004fbf38 R12: dfffffc0000000000 R13: ffff88811315bc00 R14: ffff88811315bda8 R15: ffff88811315bb80 FS: 0000000000000000(0000) GS:ffff888135f00000(0000) knlGS:0000000000000000 CS: 0010 DS: 0000 ES: 0000 CR0: 0000000080050033 CR2: 00005565222e0578 CR3: 0000000026ef0000 CR4: 00000000000006f0 Call Trace: ? __die_body+0x5f/0xb0 ? die+0x9e/0xc0 ? do_trap+0x15a/0x3a0 ? clear_inode+0x168/0x190 ? do_error_trap+0x1dc/0x2c0 ? clear_inode+0x168/0x190 ? __pfx_do_error_trap+0x10/0x10 ? report_bug+0x3cd/0x500 ? handle_invalid_op+0x34/0x40 ? clear_inode+0x168/0x190 ? exc_invalid_op+0x38/0x50 ? asm_exc_invalid_op+0x1a/0x20 ? clear_inode+0x57/0x190 ? clear_inode+0x167/0x190 ? clear_inode+0x168/0x190 ? clear_inode+0x167/0x190 jfs_evict_inode+0xb5/0x440 ? __pfx_jfs_evict_inode+0x10/0x10 evict+0x4ea/0x9b0 ? __pfx_evict+0x10/0x10 ? iput+0x713/0xa50 txUpdateMap+0x931/0xb10 ? __pfx_txUpdateMap+0x10/0x10 jfs_lazycommit+0x49a/0xb80 ? _raw_spin_unlock_irqrestore+0x8f/0x140 ? lockdep_hardirqs_on+0x99/0x150 ? __pfx_jfs_lazycommit+0x10/0x10 ? __pfx_default_wake_function+0x10/0x10 ? __kthread_parkme+0x169/0x1d0 ? __pfx_jfs_lazycommit+0x10/0x10 kthread+0x2f2/0x390 ? __pfx_jfs_lazycommit+0x10/0x10 ? __pfx_kthread+0x10/0x10 ret_from_fork+0x4d/0x80 ? __pfx_kthread+0x10/0x10 ret_from_fork_asm+0x1a/0x30 This happens when 'clear_inode()' makes an attempt to finalize an underlying JFS inode of unknown type. According to JFS layout description from https://jfs.sourceforge.net/project/pub/jfslayout.pdf, inode types from 5 to 15 are reserved for future extensions and should not be encountered on a valid filesystem. So add an extra check for valid inode type in 'copy_from_dinode()'.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38029	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: kasan: avoid sleepable page allocation from atomic context apply_to_pte_range() enters the lazy MMU mode and then invokes kasan_populate_vmalloc_pte() callback on each page table walk iteration. However, the callback can go into sleep when trying to allocate a single page, e.g. if an architecutre disables preemption on lazy MMU mode enter. On s390 if make arch_enter_lazy_mmu_mode() -> preempt_enable() and arch_leave_lazy_mmu_mode() -> preempt_disable(), such crash occurs: [0.663336] BUG: sleeping function called from invalid context at ./include/linux/sched/mm.h:321 [0.663348] in_atomic(): 1, irqs_disabled(): 0, non_block: 0, pid: 2, name: kthreadd [0.663358] preempt_count: 1, expected: 0 [0.663366] RCU nest depth: 0, expected: 0 [0.663375] no locks held by kthreadd/2. [0.663383] Preemption disabled at: [0.663386] [<0002f3284cbb4eda>] apply_to_pte_range+0xfa/0x4a0 [0.663405] CPU: 0 UID: 0 PID: 2 Comm: kthreadd Not tainted 6.15.0-rc5-gcc-kasan-00043-gd76bb1ebb558-dirty #162 PREEMPT [0.663408] Hardware name: IBM 3931 A01 701 (KVM/Linux) [0.663409] Call Trace: [0.663410] [<0002f3284c385f58>] dump_stack_lvl+0xe8/0x140 [0.663413] [<0002f3284c507b9e>] __might_resched+0x66e/0x700 [0.663415] [<0002f3284cc4f6c0>] __alloc_frozen_pages_noprof+0x370/0x4b0 [0.663419] [<0002f3284ccc73c0>] alloc_pages_mpol+0x1a0/0x4a0 [0.663421] [<0002f3284ccc8518>] alloc_frozen_pages_noprof+0x88/0xc0 [0.663424] [<0002f3284ccc8572>] alloc_pages_noprof+0x22/0x120 [0.663427] [<0002f3284cc341ac>] get_free_pages_noprof+0x2c/0xc0 [0.663429] [<0002f3284cceba70>] kasan_populate_vmalloc_pte+0x50/0x120 [0.663433] [<0002f3284cbb4ef8>] apply_to_pte_range+0x118/0x4a0 [0.663435] [<0002f3284cbc7c14>] apply_to_pmd_range+0x194/0x3e0 [0.663437] [<0002f3284cbc99be>] __apply_to_page_range+0x2fe/0x7a0 [0.663440] [<0002f3284cbc9e88>] apply_to_page_range+0x28/0x40 [0.663442] [<0002f3284ccebf12>] kasan_populate_vmalloc+0x82/0xa0 [0.663445] [<0002f3284cc1578c>] alloc_vmap_area+0x34c/0xc10 [0.663448] [<0002f3284cc1c2a6>] __get_vm_area_node+0x186/0x2a0 [0.663451] [<0002f3284cc1e696>] __vmalloc_node_range_noprof+0x116/0x310 [0.663454] [<0002f3284cc1d950>] __vmalloc_node_noprof+0xd0/0x110 [0.663457] [<0002f3284c454b88>] alloc_thread_stack_node+0xf8/0x330 [0.663460] [<0002f3284c458d56>] dup_task_struct+0x66/0x4d0 [0.663463] [<0002f3284c45be90>] copy_process+0x280/0x4b90 [0.663465] [<0002f3284c460940>] kernel_clone+0xd0/0x4b0 [0.663467] [<0002f3284c46115e>] kernel_thread+0xbe/0xe0 [0.663469] [<0002f3284c4e440e>] kthreadd+0x50e/0x7f0 [0.663472] [<0002f3284c38c04a>] __ret_from_fork+0x8a/0xf0 [0.663475] [<0002f3284ed57ff2>] ret_from_fork+0xa/0x38 Instead of allocating single pages per-PTE, bulk-allocate the shadow memory prior to applying kasan_populate_vmalloc_pte() callback on a page range.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-68211	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ksm: use range-walk function to jump over holes in scan_get_next_rmap_item Currently, scan_get_next_rmap_item() walks every page address in a VMA to locate mergeable pages. This becomes highly inefficient when scanning large virtual memory areas that contain mostly unmapped regions, causing ksmd to use large amount of cpu without deduplicating much pages. This patch replaces the per-address lookup with a range walk using walk_page_range(). The range walker allows KSM to skip over entire unmapped holes in a VMA, avoiding unnecessary lookups. This problem was previously discussed in [1]. Consider the following test program which creates a 32 TiB mapping in the virtual address space but only populates a single page: #include #include #include /* 32 TiB */ const size_t size = 32ul * 1024 * 1024 * 1024 * 1024; int main() { char *area = mmap(NULL, size, PROT_READ | PROT_WRITE, MAP_NORESERVE | MAP_PRIVATE | MAP_ANON, -1, 0); if (area == MAP_FAILED) { perror("mmap() failed\n"); return -1; } /* Populate a single page such that we get an anon_vma. */ *area = 0; /* Enable KSM. */ madvise(area, size, MADV_MERGEABLE); pause(); return 0; } $ ./ksm-sparse & $ echo 1 > /sys/kernel/mm/ksm/run Without this patch ksmd uses 100% of the cpu for a long time (more then 1 hour in my test machine) scanning all the 32 TiB virtual address space that contain only one mapped page. This makes ksmd essentially deadlocked not able to deduplicate anything of value. With this patch ksmd walks only the one mapped page and skips the rest of the 32 TiB virtual address space, making the scan fast using little cpu.
CVE-2025-71150	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ksmbd: Fix refcount leak when invalid session is found on session lookup When a session is found but its state is not SMB2_SESSION_VALID, It indicates that no valid session was found, but it is missing to decrement the reference count acquired by the session lookup, which results in a reference count leak. This patch fixes the issue by explicitly calling ksmbd_user_session_put to release the reference to the session.
CVE-2025-39720	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ksmbd: fix refcount leak causing resource not released When ksmbd_conn_releasing(opinfo->conn) returns true,the refcount was not decremented properly, causing a refcount leak that prevents the count from reaching zero and the memory from being released.
CVE-2026-23093	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ksmbd: smbd: fix dma_unmap_sg() nents The dma_unmap_sg() functions should be called with the same nents as the dma_map_sg(), not the value the map function returned.
CVE-2025-39942	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ksmbd: smbdirect: verify remaining_data_length respects max_fragmented_recv_size This is inspired by the check for data_offset + data_length.
CVE-2024-46823	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: kunit/overflow: Fix UB in overflow_allocation_test The 'device_name' array doesn't exist out of the 'overflow_allocation_test' function scope. However, it is being used as a driver name when calling 'kunit_driver_create' from 'kunit_device_register'. It produces the kernel panic with KASAN enabled. Since this variable is used in one place only, remove it and pass the device name into kunit_device_register directly as an ascii string.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2026-23120	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: l2tp: avoid one data-race in l2tp_tunnel_del_work() We should read sk->sk_socket only when dealing with kernel sockets. syzbot reported the following data-race: BUG: KCSAN: data-race in l2tp_tunnel_del_work / sk_common_release write to 0xffff88811c182b20 of 8 bytes by task 5365 on cpu 0: sk_set_socket include/net/sock.h:2092 [inline] sock_orphan include/net/sock.h:2118 [inline] sk_common_release+0xae/0x230 net/core/sock.c:4003 udp_lib_close+0x15/0x20 include/net/udp.h:325 inet_release+0xce/0xf0 net/ipv4/af_inet.c:437 __sock_release net/socket.c:662 [inline] sock_close+0x6b/0x150 net/socket.c:1455 __fput+0x29b/0x650 fs/file_table.c:468 ____fput+0x1c/0x30 fs/file_table.c:496 task_work_run+0x131/0x1a0 kernel/task_work.c:233 resume_user_mode_work include/linux/resume_user_mode.h:50 [inline] __exit_to_user_mode_loop kernel/entry/common.c:44 [inline] exit_to_user_mode_loop+0x1fe/0x740 kernel/entry/common.c:75 __exit_to_user_mode_prepare include/linux/irq-entry-common.h:226 [inline] syscall_exit_to_user_mode_prepare include/linux/irq-entry-common.h:256 [inline] syscall_exit_to_user_mode_work include/linux/entry-common.h:159 [inline] syscall_exit_to_user_mode include/linux/entry-common.h:194 [inline] do_syscall_64+0x1e1/0x2b0 arch/x86/entry/syscall_64.c:100 entry_SYSCALL_64_after_hwframe+0x77/0x7f read to 0xffff88811c182b20 of 8 bytes by task 827 on cpu 1: l2tp_tunnel_del_work+0x2f/0x1a0 net/l2tp/l2tp_core.c:1418 process_one_work kernel/workqueue.c:3257 [inline] process_scheduled_works+0x4ce/0x9d0 kernel/workqueue.c:3340 worker_thread+0x582/0x770 kernel/workqueue.c:3421 kthread+0x489/0x510 kernel/kthread.c:463 ret_from_fork+0x149/0x290 arch/x86/kernel/process.c:158 ret_from_fork_asm+0x1a/0x30 arch/x86/entry/entry_64.S:246 value changed: 0xffff88811b818000 -> 0x0000000000000000
CVE-2026-23136	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: libceph: reset sparse-read state in osd_fault() When a fault occurs, the connection is abandoned, reestablished, and any pending operations are retried. The OSD client tracks the progress of a sparse-read reply using a separate state machine, largely independent of the messenger's state. If a connection is lost mid-payload or the sparse-read state machine returns an error, the sparse-read state is not reset. The OSD client will then interpret the beginning of a new reply as the continuation of the old one. If this makes the sparse-read machinery enter a failure state, it may never recover, producing loops like: libceph: [0] got 0 extents libceph: data len 142248331 != extent len 0 libceph: osd0 (1)...:6801 socket error on read libceph: data len 142248331 != extent len 0 libceph: osd0 (1)...:6801 socket error on read Therefore, reset the sparse-read state in osd_fault(), ensuring retries start from a clean state.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38709	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: loop: Avoid updating block size under exclusive owner Syzbot came up with a reproducer where a loop device block size is changed underneath a mounted filesystem. This causes a mismatch between the block device block size and the block size stored in the superblock causing confusion in various places such as fs/buffer.c. The particular issue triggered by syzbot was a warning in __getblk_slow() due to requested buffer size not matching block device block size. Fix the problem by getting exclusive hold of the loop device to change its block size. This fails if somebody (such as filesystem) has already an exclusive ownership of the block device and thus prevents modifying the loop device under some exclusive owner which doesn't expect it.
CVE-2026-23121	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: mISDN: annotate data-race around dev->work dev->work can re read locklessly in mISDN_read() and mISDN_poll(). Add READ_ONCE()/WRITE_ONCE() annotations. BUG: KCSAN: data-race in mISDN_ioctl / mISDN_read write to 0xffff88812d848280 of 4 bytes by task 10864 on cpu 1: misdn_add_timer drivers/isdn/mISDN/timerdev.c:175 [inline] mISDN_ioctl+0x2fb/0x550 drivers/isdn/mISDN/timerdev.c:233 vfs_ioctl fs/ioctl.c:51 [inline] __do_sys_ioctl fs/ioctl.c:597 [inline] __se_sys_ioctl+0xce/0x140 fs/ioctl.c:583 __x64_sys_ioctl+0x43/0x50 fs/ioctl.c:583 x64_sys_call+0x14b0/0x3000 arch/x86/include/generated/asm/syscalls_64.h:17 do_syscall_x64 arch/x86/entry/syscall_64.c:63 [inline] do_syscall_64+0xd8/0x2c0 arch/x86/entry/syscall_64.c:94 entry_SYSCALL_64_after_hwframe+0x77/0x7f read to 0xffff88812d848280 of 4 bytes by task 10857 on cpu 0: mISDN_read+0x1f2/0x470 drivers/isdn/mISDN/timerdev.c:112 do_loop_readv_writev fs/read_write.c:847 [inline] vfs_readv+0x3fb/0x690 fs/read_write.c:1020 do_readv+0xe7/0x210 fs/read_write.c:1080 __do_sys_readv fs/read_write.c:1165 [inline] __se_sys_readv fs/read_write.c:1162 [inline] __x64_sys_readv+0x45/0x50 fs/read_write.c:1162 x64_sys_call+0x2831/0x3000 arch/x86/include/generated/asm/syscalls_64.h:20 do_syscall_x64 arch/x86/entry/syscall_64.c:63 [inline] do_syscall_64+0xd8/0x2c0 arch/x86/entry/syscall_64.c:94 entry_SYSCALL_64_after_hwframe+0x77/0x7f value changed: 0x00000000 -> 0x00000001
CVE-2025-38451	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: md/md-bitmap: fix GPF in bitmap_get_stats() The commit message of commit 6ec1f0239485 ("md/md-bitmap: fix stats collection for external bitmaps") states: Remove the external bitmap check as the statistics should be available regardless of bitmap storage location. Return -EINVAL only for invalid bitmap with no storage (neither in superblock nor in external file). But, the code does not adhere to the above, as it does only check for a valid super-block for "internal" bitmaps. Hence, we observe: Oops: GPF, probably for non-canonical address 0x1cd66f1f40000028 RIP: 0010:bitmap_get_stats+0x45/0xd0 Call Trace: seq_read_iter+0x2b9/0x46a seq_read+0x12f/0x180 proc_reg_read+0x57/0xb0 vfs_read+0xf6/0x380 ksys_read+0x6d/0xf0 do_syscall_64+0x8c/0x1b0 entry_SYSCALL_64_after_hwframe+0x76/0x7e We fix this by checking the existence of a super-block for both the internal and external case.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-22124	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: md/md-bitmap: fix wrong bitmap_limit for clustermd when write sb In clustermd, separate write-intent-bitmaps are used for each cluster node: 0 4k 8k 12k ---------- ---------- ----------	idle	md super	bm super [0] + bits		bm bits[0, contd]	bm super[1] + bits	bm bits[1, contd]		bm super[2] + bits	bm bits [2, contd]	bm super[3] + bits		bm bits [3, contd]			So in node 1, pg_index in __write_sb_page() could equal to bitmap->storage.file_pages. Then bitmap_limit will be calculated to 0. md_super_write() will be called with 0 size. That means the first 4k sb area of node 1 will never be updated through filemap_write_page(). This bug causes hang of mdadm/clustermd_tests/01r1_Grow_resize. Here use (pg_index % bitmap->storage.file_pages) to make calculation of bitmap_limit correct.
CVE-2025-22125	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: md/raid1,raid10: don't ignore IO flags If blk-wbt is enabled by default, it's found that raid write performance is quite bad because all IO are throttled by wbt of underlying disks, due to flag REQ_IDLE is ignored. And turns out this behaviour exist since blk-wbt is introduced. Other than REQ_IDLE, other flags should not be ignored as well, for example REQ_META can be set for filesystems, clearing it can cause priority reverse problems; And REQ_NOWAIT should not be cleared as well, because io will wait instead of failing directly in underlying disks. Fix those problems by keep IO flags from master bio. Fises: f51d46d0e7cb ("md: add support for REQ_NOWAIT")
CVE-2025-40325	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: md/raid10: wait barrier before returning discard request with REQ_NOWAIT raid10_handle_discard should wait barrier before returning a discard bio which has REQ_NOWAIT. And there is no need to print warning calltrace if a discard bio has REQ_NOWAIT flag. Quality engineer usually checks dmesg and reports error if dmesg has warning/error calltrace.
CVE-2024-50289	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: media: av7110: fix a spectre vulnerability As warned by smatch: drivers/staging/media/av7110/av7110_ca.c:270 dvb_ca_ioctl() warn: potential spectre issue 'av7110->ci_slot' [w] (local cap) There is a spectre-related vulnerability at the code. Fix it.
CVE-2025-38237	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: media: platform: exynos4-is: Add hardware sync wait to fimc_is_hw_change_mode() In fimc_is_hw_change_mode(), the function changes camera modes without waiting for hardware completion, risking corrupted data or system hangs if subsequent operations proceed before the hardware is ready. Add fimc_is_hw_wait_intmsr0_intmsd0() after mode configuration, ensuring hardware state synchronization and stable interrupt handling.
CVE-2025-39714	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: media: usbtv: Lock resolution while streaming When an program is streaming (ffplay) and another program (qv4l2) changes the TV standard from NTSC to PAL, the kernel crashes due to trying to copy to unmapped memory. Changing from NTSC to PAL increases the resolution in the usbtv struct, but the video plane buffer isn't adjusted, so it overflows. [hverkuil: call vb2_is_busy instead of vb2_is_streaming]

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2026-23097	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: migrate: correct lock ordering for hugetlb file folios Syzbot has found a deadlock (analyzed by Lance Yang): 1) Task (5749): Holds folio_lock, then tries to acquire i_mmap_rwsem(read lock). 2) Task (5754): Holds i_mmap_rwsem(write lock), then tries to acquire folio_lock. migrate_pages() -> migrate_hugetlbs() -> unmap_and_move_huge_page() <- Takes folio_lock! -> remove_migration_ptes() -> __rmap_walk_file() -> i_mmap_lock_read() <- Waits for i_mmap_rwsem(read lock)! hugetlbfs_fallocate() -> hugetlbfs_punch_hole() <- Takes i_mmap_rwsem(write lock)! -> hugetlbfs_zero_partial_page() -> filemap_lock_hugetlb_folio() -> filemap_lock_folio() -> __filemap_get_folio <- Waits for folio_lock! The migration path is the one taking locks in the wrong order according to the documentation at the top of mm/rmap.c. So expand the scope of the existing i_mmap_lock to cover the calls to remove_migration_ptes() too. This is (mostly) how it used to be after commit c0d0381ade79. That was removed by 336bf30eb765 for both file & anon hugetlb pages when it should only have been removed for anon hugetlb pages.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2024-49885	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: mm, slub: avoid zeroing kmalloc redzone Since commit 946fa0dbf2d8 ("mm/slub: extend redzone check to extra allocated kmalloc space than requested"), setting orig_size treats the wasted space (object_size - orig_size) as a redzone. However with init_on_free=1 we clear the full object->size, including the redzone. Additionally we clear the object metadata, including the stored orig_size, making it zero, which makes check_object() treat the whole object as a redzone. These issues lead to the following BUG report with "slub_debug=FUZ init_on_free=1": [0.000000] ========== ========== = [0.000000] BUG kmalloc-8 (Not tainted): kmalloc Redzone overwritten [0.000000] ---------- ---------- [0.000000] [0.000000] 0xffff000010032858-0xffff00001003285f @offset=2136. First byte 0x0 instead of 0xcc [0.000000] FIX kmalloc-8: Restoring kmalloc Redzone 0xffff000010032858-0xffff00001003285f=0xcc [0.000000] Slab 0xfffffdffc0400c80 objects=36 used=23 fp=0xffff000010032a18 flags=0x3fffe0000000200(workingset|node=0|zone=0|lastcpupid=0x1ffff) [0.000000] Object 0xffff000010032858 @offset=2136 fp=0xffff0000100328c8 [0.000000] [0.000000] Redzone ffff000010032850: cc cc cc cc cc cc cc cc ..... [0.000000] Object ffff000010032858: cc cc cc cc cc cc cc cc ..... [0.000000] Redzone ffff000010032860: cc cc cc cc cc cc cc cc ..... [0.000000] Padding ffff0000100328b4: 00 00 00 00 00 00 00 00 00 00 00 00 ..... [0.000000] CPU: 0 UID: 0 PID: 0 Comm: swapper/0 Not tainted 6.11.0-rc3-next-20240814-00004-g61844c55c3f4 #144 [0.000000] Hardware name: NXP i.MX95 19X19 board (DT) [0.000000] Call trace: [0.000000] dump_backtrace+0x90/0xe8 [0.000000] show_stack+0x18/0x24 [0.000000] dump_stack_lvl+0x74/0x8c [0.000000] dump_stack+0x18/0x24 [0.000000] print_trailer+0x150/0x218 [0.000000] check_object+0xe4/0x454 [0.000000] free_to_partial_list+0x2f8/0x5ec To address the issue, use orig_size to clear the used area. And restore the value of orig_size after clear the remaining area. When CONFIG_SLUB_DEBUG not defined, (get_orig_size()' directly returns s->object_size. So when using memset to init the area, the size can simply be orig_size, as orig_size returns object_size when CONFIG_SLUB_DEBUG not enabled. And orig_size can never be bigger than object_size.
CVE-2026-23142	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: mm/damon/sysfs-scheme: cleanup access_pattern subdirs on scheme dir setup failure When a DAMOS-scheme DAMON sysfs directory setup fails after setup of access_pattern/ directory, subdirectories of access_pattern/ directory are not cleaned up. As a result, DAMON sysfs interface is nearly broken until the system reboots, and the memory for the unremoved directory is leaked. Cleanup the directories under such failures.
CVE-2026-23144	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: mm/damon/sysfs: cleanup attrs subdirs on context dir setup failure When a context DAMON sysfs directory setup is failed after setup of attrs/ directory, subdirectories of attrs/ directory are not cleaned up. As a result, DAMON sysfs interface is nearly broken until the system reboots, and the memory for the unremoved directory is leaked. Cleanup the directories under such failures.
CVE-2024-42317	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: mm/huge_memory: avoid PMD-size page cache if needed xarray can't support arbitrary page cache size. the largest and

CVE	Severity	CVSS	CVSS Vector	Component	Description
					supported page cache size is defined as MAX_PAGECACHE_ORDER by commit 099d90642a71 ("mm/filemap: make MAX_PAGECACHE_ORDER acceptable to xarray"). However, it's possible to have 512MB page cache in the huge memory's collapsing path on ARM64 system whose base page size is 64KB. 512MB page cache is breaking the limitation and a warning is raised when the xarray entry is split as shown in the following example. [root@dhcp-10-26-1-207 ~]# cat /proc/1/smaps | grep KernelPageSize KernelPageSize: 64 kB [root@dhcp-10-26-1-207 ~]# cat /tmp/test.c : int main(int argc, char **argv) { const char *filename = TEST_XFS_FILENAME; int fd = 0; void *buf = (void *)-1, *p; int pgsize = getpagesize(); int ret = 0; if (pgsize != 0x10000) { fprintf(stdout, "System with 64KB base page size is required!\n"); return -EPERM; } system("echo 0 > /sys/devices/virtual/bdi/253:0/read_ahead_kb"); system("echo 1 > /proc/sys/vm/drop_caches"); /* Open the xfs file */ fd = open(filename, O_RDONLY); assert(fd > 0); /* Create VMA */ buf = mmap(NULL, TEST_MEM_SIZE, PROT_READ, MAP_SHARED, fd, 0); assert(buf != (void *)-1); fprintf(stdout, "mapped buffer at 0x%p\n", buf); /* Populate VMA */ ret = madvise(buf, TEST_MEM_SIZE, MADV_NOHUGEPAGE); assert(ret == 0); ret = madvise(buf, TEST_MEM_SIZE, MADV_POPULATE_READ); assert(ret == 0); /* Collapse VMA */ ret = madvise(buf, TEST_MEM_SIZE, MADV_HUGEPAGE); assert(ret == 0); ret = madvise(buf, TEST_MEM_SIZE, MADV_COLLAPSE); if (ret) { fprintf(stdout, "Error %d to madvise(MADV_COLLAPSE)\n", errno); goto out; } /* Split xarray entry. Write permission is needed */ munmap(buf, TEST_MEM_SIZE); buf = (void *)-1; close(fd); fd = open(filename, O_RDWR); assert(fd > 0); fallocate(fd, FALLOC_FL_KEEP_SIZE | FALLOC_FL_PUNCH_HOLE, TEST_MEM_SIZE - pgsize, pgsize); out: if (buf != (void *)-1) munmap(buf, TEST_MEM_SIZE); if (fd > 0) close(fd); return ret; } [root@dhcp-10-26-1-207 ~]# gcc /tmp/test.c -o /tmp/test [root@dhcp-10-26-1-207 ~]# /tmp/test ------------[cut here]------------ WARNING: CPU: 25 PID: 7560 at lib/xarray.c:1025 xas_split_alloc+0xf8/0x128 Modules linked in: nft_fib_inet nft_fib_ipv4 nft_fib_ipv6 nft_fib \ nft_reject_inet nf_reject_ipv4 nf_reject_ipv6 nft_reject nft_ct \ nft_chain_nat nf_nat nf_conntrack nf_defrag_ipv6 nf_defrag_ipv4 \ ip_set rfkill nf_tables nfnetlink vfat fat virtio_balloon drm fuse \ xfs libcrc32c crct10dif_ce ghash_ce sha2_ce sha256_arm64 virtio_net \ sha1_ce net_failover virtio_blk virtio_console failover dimlib virtio_mmio CPU: 25 PID: 7560 Comm: test Kdump: loaded Not tainted 6.10.0-rc7-gavin+ #9 Hardware name: QEMU KVM Virtual Machine, BIOS edk2-20240524-1.el9 05/24/2024 pstate: 83400005 (Nzcv daif +PAN -UAO +TCO +DIT -SSBS BTYPE=--) pc : xas_split_alloc+0xf8/0x128 lr : split_huge_page_to_list_to_order+0x1c4/0x780 sp : ffff8000ac32f660 x29: ffff8000ac32f660 x28: ffff0000e0969eb0 x27: ffff8000ac32f6c0 x26: 0000000000000c40 x25: ffff0000e0969eb0 x24: 000000000000000d x23: ffff8000ac32f6c0 x22: fffffffdfc0700000 x21: 0000000000000000 x20: 0000000000000000 x19: fffffffdfc0700000 x18: 0000000000000000 x17: 0000000000000000 x16: ffffd5f3708ffc70 x15: 0000000000000000 x14: 0000000000000000 x13: 0000000000000000 x12: 0000000000000000 x11: ffffffffffffffc0 x10: 0000000000000040 x9 : ffffd5f3708e692c x8 : 0000000000000003 x7 : 0000000000000000 x6 : ffff0000e0969eb8 x5 : ffffd5f37289e378 x4 : 0000000000000000 x3 : 0000000000000c40 x2 : 000000000000000d x1 : 0000000000000c x0 : 0000000000000000 Call trace: xas_split_alloc+0xf8/0x128 split_huge_page_to_list_to_order+0x1c4/0x780 truncate_inode_partial_folio+0xdc/0x160 truncate_inode_pages_range+0x1b4/0x4a8 truncate_pagecache_range+0x84/0xa ---truncated---

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2026-23100	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: mm/hugetlb: fix hugetlb_pmd_shared() Patch series "mm/hugetlb: fixes for PMD table sharing (incl. using mmu_gather)", v3. One functional fix, one performance regression fix, and two related comment fixes. I cleaned up my prototype I recently shared [1] for the performance fix, deferring most of the cleanups I had in the prototype to a later point. While doing that I identified the other things. The goal of this patch set is to be backported to stable trees "fairly" easily. At least patch #1 and #4. Patch #1 fixes hugetlb_pmd_shared() not detecting any sharing Patch #2 + #3 are simple comment fixes that patch #4 interacts with. Patch #4 is a fix for the reported performance regression due to excessive IPI broadcasts during fork()+exit(). The last patch is all about TLB flushes, IPIs and mmu_gather. Read: complicated There are plenty of cleanups in the future to be had + one reasonable optimization on x86. But that's all out of scope for this series. Runtime tested, with a focus on fixing the performance regression using the original reproducer [2] on x86. This patch (of 4): We switched from (wrongly) using the page count to an independent shared count. Now, shared page tables have a refcount of 1 (excluding speculative references) and instead use ptdesc->pt_share_count to identify sharing. We didn't convert hugetlb_pmd_shared(), so right now, we would never detect a shared PMD table as such, because sharing/unsharing no longer touches the refcount of a PMD table. Page migration, like mbind() or migrate_pages() would allow for migrating folios mapped into such shared PMD tables, even though the folios are not exclusive. In smaps we would account them as "private" although they are "shared", and we would be wrongly setting the PM_MMAP_EXCLUSIVE in the pagemap interface. Fix it by properly using ptdesc_pmd_is_shared() in hugetlb_pmd_shared().

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-40040	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: mm/ksm: fix flag-dropping behavior in ksm_madvise syzkaller discovered the following crash: (kernel BUG) [44.607039] ----------[cut here]---------- [44.607422] kernel BUG at mm/userfaultfd.c:2067! [44.608148] Oops: invalid opcode: 0000 [#1] SMP DEBUG_PAGEALLOC KASAN NOPTI [44.608814] CPU: 1 UID: 0 PID: 2475 Comm: reproducer Not tainted 6.16.0-rc6 #1 PREEMPT(none) [44.609635] Hardware name: QEMU Standard PC (i440FX + PIIX, 1996), BIOS rel-1.16.3-0-ga6ed6b701f0a-prebuilt.qemu.org 04/01/2014 [44.610695] RIP: 0010:userfaultfd_release_all+0x3a8/0x460 [44.617726] Call Trace: [44.617926] [44.619284] userfaultfd_release+0xef/0x1b0 [44.620976] __fput+0x3f9/0xb60 [44.621240] fput_close_sync+0x110/0x210 [44.622222] __x64_sys_close+0x8f/0x120 [44.622530] do_syscall_64+0x5b/0x2f0 [44.622840] entry_SYSCALL_64_after_hwframe+0x76/0x7e [44.623244] RIP: 0033:0x7f365bb3f227 Kernel panics because it detects UFFD inconsistency during userfaultfd_release_all(). Specifically, a VMA which has a valid pointer to vma->vm_userfaultfd_ctx, but no UFFD flags in vma->vm_flags. The inconsistency is caused in ksm_madvise(): when user calls madvise() with MADV_UNMEARGEABLE on a VMA that is registered for UFFD in MINOR mode, it accidentally clears all flags stored in the upper 32 bits of vma->vm_flags. Assuming x86_64 kernel build, unsigned long is 64-bit and unsigned int and int are 32-bit wide. This setup causes the following mishap during the &= ~VM_MERGEABLE assignment. VM_MERGEABLE is a 32-bit constant of type unsigned int, 0x8000'0000. After ~ is applied, it becomes 0x7fff'ffff unsigned int, which is then promoted to unsigned long before the & operation. This promotion fills upper 32 bits with leading 0s, as we're doing unsigned conversion (and even for a signed conversion, this wouldn't help as the leading bit is 0). & operation thus ends up AND-ing vm_flags with 0x0000'0000'7fff'ffff instead of intended 0xffff'ffff'7fff'ffff and hence accidentally clears the upper 32-bits of its value. Fix it by changing `VM_MERGEABLE` constant to unsigned long, using the BIT() macro. Note: other VM_* flags are not affected: This only happens to the VM_MERGEABLE flag, as the other VM_* flags are all constants of type int and after ~ operation, they end up with leading 1 and are thus converted to unsigned long with leading 1s. Note 2: After commit 31defc3b01d9 ("userfaultfd: remove (VM_)BUG_ON()s"), this is no longer a kernel BUG, but a WARNING at the same place: [45.595973] WARNING: CPU: 1 PID: 2474 at mm/userfaultfd.c:2067 but the root-cause (flag-drop) remains the same. [akpm@linux-foundation.org: rust bindgen wasn't able to handle BIT(), from Miguel]

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-37834	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: mm/vmscan: don't try to reclaim hwpoison folio Syzkaller reports a bug as follows: Injecting memory failure for pfn 0x18b00e at process virtual address 0x20ffd000 Memory failure: 0x18b00e: dirty swapcache page still referenced by 2 users Memory failure: 0x18b00e: recovery action for dirty swapcache page: Failed page: refcount:2 mapcount:0 mapping:0000000000000000 index:0x20ffd pfn:0x18b00e memcg:ffff0000dd6d9000 anon flags: 0x5ffffe00482011(locked|dirty|arch_1|swapbacked|hwpoison|node=0|zone=2|lastcpupid=0xfffff) raw: 005ffffe00482011 dead000000000100 dead000000000122 ffff0000e232a7c9 raw: 0000000000020ffd 0000000000000000 00000002ffffffff ffff0000dd6d9000 page dumped because: VM_BUG_ON_FOLIO(!folio_test_uptodate(folio)) ----------[cut here]---------- kernel BUG at mm/swap_state.c:184! Internal error: Oops - BUG: 00000000f2000800 [#1] SMP Modules linked in: CPU: 0 PID: 60 Comm: kswapd0 Not tainted 6.6.0-gcb097e7de84e #3 Hardware name: linux,dummy-virt (DT) pstate: 80400005 (Nzcv daif +PAN -UAO -TCO -DIT -SSBS BTYPE=--) pc : add_to_swap+0xbc/0x158 lr : add_to_swap+0xbc/0x158 sp : ffff800087f37340 x29: ffff800087f37340 x28: fffffc00052c0380 x27: ffff800087f37780 x26: ffff800087f37490 x25: ffff800087f37c78 x24: ffff800087f377a0 x23: ffff800087f37c50 x22: 0000000000000000 x21: fffffc00052c03b4 x20: 0000000000000000 x19: fffffc00052c0380 x18: 0000000000000000 x17: 296f696c6f662865 x16: 7461646f7470755f x15: 747365745f6f696c x14: 6f6621284f494c4f x13: 0000000000000001 x12: ffff600036d8b97b x11: 1fffe00036d8b97a x10: ffff600036d8b97a x9 : dfff800000000000 x8 : 00009fffc9274686 x7 : ffff0001b6c5cbd3 x6 : 0000000000000001 x5 : ffff0000c25896c0 x4 : 0000000000000000 x3 : 0000000000000000 x2 : 0000000000000000 x1 : ffff0000c25896c0 x0 : 0000000000000000 Call trace: add_to_swap+0xbc/0x158 shrink_folio_list+0x12ac/0x2648 shrink_inactive_list+0x318/0x948 shrink_lruvec+0x450/0x720 shrink_node_memcgs+0x280/0x4a8 shrink_node+0x128/0x978 balance_pgdat+0x4f0/0xb20 kswapd+0x228/0x438 kthread+0x214/0x230 ret_from_fork+0x10/0x20 I can reproduce this issue with the following steps: 1) When a dirty swapcache page is isolated by reclaim process and the page isn't locked, inject memory failure for the page. me_swapcache_dirty() clears uptodate flag and tries to delete from lru, but fails. Reclaim process will put the hwpoisoned page back to lru. 2) The process that maps the hwpoisoned page exits, the page is deleted the page will never be freed and will be in the lru forever. 3) If we trigger a reclaim again and tries to reclaim the page, add_to_swap() will trigger VM_BUG_ON_FOLIO due to the uptodate flag is cleared. To fix it, skip the hwpoisoned page in shrink_folio_list(). Besides, the hwpoison folio may not be unmapped by hwpoison_user_mappings() yet, unmap it in shrink_folio_list(), otherwise the folio will fail to be unmaped by hwpoison_user_mappings() since the folio isn't in lru list.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-21696	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: mm: clear uffd-wp PTE/PMD state on mremap() When mremap()ing a memory region previously registered with userfaultfd as write-protected but without UFFD_FEATURE_EVENT_REMAP, an inconsistency in flag clearing leads to a mismatch between the vma flags (which have uffd-wp cleared) and the pte/pmd flags (which do not have uffd-wp cleared). This mismatch causes a subsequent mprotect(PROT_WRITE) to trigger a warning in page_table_check_pte_flags() due to setting the pte to writable while uffd-wp is still set. Fix this by always explicitly clearing the uffd-wp pte/pmd flags on any such mremap() so that the values are consistent with the existing clearing of VM_UFFD_WP. Be careful to clear the logical flag regardless of its physical form; a PTE bit, a swap PTE bit, or a PTE marker. Cover PTE, huge PMD and hugetlb paths.
CVE-2025-21907	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: mm: memory-failure: update ttu flag inside unmap_poisoned_folio Patch series "mm: memory_failure: unmap poisoned folio during migrate properly", v3. Fix two bugs during folio migration if the folio is poisoned. This patch (of 3): Commit 6da6b1d4a7df ("mm/hwpoison: convert TTU_IGNORE_HWPOISON to TTU_HWPOISON") introduce TTU_HWPOISON to replace TTU_IGNORE_HWPOISON in order to stop send SIGBUS signal when accessing an error page after a memory error on a clean folio. However during page migration, anon folio must be set with TTU_HWPOISON during unmap_*(). For pagecache we need some policy just like the one in hwpoison_user_mappings to set this flag. So move this policy from hwpoison_user_mappings to unmap_poisoned_folio to handle this warning properly. Warning will be produced during unamp poison folio with the following log: ------------[cut here]------------ WARNING: CPU: 1 PID: 365 at mm/rmap.c:1847 try_to_unmap_one+0x8fc/0xd3c Modules linked in: CPU: 1 UID: 0 PID: 365 Comm: bash Tainted: G W 6.13.0-rc1-00018-gacdb4bbda7ab #42 Tainted: [W]=WARN Hardware name: QEMU QEMU Virtual Machine, BIOS 0.0.0 02/06/2015 pstate: 20400005 (nzCv daif +PAN -UAO -TCO -DIT -SSBS BTYPE=--) pc : try_to_unmap_one+0x8fc/0xd3c lr : try_to_unmap_one+0x3dc/0xd3c Call trace: try_to_unmap_one+0x8fc/0xd3c (P) try_to_unmap_one+0x3dc/0xd3c (L) rmap_walk_anon+0xdc/0x1f8 rmap_walk+0x3c/0x58 try_to_unmap+0x88/0x90 unmap_poisoned_folio+0x30/0xa8 do_migrate_range+0x4a0/0x568 offline_pages+0x5a4/0x670 memory_block_action+0x17c/0x374 memory_subsys_offline+0x3c/0x78 device_offline+0xa4/0xd0 state_store+0x8c/0xf0 dev_attr_store+0x18/0x2c sysfs_kf_write+0x44/0x54 kernfs_fop_write_iter+0x118/0x1a8 vfs_write+0x3a8/0x4bc ksys_write+0x6c/0xf8 __arm64_sys_write+0x1c/0x28 invoke_syscall+0x44/0x100 el0_svc_common.constprop.0+0x40/0xe0 do_el0_svc+0x1c/0x28 el0_svc+0x30/0xd0 el0t_64_sync_handler+0xc8/0xcc el0t_64_sync+0x198/0x19c ---[end trace 0000000000000000]--- [mawupeng1@huawei.com: unmap_poisoned_folio(): remove shadowed local `mapping', per Miaohe]
CVE-2025-39844	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: mm: move page table sync declarations to linux/pgtable.h During our internal testing, we started observing intermittent boot failures when the machine uses 4-level paging and has a large amount of persistent memory: BUG: unable to handle page fault for address:

CVE	Severity	CVSS	CVSS Vector	Component	Description
					ffffe70000000034 #PF: supervisor write access in kernel mode #PF: error_code(0x0002) - not-present page PGD 0 P4D 0 Oops: 0002 [#1] SMP NOPTI RIP: 0010:__init_single_page+0x9/0x6d Call Trace: __init_zone_device_page+0x17/0x5d memmap_init_zone_device+0x154/0x1bb pagemap_range+0x2e0/0x40f memremap_pages+0x10b/0x2f0 devm_memremap_pages+0x1e/0x60 dev_dax_probe+0xce/0x2ec [device_dax] dax_bus_probe+0x6d/0xc9 [... snip ...] It turns out that the kernel panics while initializing vmemmap (struct page array) when the vmemmap region spans two PGD entries, because the new PGD entry is only installed in init_mm.pgd, but not in the page tables of other tasks. And looking at __populate_section_memmap(): if (vmemmap_can_optimize(altmap, pgmap)) // does not sync top level page tables r = vmemmap_populate_compound_pages(pfn, start, end, nid, pgmap); else // sync top level page tables in x86 r = vmemmap_populate(start, end, nid, altmap); In the normal path, vmemmap_populate() in arch/x86/mm/init_64.c synchronizes the top level page table (See commit 9b861528a801 ("x86-64, mem: Update all PGDs for direct mapping and vmemmap mapping changes")) so that all tasks in the system can see the new vmemmap area. However, when vmemmap_can_optimize() returns true, the optimized path skips synchronization of top-level page tables. This is because vmemmap_populate_compound_pages() is implemented in core MM code, which does not handle synchronization of the top-level page tables. Instead, the core MM has historically relied on each architecture to perform this synchronization manually. We're not the first party to encounter a crash caused by not-sync'd top level page tables: earlier this year, Gwan-gyeong Mun attempted to address the issue [1] [2] after hitting a kernel panic when x86 code accessed the vmemmap area before the corresponding top-level entries were synced. At that time, the issue was believed to be triggered only when struct page was enlarged for debugging purposes, and the patch did not get further updates. It turns out that current approach of relying on each arch to handle the page table sync manually is fragile because 1) it's easy to forget to sync the top level page table, and 2) it's also easy to overlook that the kernel should not access the vmemmap and direct mapping areas before the sync. # The solution: Make page table sync more code robust and harder to miss To address this, Dave Hansen suggested [3] [4] introducing {pgd,p4d}_populate_kernel() for updating kernel portion of the page tables and allow each architecture to explicitly perform synchronization when installing top-level entries. With this approach, we no longer need to worry about missing the sync step, reducing the risk of future regressions. The new interface reuses existing ARCH_PAGE_TABLE_SYNC_MASK, PGTBL_P*D_MODIFIED and arch_sync_kernel_mappings() facility used by vmalloc and ioremap to synchronize page tables. pgd_populate_kernel() looks like this: static inline void pgd_populate_kernel(unsigned long addr, pgd_t *pgd, p4d_t *p4d) { pgd_populate(&init_mm, pgd, p4d); if (ARCH_PAGE_TABLE_SYNC_MASK & PGTBL_PGD_MODIFIED) arch_sync_kernel_mappings(addr, addr); } It is worth noting that vmalloc() and apply_to_range() carefully synchronizes page tables by calling p*d_alloc_track() and arch_sync_kernel_mappings(), and thus they are not affected by ---truncated---

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-71200	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: mmc: sdhci-of-dwcmshc: Prevent illegal clock reduction in HS200/HS400 mode When operating in HS200 or HS400 timing modes, reducing the clock frequency below 52MHz will lead to link broken as the Rockchip DWC MSHC controller requires maintaining a minimum clock of 52MHz in these modes. Add a check to prevent illegal clock reduction through debugfs: root@debian:/# echo 50000000 > /sys/kernel/debug/mmc0/clock root@debian:/# [30.090146] mmc0: running CQE recovery mmc0: cqhci: Failed to halt mmc0: cqhci: spurious TCN for tag 0 WARNING: drivers/mmc/host/cqhci-core.c:797 at cqhci_irq+0x254/0x818, CPU#1: kworker/1:0H/24 Modules linked in: CPU: 1 UID: 0 PID: 24 Comm: kworker/1:0H Not tainted 6.19.0-rc1-00001-g09db0998649d-dirty #204 PREEMPT Hardware name: Rockchip RK3588 EVB1 V10 Board (DT) Workqueue: kblockd blk_mq_run_work_fn pstate: 604000c9 (nZCv daIF +PAN -UAO -TCO -DIT -SSBS BTYPE=--) pc : cqhci_irq+0x254/0x818 lr : cqhci_irq+0x254/0x818 ...
CVE-2025-71088	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: mptcp: fallback earlier on simult connection Syzkaller reports a simult-connect race leading to inconsistent fallback status: WARNING: CPU: 3 PID: 33 at net/mptcp/subflow.c:1515 subflow_data_ready+0x40b/0x7c0 net/mptcp/subflow.c:1515 Modules linked in: CPU: 3 UID: 0 PID: 33 Comm: ksoftirqd/3 Not tainted syzkaller #0 PREEMPT(full) Hardware name: QEMU Standard PC (Q35 + ICH9, 2009), BIOS 1.16.3-debian-1.16.3-2~bpo12+1 04/01/2014 RIP: 0010:subflow_data_ready+0x40b/0x7c0 net/mptcp/subflow.c:1515 Code: 89 ee e8 78 61 3c f6 40 84 ed 75 21 e8 8e 66 3c f6 44 89 fe bf 07 00 00 00 e8 c1 61 3c f6 41 83 ff 07 74 09 e8 76 66 3c f6 90 <0f> 0b 90 e8 6d 66 3c f6 48 89 df e8 e5 ad ff ff 31 ff 89 c5 89 c6 RSP: 0018:ffffc900006cf338 EFLAGS: 00010246 RAX: 0000000000000000 RBX: ffff888031acd100 RCX: ffffffff8b7f2abf RDX: ffff88801e6ea440 RSI: ffffffff8b7f2aca RDI: 0000000000000005 RBP: 0000000000000000 R08: 0000000000000005 R09: 0000000000000007 R10: 0000000000000004 R11: 0000000000002c10 R12: ffff88802ba69900 R13: 1ffff920000d9e67 R14: ffff888046f81800 R15: 0000000000000004 FS: 0000000000000000(0000) GS:ffff8880d69bc000(0000) knlGS:0000000000000000 CS: 0010 DS: 0000 ES: 0000 CR0: 0000000080050033 CR2: 0000560fc0ca1670 CR3: 0000000032c3a000 CR4: 0000000000352ef0 Call Trace: tcp_data_queue+0x13b0/0x4f90 net/ipv4/tcp_input.c:5197 tcp_rcv_state_process+0xfdf/0x4ec0 net/ipv4/tcp_input.c:6922 tcp_v6_do_rcv+0x492/0x1740 net/ipv6/tcp_ipv6.c:1672 tcp_v6_rcv+0x2976/0x41e0 net/ipv6/tcp_ipv6.c:1918 ip6_protocol_deliver_rcu+0x188/0x1520 net/ipv6/ip6_input.c:438 ip6_input_finish+0x1e4/0x4b0 net/ipv6/ip6_input.c:489 NF_HOOK include/linux/netfilter.h:318 [inline] NF_HOOK include/linux/netfilter.h:312 [inline] ip6_input+0x105/0x2f0 net/ipv6/ip6_input.c:500 dst_input include/net/dst.h:471 [inline] ip6_rcv_finish net/ipv6/ip6_input.c:79 [inline] NF_HOOK include/linux/netfilter.h:318 [inline] NF_HOOK include/linux/netfilter.h:312 [inline] ipv6_rcv+0x264/0x650 net/ipv6/ip6_input.c:311 __netif_receive_skb_one_core+0x12d/0x1e0 net/core/dev.c:5979 __netif_receive_skb+0x1d/0x160 net/core/dev.c:6092 process_backlog+0x442/0x15e0 net/core/dev.c:6444

CVE	Severity	CVSS	CVSS Vector	Component	Description
					__napi_poll.constprop.0+0xba/0x550 net/core/dev.c:7494 napi_poll net/core/dev.c:7557 [inline] net_rx_action+0xa9f/0xfe0 net/core/dev.c:7684 handle_softirqs+0x216/0x8e0 kernel/softirq.c:579 run_ksoftirqd kernel/softirq.c:968 [inline] run_ksoftirqd+0x3a/0x60 kernel/softirq.c:960 smpboot_thread_fn+0x3f7/0xae0 kernel/smpboot.c:160 kthread+0x3c2/0x780 kernel/kthread.c:463 ret_from_fork+0x5d7/0x6f0 arch/x86/kernel/process.c:148 ret_from_fork_asm+0x1a/0x30 arch/x86/entry/entry_64.S:245 The TCP subflow can process the simult-connect syn-ack packet after transitioning to TCP_FIN1 state, bypassing the MPTCP fallback check, as the sk_state_change() callback is not invoked for * -> FIN_WAIT1 transitions. That will move the msk socket to an inconsistent status and the next incoming data will hit the reported splat. Close the race moving the simult-fallback check at the earliest possible stage - that is at syn-ack generation time. About the fixes tags: [2] was supposed to also fix this issue introduced by [3]. [1] is required as a dependence: it was not explicitly marked as a fix, but it is one and it has already been backported before [3]. In other words, this commit should be backported up to [3], including [2] and [1] if that's not already there.
CVE-2025-39907	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: mtd: rawnand: stm32_fmc2: avoid overlapping mappings on ECC buffer Avoid below overlapping mappings by using a contiguous non-cacheable buffer. [4.077708] DMA-API: stm32_fmc2_nfc 48810000.nand-controller: cacheline tracking EEXIST, overlapping mappings aren't supported [4.089103] WARNING: CPU: 1 PID: 44 at kernel/dma/debug.c:568 add_dma_entry+0x23c/0x300 [4.097071] Modules linked in: [4.100101] CPU: 1 PID: 44 Comm: kworker/u4:2 Not tainted 6.1.82 #1 [4.106346] Hardware name: STMicroelectronics STM32MP257F VALID1 SNOR / MB1704 (LPDDR4 Power discrete) + MB1703 + MB1708 (SNOR MB1730) (DT) [4.118824] Workqueue: events_unbound deferred_probe_work_func [4.124674] pstate: 60000005 (nZCv daif -PAN -UAO -TCO -DIT -SSBS BTYPE=--) [4.131624] pc : add_dma_entry+0x23c/0x300 [4.135658] lr : add_dma_entry+0x23c/0x300 [4.139792] sp : ffff800009dbb490 [4.143016] x29: ffff800009dbb4a0 x28: 0000000004008022 x27: ffff8000098a6000 [4.150174] x26: 0000000000000000 x25: ffff8000099e7000 x24: ffff8000099e7de8 [4.157231] x23: 00000000ffffffff x22: 0000000000000000 x21: ffff8000098a6a20 [4.164388] x20: ffff000080964180 x19: ffff800009819ba0 x18: 0000000000000006 [4.171545] x17: 6361727420656e69 x16: 6c6568636163203a x15: 72656c6c6f72746e [4.178602] x14: 6f632d646e616e2e x13: ffff800009832f58 x12: 00000000000004ec [4.185759] x11: 00000000000001a4 x10: ffff80000988af58 x9 : ffff800009832f58 [4.192916] x8 : 00000000ffffefff x7 : ffff80000988af58 x6 : 80000000fffff000 [4.199972] x5 : 000000000000bff4 x4 : 0000000000000000 x3 : 0000000000000000 [4.207128] x2 : 0000000000000000 x1 : 0000000000000000 x0 : ffff0000812d2c40 [4.214185] Call trace: [4.216605] add_dma_entry+0x23c/0x300 [4.220338] debug_dma_map_sg+0x198/0x350 [4.224373] __dma_map_sg_attrs+0xa0/0x110 [4.228411] dma_map_sg_attrs+0x10/0x2c [4.232247] stm32_fmc2_nfc_xfer.isra.0+0x1c8/0x3fc [4.237088] stm32_fmc2_nfc_seq_read_page+0xc8/0x174 [4.242127] nand_read_oob+0x1d4/0x8e0 [4.245861] mtd_read_oob_std+0x58/0x84 [4.249596] mtd_read_oob+0x90/0x150 [4.253231] mtd_read+0x68/0xac

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2024-38595	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net/mlx5: Fix peer devlink set for SF representor devlink port The cited patch change register devlink flow, and neglect to reflect the changes for peer devlink set logic. Peer devlink set is triggering a call trace if done after devl_register.[1] Hence, align peer devlink set logic with register devlink flow. [1] WARNING: CPU: 4 PID: 3394 at net/devlink/core.c:155 devlink_rel_nested_in_add+0x177/0x180 CPU: 4 PID: 3394 Comm: kworker/u40:1 Not tainted 6.9.0-rc4_for_linust_min_debug_2024_04_16_14_08 #1 Hardware name: QEMU Standard PC (Q35 + ICH9, 2009), BIOS rel-1.13.0-0-gf21b5a4aeb02-prebuilt.qemu.org 04/01/2014 Workqueue: mlx5_vhca_event0 mlx5_vhca_state_work_handler [mlx5_core] RIP: 0010:devlink_rel_nested_in_add+0x177/0x180 Call Trace: ? __warn+0x78/0x120 ? devlink_rel_nested_in_add+0x177/0x180 ? report_bug+0x16d/0x180 ? handle_bug+0x3c/0x60 ? exc_invalid_op+0x14/0x70 ? asm_exc_invalid_op+0x16/0x20 ? devlink_port_init+0x30/0x30 ? devlink_port_type_clear+0x50/0x50 ? devlink_rel_nested_in_add+0x177/0x180 ? devlink_rel_nested_in_add+0xdd/0x180 mlx5_sf_mdev_event+0x74/0xb0 [mlx5_core] notifier_call_chain+0x35/0xb0 blocking_notifier_call_chain+0x3d/0x60 mlx5_blocking_notifier_call_chain+0x22/0x30 [mlx5_core] mlx5_sf_dev_probe+0x185/0x3e0 [mlx5_core] auxiliary_bus_probe+0x38/0x80 ? driver_sysfs_add+0x51/0x80 really_probe+0xc5/0x3a0 ? driver_probe_device+0x90/0x90 __driver_probe_device+0x80/0x160 driver_probe_device+0x1e/0x90 __device_attach_driver+0x7d/0x100 bus_for_each_drv+0x80/0xd0 __device_attach+0xbc/0x1f0 bus_probe_device+0x86/0xa0 device_add+0x64f/0x860 __auxiliary_device_add+0x3b/0xa0 mlx5_sf_dev_add+0x139/0x330 [mlx5_core] mlx5_sf_dev_state_change_handler+0x1e4/0x250 [mlx5_core] notifier_call_chain+0x35/0xb0 blocking_notifier_call_chain+0x3d/0x60 mlx5_vhca_state_work_handler+0x151/0x200 [mlx5_core] process_one_work+0x13f/0x2e0 worker_thread+0x2bd/0x3c0 ? rescuer_thread+0x410/0x410 kthread+0xc4/0xf0 ? kthread_complete_and_exit+0x20/0x20 ret_from_fork+0x2d/0x50 ? kthread_complete_and_exit+0x20/0x20 ret_from_fork_asm+0x11/0x20

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-37833	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net/niu: Niu requires MSIX ENTRY_DATA fields touch before entry reads Fix niu_try_msix() to not cause a fatal trap on sparc systems. Set PCI_DEV_FLAGS_MSIX_TOUCH_ENTRY_DATA_FIRST on the struct pci_dev to work around a bug in the hardware or firmware. For each vector entry in the msix table, niu chips will cause a fatal trap if any registers in that entry are read before that entries' ENTRY_DATA register is written to. Testing indicates writes to other registers are not sufficient to prevent the fatal trap, however the value does not appear to matter. This only needs to happen once after power up, so simply rebooting into a kernel lacking this fix will NOT cause the trap. NON-RESUMABLE ERROR: Reporting on cpu 64 NON-RESUMABLE ERROR: TPC [0x00000000005f6900] NON-RESUMABLE ERROR: RAW [4010000000000016:00000e37f93e32ff:0000000202000080:ffffffffffffffff NON-RESUMABLE ERROR: 0000000800000000:0000000000000000:0000000000000000:0000000000000000] NON-RESUMABLE ERROR: handle [0x4010000000000016] stick [0x00000e37f93e32ff] NON-RESUMABLE ERROR: type [precise nonresumable] NON-RESUMABLE ERROR: attrs [0x02000080] < ASI sp-faulted priv > NON-RESUMABLE ERROR: raddr [0xffffffffffffffff] NON-RESUMABLE ERROR: insn effective address [0x000000c50020000c] NON-RESUMABLE ERROR: size [0x8] NON-RESUMABLE ERROR: asi [0x00] CPU: 64 UID: 0 PID: 745 Comm: kworker/64:1 Not tainted 6.11.5 #63 Workqueue: events work_for_cpu_fn TSTATE: 0000000011001602 TPC: 00000000005f6900 TNPC: 00000000005f6904 Y: 00000000 Not tainted TPC: g0: 00000000000002e9 g1: 000000000000000c g2: 000000c50020000c g3: 0000000000000100 g4: ffff8000470307c0 g5: ffff800fec5be000 g6: ffff800047a08000 g7: 0000000000000000 o0: ffff800014feb000 o1: ffff800047a0b620 o2: 0000000000000011 o3: ffff800047a0b620 o4: 0000000000000080 o5: 0000000000000011 sp: ffff800047a0ad51 ret_pc: 00000000005f7128 RPC: <__pci_enable_msix_range+0x3cc/0x460> l0: 000000000000000d l1: 000000000000c01f l2: ffff800014feb0a8 l3: 0000000000000020 l4: 000000000000c000 l5: 0000000000000001 l6: 0000000020000000 l7: ffff800047a0b734 i0: ffff800014feb000 i1: ffff800047a0b730 i2: 0000000000000001 i3: 000000000000000d i4: 0000000000000000 i5: 0000000000000000 i6: ffff800047a0ae81 i7: 00000000101888b0 I7: Call Trace: [<00000000101888b0>] niu_try_msix.constprop.0+0xc0/0x130 [niu] [<000000001018f840>] niu_get_invariants+0x183c/0x207c [niu] [<00000000101902fc>] niu_pci_init_one+0x27c/0x2fc [niu] [<00000000005ef3e4>] local_pci_probe+0x28/0x74 [<0000000000469240>] work_for_cpu_fn+0x8/0x1c [<000000000046b008>] process_scheduled_works+0x144/0x210 [<000000000046b518>] worker_thread+0x13c/0x1c0 [<00000000004710e0>] kthread+0xb8/0xc8 [<00000000004060c8>] ret_from_fork+0x1c/0x2c [<0000000000000000>] 0x0 Kernel panic - not syncing: Non-resumable error.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38457	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net/sched: Abort __tc_modify_qdisc if parent class does not exist Lion's patch [1] revealed an ancient bug in the qdisc API. Whenever a user creates/modifies a qdisc specifying as a parent another qdisc, the qdisc API will, during grafting, detect that the user is not trying to attach to a class and reject. However grafting is performed after qdisc_create (and thus the qdiscs' init callback) is executed. In qdiscs that eventually call qdisc_tree_reduce_backlog during init or change (such as fq, hhf, choke, etc), an issue arises. For example, executing the following commands: sudo tc qdisc add dev lo root handle a: htb default 2 sudo tc qdisc add dev lo parent a: handle beef fq Qdiscs such as fq, hhf, choke, etc unconditionally invoke qdisc_tree_reduce_backlog() in their control path init() or change() which then causes a failure to find the child class; however, that does not stop the unconditional invocation of the assumed child qdisc's qlen_notify with a null class. All these qdiscs make the assumption that class is non-null. The solution is ensure that qdisc_leaf() which looks up the parent class, and is invoked prior to qdisc_create(), should return failure on not finding the class. In this patch, we leverage qdisc_leaf to return ERR_PTRs whenever the parentid doesn't correspond to a class, so that we can detect it earlier on and abort before qdisc_create is called. [1] https://lore.kernel.org/netdev/d912cbd7-193b-4269-9857-525bee8bbb6a@gmail.com/
CVE-2026-23105	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net/sched: qfq: Use cl_is_active to determine whether class is active in qfq_rm_from_ag This is more of a preventive patch to make the code more consistent and to prevent possible exploits that employ child qlen manipulations on qfq. use cl_is_active instead of relying on the child qdisc's qlen to determine class activation.
CVE-2026-22999	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net/sched: sch_qfq: do not free existing class in qfq_change_class() Fixes qfq_change_class() error case. cl->qdisc and cl should only be freed if a new class and qdisc were allocated, or we risk various UAF.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2024-56647	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net: Fix icmp host relookup triggering ip_rt_bug arp link failure may trigger ip_rt_bug while xfrm enabled, call trace is: WARNING: CPU: 0 PID: 0 at net/ipv4/route.c:1241 ip_rt_bug+0x14/0x20 Modules linked in: CPU: 0 UID: 0 PID: 0 Comm: swapper/0 Not tainted 6.12.0-rc6-00077-g2e1b3cc9d7f7 Hardware name: QEMU Standard PC (i440FX + PIIX, 1996), BIOS rel-1.13.0-0-gf21b5a4aeb02-prebuilt.qemu.org 04/01/2014 RIP: 0010:ip_rt_bug+0x14/0x20 Call Trace: ip_send_skb+0x14/0x40 __icmp_send+0x42d/0x6a0 ipv4_link_failure+0xe2/0x1d0 arp_error_report+0x3c/0x50 neigh_invalidate+0x8d/0x100 neigh_timer_handler+0x2e1/0x330 call_timer_fn+0x21/0x120 __run_timer_base.part.0+0x1c9/0x270 run_timer_softirq+0x4c/0x80 handle_softirqs+0xac/0x280 irq_exit_rcu+0x62/0x80 sysvec_apic_timer_interrupt+0x77/0x90 The script below reproduces this scenario: ip xfrm policy add src 0.0.0.0/0 dst 0.0.0.0/0 \ dir out priority 0 ptype main flag localok icmp ip l a veth1 type veth ip a a 192.168.141.111/24 dev veth0 ip l s veth0 up ping 192.168.141.155 -c 1 icmp_route_lookup() create input routes for locally generated packets while xfrm relookup ICMP traffic.Then it will set input route (dst->out = ip_rt_bug) to skb for DESTUNREACH. For ICMP err triggered by locally generated packets, dst->dev of output route is loopback. Generally, xfrm relookup verification is not required on loopback interfaces (net.ipv4.conf.lo.disable_xfrm = 1). Skip icmp relookup for locally generated packets to fix it.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-21868	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net: allow small head cache usage with large MAX_SKB_FRAGS values Sabrina reported the following splat: WARNING: CPU: 0 PID: 1 at net/core/dev.c:6935 netif_napi_add_weight_locked+0x8f2/0xba0 Modules linked in: CPU: 0 UID: 0 PID: 1 Comm: swapper/0 Not tainted 6.14.0-rc1-net-00092-g011b03359038 #996 Hardware name: QEMU Standard PC (i440FX + PIIX, 1996), BIOS Arch Linux 1.16.3-1-1 04/01/2014 RIP: 0010:netif_napi_add_weight_locked+0x8f2/0xba0 Code: e8 c3 e6 6a fe 48 83 c4 28 5b 5d 41 5c 41 5d 41 5e 41 5f c3 cc cc cc cc c7 44 24 10 ff ff ff ff e9 8f fb ff ff e8 9e e6 6a fe <0f> 0b e9 d3 fe ff ff e8 92 e6 6a fe 48 8b 04 24 be ff ff ff ff 48 RSP: 0000:ffffc9000001fc60 EFLAGS: 00010293 RAX: 0000000000000000 RBX: ffff88806ce48128 RCX: 1ffff11001664b9e RDX: ffff888008f00040 RSI: ffffffff8317ca42 RDI: ffff88800b325cb6 RBP: ffff88800b325c40 R08: 0000000000000001 R09: ffffed100167502c R10: ffff88800b3a8163 R11: 0000000000000000 R12: ffff88800ac1c168 R13: ffff88800ac1c168 R14: ffff88800ac1c168 R15: 0000000000000007 FS: 0000000000000000(0000) GS:ffff88806ce00000(0000) knlGS:0000000000000000 CS: 0010 DS: 0000 ES: 0000 CR0: 0000000080050033 CR2: ffff888008201000 CR3: 0000000004c94001 CR4: 0000000000370ef0 DR0: 0000000000000000 DR1: 0000000000000000 DR2: 0000000000000000 DR3: 0000000000000000 DR6: 00000000fffe0ff0 DR7: 0000000000000400 Call Trace: gro_cells_init+0x1ba/0x270 xfrm_input_init+0x4b/0x2a0 xfrm_init+0x38/0x50 ip_rt_init+0x2d7/0x350 ip_init+0xf/0x20 inet_init+0x406/0x590 do_one_initcall+0x9d/0x2e0 do_initcalls+0x23b/0x280 kernel_init_freeable+0x445/0x490 kernel_init+0x20/0x1d0 ret_from_fork+0x46/0x80 ret_from_fork_asm+0x1a/0x30 irq event stamp: 584330 hardirqs last enabled at (584338): [] __up_console_sem+0x77/0xb0 hardirqs last disabled at (584345): [] __up_console_sem+0x5c/0xb0 softirqs last enabled at (583242): [] netlink_insert+0x14d/0x470 softirqs last disabled at (583754): [] netif_napi_add_weight_locked+0x77d/0xba0 on kernel built with MAX_SKB_FRAGS=45, where SKB_WITH_OVERHEAD(1024) is smaller than GRO_MAX_HEAD. Such built additionally contains the revert of the single page frag cache so that napi_get_frags() ends up using the page frag allocator, triggering the splat. Note that the underlying issue is independent from the mentioned revert; address it ensuring that the small head cache will fit either TCP and GRO allocation and updating napi_alloc_skb() and __netdev_alloc_skb() to select kmalloc() usage for any allocation fitting such cache.
CVE-2025-38542	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net: appletalk: Fix device refcount leak in atrtr_create() When updating an existing route entry in atrtr_create(), the old device reference was not being released before assigning the new device, leading to a device refcount leak. Fix this by calling dev_put() to release the old device reference before holding the new one.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-21884	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net: better track kernel sockets lifetime While kernel sockets are dismantled during pernet_operations->exit(), their freeing can be delayed by any tx packets still held in qdisc or device queues, due to skb_set_owner_w() prior calls. This then trigger the following warning from ref_tracker_dir_exit() [1] To fix this, make sure that kernel sockets own a reference on net->passive. Add sk_net_refcnt_upgrade() helper, used whenever a kernel socket is converted to a refcounted one. [1] [136.263918][T35] ref_tracker: net notrefcnt@ffff8880638f01e0 has 1/2 users at [136.263918][T35] sk_alloc+0x2b3/0x370 [136.263918][T35] inet6_create+0x6ce/0x10f0 [136.263918][T35] __sock_create+0x4c0/0xa30 [136.263918][T35] inet_ctl_sock_create+0xc2/0x250 [136.263918][T35] igmp6_net_init+0x39/0x390 [136.263918][T35] ops_init+0x31e/0x590 [136.263918][T35] setup_net+0x287/0x9e0 [136.263918][T35] copy_net_ns+0x33f/0x570 [136.263918][T35] create_new_namespaces+0x425/0x7b0 [136.263918][T35] unshare_nsproxy_namespaces+0x124/0x180 [136.263918][T35] ksys_unshare+0x57d/0xa70 [136.263918][T35] __x64_sys_unshare+0x38/0x40 [136.263918][T35] do_syscall_64+0xf3/0x230 [136.263918][T35] entry_SYSCALL_64_after_hwframe+0x77/0x7f [136.263918][T35] [136.343488][T35] ref_tracker: net notrefcnt@ffff8880638f01e0 has 1/2 users at [136.343488][T35] sk_alloc+0x2b3/0x370 [136.343488][T35] inet6_create+0x6ce/0x10f0 [136.343488][T35] __sock_create+0x4c0/0xa30 [136.343488][T35] inet_ctl_sock_create+0xc2/0x250 [136.343488][T35] ndisc_net_init+0xa7/0x2b0 [136.343488][T35] ops_init+0x31e/0x590 [136.343488][T35] setup_net+0x287/0x9e0 [136.343488][T35] copy_net_ns+0x33f/0x570 [136.343488][T35] create_new_namespaces+0x425/0x7b0 [136.343488][T35] unshare_nsproxy_namespaces+0x124/0x180 [136.343488][T35] ksys_unshare+0x57d/0xa70 [136.343488][T35] __x64_sys_unshare+0x38/0x40 [136.343488][T35] do_syscall_64+0xf3/0x230 [136.343488][T35] entry_SYSCALL_64_after_hwframe+0x77/0x7f
CVE-2026-22997	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net: can: j1939: j1939_xtp_rx_rts_session_active(): deactivate session upon receiving the second rts Since j1939_session_deactivate_activate_next() in j1939_tp_rxtimer() is called only when the timer is enabled, we need to call j1939_session_deactivate_activate_next() if we cancelled the timer. Otherwise, refcount for j1939_session leaks, which will later appear as | unregister_netdevice: waiting for vcan0 to become free. Usage count = 2. problem.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38622	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net: drop UFO packets in udp_rcv_segment() When sending a packet with virtio_net_hdr to tun device, if the gso_type in virtio_net_hdr is SKB_GSO_UDP and the gso_size is less than udphdr size, below crash may happen. ------------[cut here]------------ kernel BUG at net/core/skbuff.c:4572! Oops: invalid opcode: 0000 [#1] SMP NOPTI CPU: 0 UID: 0 PID: 62 Comm: mytest Not tainted 6.16.0-rc7 #203 PREEMPT(voluntary) Hardware name: QEMU Standard PC (i440FX + PIIX, 1996), BIOS 1.15.0-1 04/01/2014 RIP: 0010:skb_pull_rcsum+0x8e/0xa0 Code: 00 00 5b c3 cc cc cc cc 8b 93 88 00 00 00 f7 da e8 37 44 38 00 f7 d8 89 83 88 00 00 00 48 8b 83 c8 00 00 00 5b c3 cc cc cc cc <0f> 0b 0f 0b 66 66 2e 0f 1f 84 00 000 RSP: 0018:ffffc900001fba38 EFLAGS: 00000297 RAX: 0000000000000004 RBX: ffff8880040c1000 RCX: ffffc900001fb948 RDX: ffff888003e6d700 RSI: 0000000000000008 RDI: ffff88800411a062 RBP: ffff8880040c1000 R08: 0000000000000000 R09: 0000000000000001 R10: ffff888003606c00 R11: 0000000000000001 R12: 0000000000000000 R13: ffff888004060900 R14: ffff888004050000 R15: ffff888004060900 FS: 000000002406d3c0(0000) GS:ffff888084a19000(0000) knlGS:0000000000000000 CS: 0010 DS: 0000 ES: 0000 CR0: 0000000080050033 CR2: 0000000020000040 CR3: 0000000004007000 CR4: 00000000000006f0 Call Trace: udp_queue_rcv_one_skb+0x176/0x4b0 net/ipv4/udp.c:2445 udp_queue_rcv_skb+0x155/0x1f0 net/ipv4/udp.c:2475 udp_unicast_rcv_skb+0x71/0x90 net/ipv4/udp.c:2626 __udp4_lib_rcv+0x433/0xb00 net/ipv4/udp.c:2690 ip_protocol_deliver_rcu+0xa6/0x160 net/ipv4/ip_input.c:205 ip_local_deliver_finish+0x72/0x90 net/ipv4/ip_input.c:233 ip_sublist_rcv_finish+0x5f/0x70 net/ipv4/ip_input.c:579 ip_sublist_rcv+0x122/0x1b0 net/ipv4/ip_input.c:636 ip_list_rcv+0xf7/0x130 net/ipv4/ip_input.c:670 __netif_receive_skb_list_core+0x21d/0x240 net/core/dev.c:6067 netif_receive_skb_list_internal+0x186/0x2b0 net/core/dev.c:6210 napi_complete_done+0x78/0x180 net/core/dev.c:6580 tun_get_user+0xa63/0x1120 drivers/net/tun.c:1909 tun_chr_write_iter+0x65/0xb0 drivers/net/tun.c:1984 vfs_write+0x300/0x420 fs/read_write.c:593 ksys_write+0x60/0xd0 fs/read_write.c:686 do_syscall_64+0x50/0x1c0 arch/x86/entry/syscall_64.c:63 To trigger gso segment in udp_queue_rcv_skb(), we should also set option UDP_ENCAP_ESPINUDP to enable udp_sk(sk)->encap_rcv. When the encap_rcv hook return 1 in udp_queue_rcv_one_skb(), udp_csum_pull_header() will try to pull udphdr, but the skb size has been segmented to gso size, which leads to this crash. Previous commit cf329aa42b66 ("udp: cope with UDP GRO packet misdirection") introduces segmentation in UDP receive path only for GRO, which was never intended to be used for UFO, so drop UFO packets in udp_rcv_segment().

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38272	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net: dsa: b53: do not enable EEE on bcm63xx BCM63xx internal switches do not support EEE, but provide multiple RGMII ports where external PHYs may be connected. If one of these PHYs are EEE capable, we may try to enable EEE for the MACs, which then hangs the system on access of the (non-existent) EEE registers. Fix this by checking if the switch actually supports EEE before attempting to configure it.
CVE-2024-26596	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net: dsa: fix netdev_priv() dereference before check on non-DSA netdevice events After the blamed commit, we started doing this dereference for every NETDEV_CHANGEUPPER and NETDEV_PRECHANGEUPPER event in the system. static inline struct dsa_port *dsa_user_to_port(const struct net_device *dev) { struct dsa_user_priv *p = netdev_priv(dev); return p->dp; } Which is obviously bogus, because not all net_devices have a netdev_priv() of type struct dsa_user_priv. But struct dsa_user_priv is fairly small, and p->dp means dereferencing 8 bytes starting with offset 16. Most drivers allocate that much private memory anyway, making our access not fault, and we discard the bogus data quickly afterwards, so this wasn't caught. But the dummy interface is somewhat special in that it calls alloc_netdev() with a priv size of 0. So every netdev_priv() dereference is invalid, and we get this when we emit a NETDEV_PRECHANGEUPPER event with a VLAN as its new upper: $ ip link add dummy1 type dummy $ ip link add link dummy1 name dummy1.100 type vlan id 100 [43.309174] ========== ========== [43.316456] BUG: KASAN: slab-out-of-bounds in dsa_user_prechangeupper+0x30/0xe8 [43.323835] Read of size 8 at addr ffff3f86481d2990 by task ip/374 [43.330058] [43.342436] Call trace: [43.366542] dsa_user_prechangeupper+0x30/0xe8 [43.371024] dsa_user_netdevice_event+0xb38/0xee8 [43.375768] notifier_call_chain+0xa4/0x210 [43.379985] raw_notifier_call_chain+0x24/0x38 [43.384464] __netdev_upper_dev_link+0x3ec/0x5d8 [43.389120] netdev_upper_dev_link+0x70/0xa8 [43.393424] register_vlan_dev+0x1bc/0x310 [43.397554] vlan_newlink+0x210/0x248 [43.401247] rtnl_newlink+0x9fc/0xe30 [43.404942] rtnetlink_rcv_msg+0x378/0x580 Avoid the kernel oops by dereferencing after the type check, as customary.
CVE-2024-40999	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net: ena: Add validation for completion descriptors consistency Validate that `first` flag is set only for the first descriptor in multi-buffer packets. In case of an invalid descriptor, a reset will occur. A new reset reason for RX data corruption has been added.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2026-23154	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net: fix segmentation of forwarding fraglist GRO This patch enhances GSO segment handling by properly checking the SKB_GSO_DODGY flag for frag_list GSO packets, addressing low throughput issues observed when a station accesses IPv4 servers via hotspots with an IPv6-only upstream interface. Specifically, it fixes a bug in GSO segmentation when forwarding GRO packets containing a frag_list. The function skb_segment_list cannot correctly process GRO skbs that have been converted by XLAT, since XLAT only translates the header of the head skb. Consequently, skbs in the frag_list may remain untranslated, resulting in protocol inconsistencies and reduced throughput. To address this, the patch explicitly sets the SKB_GSO_DODGY flag for GSO packets in XLAT's IPv4/IPv6 protocol translation helpers (bpf_skb_proto_4_to_6 and bpf_skb_proto_6_to_4). This marks GSO packets as potentially modified after protocol translation. As a result, GSO segmentation will avoid using skb_segment_list and instead falls back to skb_segment for packets with the SKB_GSO_DODGY flag. This ensures that only safe and fully translated frag_list packets are processed by skb_segment_list, resolving protocol inconsistencies and improving throughput when forwarding GRO packets converted by XLAT.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-39770	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net: gso: Forbid IPv6 TSO with extensions on devices with only IPV6_CSUM When performing Generic Segmentation Offload (GSO) on an IPv6 packet that contains extension headers, the kernel incorrectly requests checksum offload if the egress device only advertises NETIF_F_IPV6_CSUM feature, which has a strict contract: it supports checksum offload only for plain TCP or UDP over IPv6 and explicitly does not support packets with extension headers. The current GSO logic violates this contract by failing to disable the feature for packets with extension headers, such as those used in GREoIPv6 tunnels. This violation results in the device being asked to perform an operation it cannot support, leading to a `skb_warn_bad_offload` warning and a collapse of network throughput. While device TSO/USO is correctly bypassed in favor of software GSO for these packets, the GSO stack must be explicitly told not to request checksum offload. Mask NETIF_F_IPV6_CSUM, NETIF_F_TSO6 and NETIF_F_GSO_UDP_L4 in gso_features_check if the IPv6 header contains extension headers to compute checksum in software. The exception is a BIG TCP extension, which, as stated in commit 68e068cabd2c6c53 ("net: reenable NETIF_F_IPV6_CSUM offload for BIG TCP packets"): "The feature is only enabled on devices that support BIG TCP TSO. The header is only present for PF_PACKET taps like tcpdump, and not transmitted by physical devices." kernel log output (truncated): WARNING: CPU: 1 PID: 5273 at net/core/dev.c:3535 skb_warn_bad_offload+0x81/0x140 ... Call Trace: skb_checksum_help+0x12a/0x1f0 validate_xmit_skb+0x1a3/0x2d0 validate_xmit_skb_list+0x4f/0x80 sch_direct_xmit+0x1a2/0x380 __dev_xmit_skb+0x242/0x670 __dev_queue_xmit+0x3fc/0x7f0 ip6_finish_output2+0x25e/0x5d0 ip6_finish_output+0x1fc/0x3f0 ip6_tnl_xmit+0x608/0xc00 [ip6_tunnel] ip6gre_tunnel_xmit+0x1c0/0x390 [ip6_gre] dev_hard_start_xmit+0x63/0x1c0 __dev_queue_xmit+0x6d0/0x7f0 ip6_finish_output2+0x214/0x5d0 ip6_finish_output+0x1fc/0x3f0 ip6_xmit+0x2ca/0x6f0 ip6_finish_output+0x1fc/0x3f0 ip6_xmit+0x2ca/0x6f0 inet6_csk_xmit+0xeb/0x150 __tcp_transmit_skb+0x555/0xa80 tcp_write_xmit+0x32a/0xe90 tcp_sendmsg_locked+0x437/0x1110 tcp_sendmsg+0x2f/0x50 ... skb linear: 00000000: e4 3d 1a 7d ec 30 e4 3d 1a 7e 5d 90 86 dd 60 0e skb linear: 00000010: 00 0a 1b 34 3c 40 20 11 00 00 00 00 00 00 00 00 skb linear: 00000020: 00 00 00 00 00 12 20 11 00 00 00 00 00 00 00 00 skb linear: 00000030: 00 00 00 00 00 00 11 2f 00 04 01 04 01 01 00 00 00 skb linear: 00000040: 86 dd 60 0e 00 0a 1b 00 06 40 20 23 00 00 00 00 skb linear: 00000050: 00 00 00 00 00 00 00 00 00 12 20 23 00 00 00 00 skb linear: 00000060: 00 00 00 00 00 00 00 00 00 11 bf 96 14 51 13 f9 skb linear: 00000070: ae 27 a0 a8 2b e3 80 18 00 40 5b 6f 00 00 01 01 skb linear: 00000080: 08 0a 42 d4 50 d5 4b 70 f8 1a
CVE-2025-22101	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net: libwx: fix Tx L4 checksum The hardware only supports L4 checksum offload for TCP/UDP/SCTP protocol. There was a bug to set Tx checksum flag for the other protocol that results in Tx ring hang. Fix to compute software checksum for these packets.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-39805	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net: macb: fix unregister_netdev call order in macb_remove() When removing a macb device, the driver calls phy_exit() before unregister_netdev(). This leads to a WARN from kernfs: ------------[cut here]------------ kernfs: can not remove 'attached_dev', no directory WARNING: CPU: 1 PID: 27146 at fs/kernfs/dir.c:1683 Call trace: kernfs_remove_by_name_ns+0xd8/0xf0 sysfs_remove_link+0x24/0x58 phy_detach+0x5c/0x168 phy_disconnect+0x4c/0x70 phylink_disconnect_phy+0x6c/0xc0 [phylink] macb_close+0x6c/0x170 [macb] ... macb_remove+0x60/0x168 [macb] platform_remove+0x5c/0x80 ... The warning happens because the PHY is being exited while the netdev is still registered. The correct order is to unregister the netdev before shutting down the PHY and cleaning up the MDIO bus. Fix this by moving unregister_netdev() ahead of phy_exit() in macb_remove().
CVE-2025-38537	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net: phy: Don't register LEDs for genphy If a PHY has no driver, the genphy driver is probed/removed directly in phy_attach/detach. If the PHY's ofnode has an "leds" subnode, then the LEDs will be (un)registered when probing/removing the genphy driver. This could occur if the leds are for a non-generic driver that isn't loaded for whatever reason. Synchronously removing the PHY device in phy_detach leads to the following deadlock: rtnl_lock() ndo_close() ... phy_detach() phy_remove() phy_leds_unregister() led_classdev_unregister() led_trigger_set() netdev_trigger_deactivate() unregister_netdevice_notifier() rtnl_lock() There is a corresponding deadlock on the open/register side of things (and that one is reported by lockdep), but it requires a race while this one is deterministic. Generic PHYs do not support LEDs anyway, so don't bother registering them.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-21801	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net: ravb: Fix missing rtnl lock in suspend/resume path Fix the suspend/resume path by ensuring the rtnl lock is held where required. Calls to ravb_open, ravb_close and wol operations must be performed under the rtnl lock to prevent conflicts with ongoing ndo operations. Without this fix, the following warning is triggered: [39.032969] ============================= [39.032983] WARNING: suspicious RCU usage [39.033019] ----------------------------- [39.033033] drivers/net/phy/phy_device.c:2004 suspicious rcu_dereference_protected() usage! ... [39.033597] stack backtrace: [39.033613] CPU: 0 UID: 0 PID: 174 Comm: python3 Not tainted 6.13.0-rc7-next-20250116-arm64-renesas-00002-g35245dfdc62c #7 [39.033623] Hardware name: Renesas SMARC EVK version 2 based on r9a08g045s33 (DT) [39.033628] Call trace: [39.033633] show_stack+0x14/0x1c (C) [39.033652] dump_stack_lvl+0xb4/0xc4 [39.033664] dump_stack+0x14/0x1c [39.033671] lockdep_rcu_suspicious+0x16c/0x22c [39.033682] phy_detach+0x160/0x190 [39.033694] phy_disconnect+0x40/0x54 [39.033703] ravb_close+0x6c/0x1cc [39.033714] ravb_suspend+0x48/0x120 [39.033721] dpm_run_callback+0x4c/0x14c [39.033731] device_suspend+0x11c/0x4dc [39.033740] dpm_suspend+0xdc/0x214 [39.033748] dpm_suspend_start+0x48/0x60 [39.033758] suspend_devices_and_enter+0x124/0x574 [39.033769] pm_suspend+0x1ac/0x274 [39.033778] state_store+0x88/0x124 [39.033788] kobj_attr_store+0x14/0x24 [39.033798] sysfs_kf_write+0x48/0x6c [39.033808] kernfs_fop_write_iter+0x118/0x1a8 [39.033817] vfs_write+0x27c/0x378 [39.033825] ksys_write+0x64/0xf4 [39.033833] __arm64_sys_write+0x18/0x20 [39.033841] invoke_syscall+0x44/0x104 [39.033852] el0_svc_common.constprop.0+0xb4/0xd4 [39.033862] do_el0_svc+0x18/0x20 [39.033870] el0_svc+0x3c/0xf0 [39.033880] el0t_64_sync_handler+0xc0/0xc4 [39.033888] el0t_64_sync+0x154/0x158 [39.041274] ravb 11c30000.ethernet eth0: Link is Down
CVE-2025-39827	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net: rose: include node references in rose_neigh refcount Current implementation maintains two separate reference counting mechanisms: the 'count' field in struct rose_neigh tracks references from rose_node structures, while the 'use' field (now refcount_t) tracks references from rose_sock. This patch merges these two reference counting systems using 'use' field for proper reference management. Specifically, this patch adds incrementing and decrementing of rose_neigh->use when rose_neigh->count is incremented or decremented. This patch also modifies rose_rt_free(), rose_rt_device_down() and rose_clear_route() to properly release references to rose_neigh objects before freeing a rose_node through rose_remove_node(). These changes ensure rose_neigh structures are properly freed only when all references, including those from rose_node structures, are released. As a result, this resolves a slab-use-after-free issue reported by Syzbot.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-23155	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net: stmmac: Fix accessing freed irq affinity_hint In stmmac_request_irq_multi_msi(), a pointer to the stack variable cpu_mask is passed to irq_set_affinity_hint(). This value is stored in irq_desc->affinity_hint, but once stmmac_request_irq_multi_msi() returns, the pointer becomes dangling. The affinity_hint is exposed via procfs with S_IRUGO permissions, allowing any unprivileged process to read it. Accessing this stale pointer can lead to: - a kernel oops or panic if the referenced memory has been released and unmapped, or - leakage of kernel data into userspace if the memory is re-used for other purposes. All platforms that use stmmac with PCI MSI (Intel, Loongson, etc) are affected.
CVE-2025-71095	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net: stmmac: fix the crash issue for zero copy XDP_TX action There is a crash issue when running zero copy XDP_TX action, the crash log is shown below. [216.122464] Unable to handle kernel paging request at virtual address fffeffff80000000 [216.187524] Internal error: Oops: 0000000096000144 [#1] SMP [216.301694] Call trace: [216.304130] dcache_clean_poc+0x20/0x38 (P) [216.308308] __dma_sync_single_for_device+0x1bc/0x1e0 [216.313351] stmmac_xdp_xmit_xdpf+0x354/0x400 [216.317701] __stmmac_xdp_run_prog+0x164/0x368 [216.322139] stmmac_napi_poll_rxtx+0xba8/0xf00 [216.326576] __napi_poll+0x40/0x218 [216.408054] Kernel panic - not syncing: Oops: Fatal exception in interrupt For XDP_TX action, the xdp_buff is converted to xdp_frame by xdp_convert_buff_to_frame(). The memory type of the resulting xdp_frame depends on the memory type of the xdp_buff. For page pool based xdp_buff it produces xdp_frame with memory type MEM_TYPE_PAGE_POOL. For zero copy XSK pool based xdp_buff it produces xdp_frame with memory type MEM_TYPE_PAGE_ORDER0. However, stmmac_xdp_xmit_back() does not check the memory type and always uses the page pool type, this leads to invalid mappings and causes the crash. Therefore, check the xdp_buff memory type in stmmac_xdp_xmit_back() to fix this issue.
CVE-2025-38125	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net: stmmac: make sure that ptp_rate is not 0 before configuring EST If the ptp_rate recorded earlier in the driver happens to be 0, this bogus value will propagate up to EST configuration, where it will trigger a division by 0. Prevent this division by 0 by adding the corresponding check and error code.
CVE-2025-71094	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net: usb: asix: validate PHY address before use The ASIX driver reads the PHY address from the USB device via asix_read_phy_addr(). A malicious or faulty device can return an invalid address (>= PHY_MAX_ADDR), which causes a warning in mdiobus_get_phy(): addr 207 out of range WARNING: drivers/net/phy/mdio_bus.c:76 Validate the PHY address in asix_read_phy_addr() and remove the now-redundant check in ax88172a.c.
CVE-2025-38470	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net: vlan: fix VLAN 0 refcount imbalance of toggling filtering during runtime Assuming the "rx-vlan-filter" feature is enabled on a net device, the 8021q module will automatically add or remove VLAN 0 when the net device is put administratively up or down, respectively. There are a couple of problems with the above scheme.

CVE	Severity	CVSS	CVSS Vector	Component	Description
					The first problem is a memory leak that can happen if the "rx-vlan-filter" feature is disabled while the device is running: # ip link add bond1 up type bond mode 0 # ethtool -K bond1 rx-vlan-filter off # ip link del dev bond1 When the device is put administratively down the "rx-vlan-filter" feature is disabled, so the 8021q module will not remove VLAN 0 and the memory will be leaked [1]. Another problem that can happen is that the kernel can automatically delete VLAN 0 when the device is put administratively down despite not adding it when the device was put administratively up since during that time the "rx-vlan-filter" feature was disabled. null-ptr-unref or bug_on[2] will be triggered by unregister_vlan_dev() for refcount imbalance if toggling filtering during runtime: $ ip link add bond0 type bond mode 0 $ ip link add link bond0 name vlan0 type vlan id 0 protocol 802.1q $ ethtool -K bond0 rx-vlan-filter off $ ifconfig bond0 up $ ethtool -K bond0 rx-vlan-filter on $ ifconfig bond0 down $ ip link del vlan0 Root cause is as below: step1: add vlan0 for real_dev, such as bond, team. register_vlan_dev vlan_vid_add(real_dev,htons(ETH_P_8021Q),0) //refcnt=1 step2: disable vlan filter feature and enable real_dev step3: change filter from 0 to 1 vlan_device_event vlan_filter_push_vids ndo_vlan_rx_add_vid //No refcnt added to real_dev vlan0 step4: real_dev down vlan_device_event vlan_vid_del(dev, htons(ETH_P_8021Q), 0); //refcnt=0 vlan_info_rcu_free //free vlan0 step5: delete vlan0 unregister_vlan_dev BUG_ON(!vlan_info); //vlan_info is null Fix both problems by noting in the VLAN info whether VLAN 0 was automatically added upon NETDEV_UP and based on that decide whether it should be deleted upon NETDEV_DOWN, regardless of the state of the "rx-vlan-filter" feature. [1] unreferenced object 0xffff8880068e3100 (size 256): comm "ip", pid 384, jiffies 4296130254 hex dump (first 32 bytes): 00 20 30 0d 80 88 ff ff 00 00 00 00 00 00 00 00 . 0..... 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ..... backtrace (crc 81ce31fa): __kmalloc_cache_noprof+0x2b5/0x340 vlan_vid_add+0x434/0x940 vlan_device_event.cold+0x75/0xa8 notifier_call_chain+0xca/0x150 __dev_notify_flags+0xe3/0x250 rtnl_configure_link+0x193/0x260 rtnl_newlink_create+0x383/0x8e0 __rtnl_newlink+0x22c/0xa40 rtnl_newlink+0x627/0xb00 rtnetlink_rcv_msg+0x6fb/0xb70 netlink_rcv_skb+0x11f/0x350 netlink_unicast+0x426/0x710 netlink_sendmsg+0x75a/0xc20 __sock_sendmsg+0xc1/0x150 ____sys_sendmsg+0x5aa/0x7b0 ___sys_sendmsg+0xfc/0x180 [2] kernel BUG at net/8021q/vlan.c:99! Oops: invalid opcode: 0000 [#1] SMP KASAN PTI CPU: 0 UID: 0 PID: 382 Comm: ip Not tainted 6.16.0-rc3 #61 PREEMPT(voluntary) Hardware name: QEMU Standard PC (i440FX + PIIX, 1996), BIOS rel-1.13.0-0-gf21b5a4aeb02-prebuilt.qemu.org 04/01/2014 RIP: 0010:unregister_vlan_dev (net/8021q/vlan.c:99 (discriminator 1)) RSP: 0018:ffff88810badf310 EFLAGS: 00010246 RAX: 0000000000000000 RBX: ffff88810da84000 RCX: ffffffffb47ceb9a RDX: dffffc0000000000 RSI: 0000000000000008 RDI: ffff88810e8b43c8 RBP: 0000000000000000 R08: 0000000000000000 R09: fffffbfff6cefe80 R10: ffffffffb677f407 R11: ffff88810badf3c0 R12: ffff88810e8b4000 R13: 0000000000000000 R14: ffff88810642a5c0 R15: 000000000000017e FS: 00007f1ff68c20c0(0000) GS:ffff888163a24000(0000) knlGS:0000000000000000 CS: 0010 DS: 0000 ES: 0000 CR0: 0000000080050033 CR2: 00007f1ff5dad240 CR3: 0000000107e56000 CR4: 00000000000006f0 Call Trace:

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-39894	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: netfilter: br_netfilter: do not check confirmed bit in br_nf_local_in() after confirm When send a broadcast packet to a tap device, which was added to a bridge, br_nf_local_in() is called to confirm the conntrack. If another conntrack with the same hash value is added to the hash table, which can be triggered by a normal packet to a non-bridge device, the below warning may happen. ------------[cut here]------------ WARNING: CPU: 1 PID: 96 at net/bridge/br_netfilter_hooks.c:632 br_nf_local_in+0x168/0x200 CPU: 1 UID: 0 PID: 96 Comm: tap_send Not tainted 6.17.0-rc2-dirty #44 PREEMPT(voluntary) RIP: 0010:br_nf_local_in+0x168/0x200 Call Trace: nf_hook_slow+0x3e/0xf0 br_pass_frame_up+0x103/0x180 br_handle_frame_finish+0x2de/0x5b0 br_nf_hook_thresh+0xc0/0x120 br_nf_pre_routing_finish+0x168/0x3a0 br_nf_pre_routing+0x237/0x5e0 br_handle_frame+0x1ec/0x3c0 __netif_receive_skb_core+0x225/0x1210 __netif_receive_skb_one_core+0x37/0xa0 netif_receive_skb+0x36/0x160 tun_get_user+0xa54/0x10c0 tun_chr_write_iter+0x65/0xb0 vfs_write+0x305/0x410 ksys_write+0x60/0xd0 do_syscall_64+0xa4/0x260 entry_SYSCALL_64_after_hwframe+0x77/0x7f ---[end trace 0000000000000000]--- To solve the hash conflict, nf_ct_resolve_clash() try to merge the conntracks, and update skb->_nfct. However, br_nf_local_in() still use the old ct from local variable 'nfct' after confirm(), which leads to this warning. If confirm() does not insert the conntrack entry and return NF_DROP, the warning may also occur. There is no need to reserve the WARN_ON_ONCE, just remove it.
CVE-2025-38721	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: netfilter: ctnetlink: fix refcount leak on table dump There is a reference count leak in ctnetlink_dump_table(): if (res < 0) { nf_conntrack_get(&ct->ct_general); // HERE cb->args[1] = (unsigned long)ct; ... While its very unlikely, its possible that ct == last. If this happens, then the refcount of ct was already incremented. This 2nd increment is never undone. This prevents the conntrack object from being released, which in turn keeps prevents cnet->count from dropping back to 0. This will then block the netns dismantle (or conntrack rmmod) as nf_conntrack_cleanup_net_list() will wait forever. This can be reproduced by running conntrack_resize.sh selftest in a loop. It takes ~20 minutes for me on a preemptible kernel on average before I see a runaway kworker spinning in nf_conntrack_cleanup_net_list. One fix would to change this to: if (res < 0) { if (ct != last) nf_conntrack_get(&ct->ct_general); But this reference counting isn't needed in the first place. We can just store a cookie value instead. A followup patch will do the same for ctnetlink_exp_dump_table, it looks to me as if this has the same problem and like ctnetlink_dump_table, we only need a 'skip hint', not the actual object so we can apply the same cookie strategy there as well.
CVE-2026-23139	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: netfilter: nf_conncount: update last_gc only when GC has been performed Currently last_gc is being updated everytime a new connection is tracked, that means that it is updated even if a GC wasn't performed. With a sufficiently high packet rate, it is possible to always bypass the GC, causing the list to grow infinitely. Update the last_gc value only when a GC has been actually performed.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38732	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: netfilter: nf_reject: don't leak dst refcount for loopback packets recent patches to add a WARN() when replacing skb dst entry found an old bug: WARNING: include/linux/skbuff.h:1165 skb_dst_check_unset include/linux/skbuff.h:1164 [inline] WARNING: include/linux/skbuff.h:1165 skb_dst_set include/linux/skbuff.h:1210 [inline] WARNING: include/linux/skbuff.h:1165 nf_reject_fill_skb_dst+0x2a4/0x330 net/ipv4/netfilter/nf_reject_ipv4.c:234 [..] Call Trace: nf_send_unreach+0x17b/0x6e0 net/ipv4/netfilter/nf_reject_ipv4.c:325 nft_reject_inet_eval+0x4bc/0x690 net/netfilter/nft_reject_inet.c:27 expr_call_ops_eval net/netfilter/nf_tables_core.c:237 [inline] .. This is because blamed commit forgot about loopback packets. Such packets already have a dst_entry attached, even at PRE_ROUTING stage. Instead of checking hook just check if the skb already has a route attached to it.
CVE-2025-71160	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: netfilter: nf_tables: avoid chain re-validation if possible Hamza Mahfooz reports cpu soft lock-ups in nft_chain_validate(): watchdog: BUG: soft lockup - CPU#1 stuck for 27s! [iptables-nft-re:37547] [..] RIP: 0010:nft_chain_validate+0xcb/0x110 [nf_tables] [..] nft_immediate_validate+0x36/0x50 [nf_tables] nft_chain_validate+0xc9/0x110 [nf_tables] nft_immediate_validate+0x36/0x50 [nf_tables] nft_chain_validate+0xc9/0x110 [nf_tables] nft_immediate_validate+0x36/0x50 [nf_tables] nft_chain_validate+0xc9/0x110 [nf_tables] nft_immediate_validate+0x36/0x50 [nf_tables] nft_chain_validate+0xc9/0x110 [nf_tables] nft_immediate_validate+0x36/0x50 [nf_tables] nft_chain_validate+0xc9/0x110 [nf_tables] nft_table_validate+0x6b/0xb0 [nf_tables] nf_tables_validate+0x8b/0xa0 [nf_tables] nf_tables_commit+0x1df/0x1eb0 [nf_tables] [..] Currently nf_tables will traverse the entire table (chain graph), starting from the entry points (base chains), exploring all possible paths (chain jumps). But there are cases where we could avoid revalidation. Consider: 1 input -> j2 -> j3 2 input -> j2 -> j3 3 input -> j1 -> j2 -> j3 Then the second rule does not need to revalidate j2, and, by extension j3, because this was already checked during validation of the first rule. We need to validate it only for rule 3. This is needed because chain loop detection also ensures we do not exceed the jump stack: Just because we know that j2 is cycle free, its last jump might now exceed the allowed stack size. We also need to update all reachable chains with the new largest observed call depth. Care has to be taken to revalidate even if the chain depth won't be an issue: chain validation also ensures that expressions are not called from invalid base chains. For example, the masquerade expression can only be called from NAT postrouting base chains. Therefore we also need to keep record of the base chain context (type, hooknum) and revalidate if the chain becomes reachable from a different hook location.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38678	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: netfilter: nf_tables: reject duplicate device on updates A chain/flowtable update with duplicated devices in the same batch is possible. Unfortunately, netdev event path only removes the first device that is found, leaving unregistered the hook of the duplicated device. Check if a duplicated device exists in the transaction batch, bail out with EEXIST in such case. WARNING is hit when unregistering the hook: [49042.221275] WARNING: CPU: 4 PID: 8425 at net/netfilter/core.c:340 nf_hook_entry_head+0xaa/0x150 [49042.221375] CPU: 4 UID: 0 PID: 8425 Comm: nft Tainted: G S 6.16.0+ #170 PREEMPT(full) [...] [49042.221382] RIP: 0010:nf_hook_entry_head+0xaa/0x150
CVE-2025-38162	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: netfilter: nft_set_pipapo: prevent overflow in lookup table allocation When calculating the lookup table size, ensure the following multiplication does not overflow: - desc->field_len[] maximum value is U8_MAX multiplied by NFT_PIPAPO_GROUPS_PER_BYTE(f) that can be 2, worst case. - NFT_PIPAPO_BUCKETS(f->bb) is 2^8, worst case. - sizeof(unsigned long), from sizeof(*f->lt), lt in struct nft_pipapo_field. Then, use check_mul_overflow() to multiply by bucket size and then use check_add_overflow() to the alignment for avx2 (if needed). Finally, add lt_size_check_overflow() helper and use it to consolidate this. While at it, replace leftover allocation using the GFP_KERNEL to GFP_KERNEL_ACCOUNT for consistency, in pipapo_resize().
CVE-2025-38639	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: netfilter: xt_nfacct: don't assume acct name is null-terminated BUG: KASAN: slab-out-of-bounds in .. lib/vsprintf.c:721 Read of size 1 at addr ffff88801eac95c8 by task syz-executor183/5851 [..] string+0x231/0x2b0 lib/vsprintf.c:721 vsnprintf+0x739/0xf00 lib/vsprintf.c:2874 [..] nfacct_mt_checkentry+0xd2/0xe0 net/netfilter/xt_nfacct.c:41 xt_check_match+0x3d1/0xab0 net/netfilter/x_tables.c:523 nfnl_acct_find_get() handles non-null input, but the error printk relied on its presence.
CVE-2025-38400	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: nfs: Clean up /proc/net/rpc/nfs when nfs_fs_proc_net_init() fails. syzbot reported a warning below [1] following a fault injection in nfs_fs_proc_net_init(). [0] When nfs_fs_proc_net_init() fails, /proc/net/rpc/nfs is not removed. Later, rpc_proc_exit() tries to remove /proc/net/rpc, and the warning is logged as the directory is not empty. Let's handle the error of nfs_fs_proc_net_init() properly. [0]: FAULT_INJECTION: forcing a failure. name failslab, interval 1, probability 0, space 0, times 0 CPU: 1 UID: 0 PID: 6120 Comm: syz.2.27 Not tainted 6.16.0-rc1-syzkaller-00010-g2c4a1f3fe03e #0 PREEMPT(full) Hardware name: Google Google Compute Engine/Google Compute Engine, BIOS Google 05/07/2025 Call Trace: dump_stack_lvl (lib/dump_stack.c:123) should_fail_ex (lib/fault-inject.c:73 lib/fault-inject.c:174) should_failslab (mm/failslab.c:46) kmem_cache_alloc_noprof (mm/slub.c:4178 mm/slub.c:4204) __proc_create (fs/proc/generic.c:427) proc_create_reg (fs/proc/generic.c:554) proc_create_net_data (fs/proc/proc_net.c:120) nfs_fs_proc_net_init (fs/nfs/client.c:1409) nfs_net_init (fs/nfs/inode.c:2600) ops_init (net/core/net_namespace.c:138) setup_net (net/core/net_namespace.c:443) copy_net_ns (net/core/net_namespace.c:576) create_new_namespaces (kernel/nsproxy.c:110) unshare_nsproxy_namespaces

CVE	Severity	CVSS	CVSS Vector	Component	Description
					(kernel/nsproxy.c:218 (discriminator 4)) ksys_unshare (kernel/fork.c:3123) __x64_sys_unshare (kernel/fork.c:3190) do_syscall_64 (arch/x86/entry/syscall_64.c:63 arch/x86/entry/syscall_64.c:94) entry_SYSCALL_64_after_hwframe (arch/x86/entry/entry_64.S:130) [1]: remove_proc_entry: removing non-empty directory 'net/rpc', leaking at least 'nfs' WARNING: CPU: 1 PID: 6120 at fs/proc/generic.c:727 remove_proc_entry+0x45e/0x530 fs/proc/generic.c:727 Modules linked in: CPU: 1 UID: 0 PID: 6120 Comm: syz.2.27 Not tainted 6.16.0-rc1-syzkaller-00010-g2c4a1f3fe03e #0 PREEMPT(full) Hardware name: Google Google Compute Engine/Google Compute Engine, BIOS Google 05/07/2025 RIP: 0010:remove_proc_entry+0x45e/0x530 fs/proc/generic.c:727 Code: 3c 02 00 0f 85 85 00 00 00 48 8b 93 d8 00 00 00 4d 89 f0 4c 89 e9 48 c7 c6 40 ba a2 8b 48 c7 c7 60 b9 a2 8b e8 33 81 1d ff 90 <0f> 0b 90 90 e9 5f fe ff ff e8 04 69 5e ff 90 48 b8 00 00 00 00 00 RSP: 0018:ffffc90003637b08 EFLAGS: 00010282 RAX: 0000000000000000 RBX: ffff88805f534140 RCX: ffffffff817a92c8 RDX: ffff88807da99e00 RSI: ffffffff817a92d5 RDI: 0000000000000001 RBP: ffff888033431ac0 R08: 0000000000000001 R09: 0000000000000000 R10: 0000000000000001 R11: 0000000000000001 R12: ffff888033431a00 R13: ffff888033431ae4 R14: ffff888033184724 R15: dfffffc0000000000 FS: 0000555580328500(0000) GS:ffff888124a62000(0000) knlGS:0000000000000000 CS: 0010 DS: 0000 ES: 0000 CR0: 0000000080050033 CR2: 00007f71733743e0 CR3: 000000007f618000 CR4: 00000000003526f0 DR0: 0000000000000000 DR1: 0000000000000000 DR2: 0000000000000000 DR3: 0000000000000000 DR6: 00000000fffe0ff0 DR7: 0000000000000400 Call Trace: sunrpc_exit_net+0x46/0x90 net/sunrpc/sunrpc_syms.c:76 ops_exit_list net/core/net_namespace.c:200 [inline] ops_undo_list+0x2eb/0xab0 net/core/net_namespace.c:253 setup_net+0x2e1/0x510 net/core/net_namespace.c:457 copy_net_ns+0x2a6/0x5f0 net/core/net_namespace.c:574 create_new_namespaces+0x3ea/0xa90 kernel/nsproxy.c:110 unshare_nsproxy_namespaces+0xc0/0x1f0 kernel/nsproxy.c:218 ksys_unshare+0x45b/0xa40 kernel/fork.c:3121 __do_sys_unshare kernel/fork.c:3192 [inline] __se_sys_unshare kernel/fork.c:3190 [inline] __x64_sys_unshare+0x31/0x40 kernel/fork.c:3190 do_syscall_x64 arch/x86/entry/syscall_64.c:63 [inline] do_syscall_64+0xcd/0x490 arch/x86/entry/syscall_64.c:94 entry_SYSCALL_64_after_hwframe+0x77/0x7f RIP: 0033:0x7fa1a6b8e929 Code: ff ff c3 66 2e 0f 1f 84 00 00 00 00 00 0f 1f 40 00 48 89 f8 48 89 f7 48 89 d6 48 89 ca 4d 89 c2 4d 89 c8 4c 8b 4c 24 08 0f 05 <48> 3d 01 f0 ff ff 73 01 c ---truncated---

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-71237	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: nilfs2: Fix potential block overflow that cause system hang When a user executes the FITRIM command, an underflow can occur when calculating nblocks if end_block is too small. Since nblocks is of type sector_t, which is u64, a negative nblocks value will become a very large positive integer. This ultimately leads to the block layer function __blkdev_issue_discard() taking an excessively long time to process the bio chain, and the ns_segctor_sem lock remains held for a long period. This prevents other tasks from acquiring the ns_segctor_sem lock, resulting in the hang reported by syzbot in [1]. If the ending block is too small, typically if it is smaller than 4KiB range, depending on the usage of the segment 0, it may be possible to attempt a discard request beyond the device size causing the hang. Exiting successfully and assign the discarded size (0 in this case) to range->len. Although the start and len values in the user input range are too small, a conservative strategy is adopted here to safely ignore them, which is equivalent to a no-op; it will not perform any trimming and will not throw an error. [1] task:segctord state:D stack:28968 pid:6093 tgid:6093 ppid:2 task_flags:0x200040 flags:0x00080000 Call Trace: rwbase_write_lock+0x3dd/0x750 kernel/locking/rwbase_rt.c:272 nilfs_transaction_lock+0x253/0x4c0 fs/nilfs2/segment.c:357 nilfs_segctor_thread_construct fs/nilfs2/segment.c:2569 [inline] nilfs_segctor_thread+0x6ec/0xe00 fs/nilfs2/segment.c:2684 [ryusuke: corrected part of the commit message about the consequences]
CVE-2025-38663	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: nilfs2: reject invalid file types when reading inodes To prevent inodes with invalid file types from tripping through the vfs and causing malfunctions or assertion failures, add a missing sanity check when reading an inode from a block device. If the file type is not valid, treat it as a filesystem error.
CVE-2024-41082	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: nvme-fabrics: use reserved tag for reg read/write command In some scenarios, if too many commands are issued by nvme command in the same time by user tasks, this may exhaust all tags of admin_q. If a reset (nvme reset or IO timeout) occurs before these commands finish, reconnect routine may fail to update nvme regs due to insufficient tags, which will cause kernel hang forever. In order to workaround this issue, maybe we can let reg_read32()/reg_read64()/reg_write32() use reserved tags. This maybe safe for nvmf: 1. For the disable ctrl path, we will not issue connect command 2. For the enable ctrl / fw activate path, since connect and reg_xx() are called serially. So the reserved tags may still be enough while reg_xx() use reserved tags.
CVE-2025-38264	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: nvme-tcp: sanitize request list handling Validate the request in nvme_tcp_handle_r2t() to ensure it's not part of any list, otherwise a malicious R2T PDU might inject a loop in request list processing.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-39842	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ocfs2: prevent release journal inode after journal shutdown Before calling ocfs2_delete_osb(), ocfs2_journal_shutdown() has already been executed in ocfs2_dismount_volume(), so osb->journal must be NULL. Therefore, the following calltrace will inevitably fail when it reaches jbd2_journal_release_jbd_inode(). ocfs2_dismount_volume()-> ocfs2_delete_osb()-> ocfs2_free_slot_info()-> __ocfs2_free_slot_info()-> evict()-> ocfs2_evict_inode()-> ocfs2_clear_inode()-> jbd2_journal_release_jbd_inode(osb->journal->j_journal, Adding osb->journal checks will prevent null-ptr-deref during the above execution path.
CVE-2025-71121	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: parisc: Do not reprogram affinitiy on ASP chip The ASP chip is a very old variant of the GSP chip and is used e.g. in HP 730 workstations. When trying to reprogram the affinity it will crash with a HPMC as the relevant registers don't seem to be at the usual location. Let's avoid the crash by checking the sversion. Also note, that reprogramming isn't necessary either, as the HP730 is a just a single-CPU machine.
CVE-2025-39781	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: parisc: Drop WARN_ON_ONCE() from flush_cache_vmap I have observed warning to occassionally trigger.
CVE-2025-39716	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: parisc: Revise __get_user() to probe user read access Because of the way read access support is implemented, read access interruptions are only triggered at privilege levels 2 and 3. The kernel executes at privilege level 0, so __get_user() never triggers a read access interruption (code 26). Thus, it is currently possible for user code to access a read protected address via a system call. Fix this by probing read access rights at privilege level 3 (PRIV_USER) and setting __gu_err to -EFAULT (-14) if access isn't allowed. Note the cmpiclr instruction does a 32-bit compare because COND macro doesn't work inside asm.
CVE-2025-39715	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: parisc: Revise gateway LWS calls to probe user read access We use load and stbys,e instructions to trigger memory reference interruptions without writing to memory. Because of the way read access support is implemented, read access interruptions are only triggered at privilege levels 2 and 3. The kernel and gateway page execute at privilege level 0, so this code never triggers a read access interruption. Thus, it is currently possible for user code to execute a LWS compare and swap operation at an address that is read protected at privilege level 3 (PRIV_USER). Fix this by probing read access rights at privilege level 3 and branching to lws_fault if access isn't allowed.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38322	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: perf/x86/intel: Fix crash in icl_update_topdown_event() The perf_fuzzer found a hard-lockup crash on a RaptorLake machine: Oops: general protection fault, maybe for address 0xffff89aeceab400: 0000 CPU: 23 UID: 0 PID: 0 Comm: swapper/23 Tainted: [W]=WARN Hardware name: Dell Inc. Precision 9660/0VJ762 RIP: 0010:native_read_pmc+0x7/0x40 Code: cc e8 8d a9 01 00 48 89 03 5b cd cc cc cc cc 0f 1f ... RSP: 000:fffb03100273de8 EFLAGS: 00010046 .... Call Trace: icl_update_topdown_event+0x165/0x190 ? ktime_get+0x38/0xd0 intel_pmu_read_event+0xf9/0x210 __perf_event_read+0xf9/0x210 CPUs 16-23 are E-core CPUs that don't support the perf metrics feature. The icl_update_topdown_event() should not be invoked on these CPUs. It's a regression of commit: f9bdf1f95339 ("perf/x86/intel: Avoid disable PMU if !cpuc->enabled in sample read") The bug introduced by that commit is that the is_topdown_event() function is mistakenly used to replace the is_topdown_count() call to check if the topdown functions for the perf metrics feature should be invoked. Fix it.
CVE-2025-37747	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: perf: Fix hang while freeing sigtrap event Perf can hang while freeing a sigtrap event if a related deferred signal hadn't managed to be sent before the file got closed: perf_event_overflow() task_work_add(perf_pending_task) fput() task_work_add(____fput()) task_work_run() ____fput() perf_release() perf_event_release_kernel() _free_event() perf_pending_task_sync() task_work_cancel() -> FAILED rcuwait_wait_event() Once task_work_run() is running, the list of pending callbacks is removed from the task_struct and from this point on task_work_cancel() can't remove any pending and not yet started work items, hence the task_work_cancel() failure and the hang on rcuwait_wait_event(). Task work could be changed to remove one work at a time, so a work running on the current task can always cancel a pending one, however the wait / wake design is still subject to inverted dependencies when remote targets are involved, as pictured by Oleg: T1 T2 fd = perf_event_open(pid => T2->pid); fd = perf_event_open(pid => T1->pid); close(fd) close(fd) perf_event_overflow() perf_event_overflow() task_work_add(perf_pending_task) task_work_add(perf_pending_task) fput() fput() task_work_add(____fput()) task_work_add(____fput()) task_work_run() task_work_run() ____fput() ____fput() perf_release() perf_release() perf_event_release_kernel() perf_event_release_kernel() _free_event() _free_event() perf_pending_task_sync() perf_pending_task_sync() rcuwait_wait_event() rcuwait_wait_event() Therefore the only option left is to acquire the event reference count upon queueing the perf task work and release it from the task work, just like it was done before 3a5465418f5f ("perf: Fix event leak upon exec and file release") but without the leaks it fixed. Some adjustments are necessary to make it work: * A child event might dereference its parent upon freeing. Care must be taken to release the parent last. * Some places assuming the event doesn't have any reference held and therefore can be freed right away must instead put the reference and let the reference counting to its job.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38466	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: perf: Revert to requiring CAP_SYS_ADMIN for uprobes Jann reports that uprobes can be used destructively when used in the middle of an instruction. The kernel only verifies there is a valid instruction at the requested offset, but due to variable instruction length cannot determine if this is an instruction as seen by the intended execution stream. Additionally, Mark Rutland notes that on architectures that mix data in the text segment (like arm64), a similar things can be done if the data word is 'mistaken' for an instruction. As such, require CAP_SYS_ADMIN for uprobes.
CVE-2025-38412	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: platform/x86: dell-wmi-sysman: Fix WMI data block retrieval in sysfs callbacks After retrieving WMI data blocks in sysfs callbacks, check for the validity of them before dereferencing their content.
CVE-2026-23131	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: platform/x86: hp-bioscfg: Fix kobject warnings for empty attribute names The hp-bioscfg driver attempts to register kobjects with empty names when the HP BIOS returns attributes with empty name strings. This causes multiple kernel warnings: kobject: (00000000135fb5e6): attempted to be registered with empty name! WARNING: CPU: 14 PID: 3336 at lib/kobject.c:219 kobject_add_internal+0x2eb/0x310 Add validation in hp_init_bios_buffer_attribute() to check if the attribute name is empty after parsing it from the WMI buffer. If empty, log a debug message and skip registration of that attribute, allowing the module to continue processing other valid attributes.
CVE-2026-23116	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: pmdomain: imx8m-blk-ctrl: Remove separate rst and clk mask for 8mq vpu For i.MX8MQ platform, the ADB in the VPUMIX domain has no separate reset and clock enable bits, but is ungated and reset together with the VPUs. So we can't reset G1 or G2 separately, it may led to the system hang. Remove rst_mask and clk_mask of imx8mq_vpu_blk_ctl_domain_data. Let imx8mq_vpu_power_notifier() do really vpu reset.
CVE-2025-38576	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: powerpc/eeh: Make EEH driver device hotplug safe Multiple race conditions existed between the PCIe hotplug driver and the EEH driver, leading to a variety of kernel oopses of the same general nature: A second class of oops is also seen when the underlying bus disappears during device recovery. Refactor the EEH module to be PCI rescan and remove safe. Also clean up a few minor formatting / readability issues.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-71119	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: powerpc/kexec: Enable SMT before waking offline CPUs If SMT is disabled or a partial SMT state is enabled, when a new kernel image is loaded for kexec, on reboot the following warning is observed: kexec: Waking offline cpu 228. WARNING: CPU: 0 PID: 9062 at arch/powerpc/kexec/core_64.c:223 kexec_prepare_cpus+0x1b0/0x1bc [snip] NIP kexec_prepare_cpus+0x1b0/0x1bc LR kexec_prepare_cpus+0x1a0/0x1bc Call Trace: kexec_prepare_cpus+0x1a0/0x1bc (unreliable) default_machine_kexec+0x160/0x19c machine_kexec+0x80/0x88 kernel_kexec+0xd0/0x118 __do_sys_reboot+0x210/0x2c4 system_call_exception+0x124/0x320 system_call_vectored_common+0x15c/0x2ec This occurs as add_cpu() fails due to cpu_bootable() returning false for CPUs that fail the cpu_smt_thread_allowed() check or non primary threads if SMT is disabled. Fix the issue by enabling SMT and resetting the number of SMT threads to the number of threads per core, before attempting to wake up all present CPUs.
CVE-2024-57999	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: powerpc/pseries/iommu: IOMMU incorrectly marks MMIO range in DDW Power Hypervisor can possibly allocate MMIO window intersecting with Dynamic DMA Window (DDW) range, which is over 32-bit addressing. These MMIO pages needs to be marked as reserved so that IOMMU doesn't map DMA buffers in this range. The current code is not marking these pages correctly which is resulting in LPAR to OOPS while booting. The stack is at below BUG: Unable to handle kernel data access on read at 0xc00800005cd40000 Faulting instruction address: 0xc00000000005cdac Oops: Kernel access of bad area, sig: 11 [#1] LE PAGE_SIZE=64K MMU=Hash SMP NR_CPUS=2048 NUMA pSeries Modules linked in: af_packet rfkill ibmveth(X) lpfc(+) nvmet_fc nvmet nvme_keyring crct10dif_vpmsum nvme_fc nvme_fabrics nvme_core be2net(+) nvme_auth rtc_generic nfsd auth_rpcgss nfs_acl lockd grace sunrpc fuse configfs ip_tables x_tables xfs libcrc32c dm_service_time ibmvfc(X) scsi_transport_fc vmx_crypto gf128mul crc32c_vpmsum dm_mirror dm_region_hash dm_log dm_multipath dm_mod sd_mod scsi_dh_emc scsi_dh_rdac scsi_dh_alua t10_pi crc64_rocksoft_generic crc64_rocksoft sg crc64 scsi_mod Supported: Yes, External CPU: 8 PID: 241 Comm: kworker/8:1 Kdump: loaded Not tainted 6.4.0-150600.23.14-default #1 SLE15-SP6 b44ee71c81261b9e4bab5e0cde1f2ed891d5359b Hardware name: IBM,9080-M9S POWER9 (raw) 0x4e2103 0xf000005 of:IBM,FW950.B0 (VH950_149) hv:phyp pSeries Workqueue: events work_for_cpu_fn NIP: c00000000005cdac LR: c00000000005e830 CTR: 0000000000000000 REGS: c00001400c9ff770 TRAP: 0300 Not tainted (6.4.0-150600.23.14-default) MSR: 800000000280b033 CR: 24228448 XER: 00000001 CFAR: c00000000005cdd4 DAR: c00800005cd40000 DSISR: 40000000 IRQMASK: 0 GPR00: c00000000005e830 c00001400c9ffa10 c000000001987d00 c00001400c4fe800 GPR04: 0000080000000000 0000000000000001 0000000004000000 0000000000800000 GPR08: 0000000004000000 0000000000000001 c00800005cd40000 ffffffffffffffff GPR12: 0000000084228882 c00000000a4c4f00 0000000000000010 0000080000000000 GPR16: c00001400c4fe800 0000000004000000 0800000000000000 c00000006088b800 GPR20: c00001401a7be980 c00001400eff3800 c000000002a2da68 000000000000002b GPR24: c00000000026793a8 c000000002679368 000000000000002a

CVE	Severity	CVSS	CVSS Vector	Component	Description
					c0000000026793c8 GPR28: 000008007effffff 0000080000000000 0000000000800000 c00001400c4fe800 NIP [c00000000005cdac] iommu_table_reserve_pages+0xac/0x100 LR [c00000000005e830] iommu_init_table+0x80/0x1e0 Call Trace: [c00001400c9ffa10] [c00000000005e810] iommu_init_table+0x60/0x1e0 (unreliable) [c00001400c9ffa90] [c00000000010356c] iommu_bypass_supported_pSeriesLP+0x9cc/0xe40 [c00001400c9ffc30] [c00000000005c300] dma_iommu_dma_supported+0xf0/0x230 [c00001400c9ffcb0] [c00000000024b0c4] dma_supported+0x44/0x90 [c00001400c9ffcd0] [c00000000024b14c] dma_set_mask+0x3c/0x80 [c00001400c9ffd00] [c0080000555b715c] be_probe+0xc4/0xb90 [be2net] [c00001400c9ffdc0] [c000000000986f3c] local_pci_probe+0x6c/0x110 [c00001400c9ffe40] [c000000000188f28] work_for_cpu_fn+0x38/0x60 [c00001400c9ffe70] [c00000000018e454] process_one_work+0x314/0x620 [c00001400c9fff10] [c00000000018f280] worker_thread+0x2b0/0x620 [c00001400c9fff90] [c000000000019bb18] kthread+0x148/0x150 [c00001400c9fffe0] [c00000000000ded8] start_kernel_thread+0x14/0x18 There are 2 issues in the code 1. The index is "int" while the address is "unsigned long". This results in negative value when setting the bitmap. 2. The DMA offset is page shifted but the MMIO range is used as-is (64-bit address). MMIO address needs to be page shifted as well.
CVE-2025-39949	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: qed: Don't collect too many protection override GRC elements In the protection override dump path, the firmware can return far too many GRC elements, resulting in attempting to write past the end of the previously-kmalloc'ed dump buffer. This will result in a kernel panic with reason: BUG: unable to handle kernel paging request at ADDRESS where "ADDRESS" is just past the end of the protection override dump buffer. The start address of the buffer is: p_hwfn->cdev->dbg_features[DBG_FEATURE_PROTECTION_OVERRIDE].dump_buf and the size of the buffer is buf_size in the same data structure. The panic can be arrived at from either the qede Ethernet driver path: [exception RIP: qed_grc_dump_addr_range+0x108] qed_protection_override_dump at ffffffffc02662ed [qed] qed_dbg_protection_override_dump at ffffffffc0267792 [qed] qed_dbg_feature at ffffffffc026aa8f [qed] qed_dbg_all_data at ffffffffc026b211 [qed] qed_fw_fatal_reporter_dump at ffffffffc027298a [qed] devlink_health_do_dump at ffffffff82497f61 devlink_health_report at ffffffff8249cf29 qed_report_fatal_error at ffffffffc0272baf [qed] qede_sp_task at ffffffffc045ed32 [qede] process_one_work at ffffffff81d19783 or the qedf storage driver path: [exception RIP: qed_grc_dump_addr_range+0x108] qed_protection_override_dump at ffffffffc068b2ed [qed] qed_dbg_protection_override_dump at ffffffffc068c792 [qed] qed_dbg_feature at ffffffffc068fa8f [qed] qed_dbg_all_data at ffffffffc0690211 [qed] qed_fw_fatal_reporter_dump at ffffffffc069798a [qed] devlink_health_do_dump at ffffffff8aa95e51 devlink_health_report at ffffffff8aa9ae19 qed_report_fatal_error at ffffffffc0697baf [qed] qed_hw_err_notify at ffffffffc06d32d7 [qed] qed_spq_post at ffffffffc06b1011 [qed] qed_fcoe_destroy_conn at ffffffffc06b2e91 [qed] qedf_cleanup_fcport at ffffffffc05e7597 [qedf] qedf_rport_event_handler at ffffffffc05e7bf7 [qedf] fc_rport_work at ffffffffc02da715 [libfc] process_one_work at ffffffff8a319663 Resolve this by clamping the firmware's return value to the maximum number of legal elements the firmware should return.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2024-49926	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: rcu-tasks: Fix access non-existent percpu rtpcp variable in rcu_tasks_need_gpcb() For kernels built with CONFIG_FORCE_NR_CPUS=y, the nr_cpu_ids is defined as NR_CPUS instead of the number of possible cpus, this will cause the following system panic: smpboot: Allowing 4 CPUs, 0 hotplug CPUs ... setup_percpu: NR_CPUS:512 nr_cpumask_bits:512 nr_cpu_ids:512 nr_node_ids:1 ... BUG: unable to handle page fault for address: ffffffff9911c8c8 Oops: 0000 [#1] PREEMPT SMP PTI CPU: 0 PID: 15 Comm: rcu_tasks_trace Tainted: G W 6.6.21 #1 5dc7acf91a5e8e9ac9dcfc35bee0245691283ea6 RIP: 0010:rcu_tasks_need_gpcb+0x25d/0x2c0 RSP: 0018:ffffa371c00a3e60 EFLAGS: 00010082 CR2: ffffffff9911c8c8 CR3: 000000040fa20005 CR4: 00000000001706f0 Call Trace: ? __die+0x23/0x80 ? page_fault_oops+0xa4/0x180 ? exc_page_fault+0x152/0x180 ? asm_exc_page_fault+0x26/0x40 ? rcu_tasks_need_gpcb+0x25d/0x2c0 ? __pfx_rcu_tasks_kthread+0x40/0x40 rcu_tasks_one_gp+0x69/0x180 rcu_tasks_kthread+0x94/0xc0 kthread+0xe8/0x140 ? __pfx_kthread+0x40/0x40 ret_from_fork+0x34/0x80 ? __pfx_kthread+0x40/0x40 ret_from_fork_asm+0x1b/0x80 Considering that there may be holes in the CPU numbers, use the maximum possible cpu number, instead of nr_cpu_ids, for configuring enqueue and dequeue limits. [neeraj.upadhyay: Fix htmldocs build error reported by Stephen Rothwell]
CVE-2025-39745	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: rcutorture: Fix rcutorture_one_extend_check() splat in RT kernels For built with CONFIG_PREEMPT_RT=y kernels, running rcutorture tests resulted in the following splat: [68.797425] rcutorture_one_extend_check during change: Current 0x1 To add 0x1 To remove 0x0 preempt_count() 0x0 [68.797533] WARNING: CPU: 2 PID: 512 at kernel/rcu/rcutorture.c:1993 rcutorture_one_extend_check+0x419/0x560 [rcutorture] [68.797601] Call Trace: [68.797602] [68.797619] ? lockdep_softirqs_off+0xa5/0x160 [68.797631] rcutorture_one_extend+0x18e/0xcc0 [rcutorture 2466dbd2ff34dbaa36049cb323a80c3306ac997c] [68.797646] ? local_clock+0x19/0x40 [68.797659] rcu_torture_one_read+0xf0/0x280 [rcutorture 2466dbd2ff34dbaa36049cb323a80c3306ac997c] [68.797678] ? __pfx_rcu_torture_one_read+0x10/0x10 [rcutorture 2466dbd2ff34dbaa36049cb323a80c3306ac997c] [68.797804] ? __pfx_rcu_torture_timer+0x10/0x10 [rcutorture 2466dbd2ff34dbaa36049cb323a80c3306ac997c] [68.797815] rcu-torture: rcu_torture_reader task started [68.797824] rcu-torture: Creating rcu_torture_reader task [68.797824] rcu_torture_reader+0x238/0x580 [rcutorture 2466dbd2ff34dbaa36049cb323a80c3306ac997c] [68.797836] ? kvm_sched_clock_read+0x15/0x30 Disable BH does not change the SOFTIRQ corresponding bits in preempt_count() for RT kernels, this commit therefore use softirq_count() to check the if BH is disabled.
CVE-2024-50137	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: reset: starfive: jh71x0: Fix accessing the empty member on JH7110 SoC data->asserted will be NULL on JH7110 SoC since commit 82327b127d41 ("reset: starfive: Add StarFive JH7110 reset driver") was added. Add the judgment condition to avoid errors when calling reset_control_status on JH7110 SoC.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38261	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: riscv: save the SR_SUM status over switches When threads/tasks are switched we need to ensure the old execution's SR_SUM state is saved and the new thread has the old SR_SUM state restored. The issue was seen under heavy load especially with the syz-stress tool running, with crashes as follows in schedule_tail: Unable to handle kernel access to user memory without uaccess routines at virtual address 0000000002749f0d0 Oops [#1] Modules linked in: CPU: 1 PID: 4875 Comm: syz-executor.0 Not tainted 5.12.0-rc2-syzkaller-00467-g0d7588ab9ef9 #0 Hardware name: riscv-virtio,qemu (DT) epc : schedule_tail+0x72/0xb2 kernel/sched/core.c:4264 ra : task_pid_vnr include/linux/sched.h:1421 [inline] ra : schedule_tail+0x70/0xb2 kernel/sched/core.c:4264 epc : ffffffe00008c8b0 ra : ffffffe00008c8ae sp : ffffffe025d17ec0 gp : ffffffe005d25378 tp : ffffffe00f0d0000 t0 : 0000000000000000 t1 : 0000000000000001 t2 : 00000000000f4240 s0 : ffffffe025d17ee0 s1 : 0000000002749f0d0 a0 : 000000000000002a a1 : 0000000000000003 a2 : 1fffffffc0cfac500 a3 : ffffffe0000c80cc a4 : 5ae9db91c19bbe00 a5 : 0000000000000000 a6 : 0000000000f00000 a7 : ffffffe000082eba s2 : 0000000000040000 s3 : ffffffe00eef96c0 s4 : ffffffe022c77fe0 s5 : 0000000000004000 s6 : ffffffe067d74e00 s7 : ffffffe067d74850 s8 : ffffffe067d73e18 s9 : ffffffe067d74e00 s10: ffffffe00eef96e8 s11: 000000ae6cdf8368 t3 : 5ae9db91c19bbe00 t4 : ffffffc4043cafb2 t5 : ffffffc4043cafba t6 : 0000000000040000 status: 0000000000000120 badaddr: 0000000002749f0d0 cause: 000000000000000f Call Trace: [] schedule_tail+0x72/0xb2 kernel/sched/core.c:4264 [] ret_from_exception+0x0/0x14 Dumping ftrace buffer: (ftrace buffer empty) ---[end trace b5f8f9231dc87dda]--- The issue comes from the put_user() in schedule_tail (kernel/sched/core.c) doing the following: asmlinkage __visible void schedule_tail(struct task_struct *prev) { ... if (current->set_child_tid) put_user(task_pid_vnr(current), current->set_child_tid); ... } the put_user() macro causes the code sequence to come out as follows: 1: __enable_user_access() 2: reg = task_pid_vnr(current); 3: *current->set_child_tid = reg; 4: __disable_user_access() The problem is that we may have a sleeping function as argument which could clear SR_SUM causing the panic above. This was fixed by evaluating the argument of the put_user() macro outside the user-enabled section in commit 285a76bb2cf5 ("riscv: evaluate put_user() arg before enabling user access")" In order for riscv to take advantage of unsafe_get/put_XXX() macros and to avoid the same issue we had with put_user() and sleeping functions we must ensure code flow can go through switch_to() from within a region of code with SR_SUM enabled and come back with SR_SUM still enabled. This patch addresses the problem allowing future work to enable full use of unsafe_get/put_XXX() macros without needing to take a CSR bit flip cost on every access. Make switch_to() save and restore SR_SUM.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38067	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: rseq: Fix segfault on registration when rseq_cs is non-zero The rseq_cs field is documented as being set to 0 by user-space prior to registration, however this is not currently enforced by the kernel. This can result in a segfault on return to user-space if the value stored in the rseq_cs field doesn't point to a valid struct rseq_cs. The correct solution to this would be to fail the rseq registration when the rseq_cs field is non-zero. However, some older versions of glibc will reuse the rseq area of previous threads without clearing the rseq_cs field and will also terminate the process if the rseq registration fails in a secondary thread. This wasn't caught in testing because in this case the leftover rseq_cs does point to a valid struct rseq_cs. What we can do is clear the rseq_cs field on registration when it's non-zero which will prevent segfaults on registration and won't break the glibc versions that reuse rseq areas on thread creation.
CVE-2025-38514	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: rxrpc: Fix oops due to non-existence of prealloc backlog struct If an AF_RXRPC service socket is opened and bound, but calls are preallocated, then rxrpc_alloc_incoming_call() will oops because the rxrpc_backlog struct doesn't get allocated until the first preallocation is made. Fix this by returning NULL from rxrpc_alloc_incoming_call() if there is no backlog struct. This will cause the incoming call to be aborted.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2024-53128	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: sched/task_stack: fix object_is_on_stack() for KASAN tagged pointers When CONFIG_KASAN_SW_TAGS and CONFIG_KASAN_STACK are enabled, the object_is_on_stack() function may produce incorrect results due to the presence of tags in the obj pointer, while the stack pointer does not have tags. This discrepancy can lead to incorrect stack object detection and subsequently trigger warnings if CONFIG_DEBUG_OBJECTS is also enabled. Example of the warning: ODEBUG: object 3eff800082ea7bb0 is NOT on stack ffff800082ea0000, but annotated. ----------[cut here]---------- WARNING: CPU: 0 PID: 1 at lib/debugobjects.c:557 __debug_object_init+0x330/0x364 Modules linked in: CPU: 0 UID: 0 PID: 1 Comm: swapper/0 Not tainted 6.12.0-rc5 #4 Hardware name: linux,dummy-virt (DT) pstate: 600000c5 (nZCv daIF -PAN -UAO -TCO -DIT -SSBS BTYPE=--) pc : __debug_object_init+0x330/0x364 lr : __debug_object_init+0x330/0x364 sp : ffff800082ea7b40 x29: ffff800082ea7b40 x28: 98ff0000c0164518 x27: 98ff0000c0164534 x26: ffff800082d93ec8 x25: 0000000000000001 x24: 1cff0000c00172a0 x23: 0000000000000000 x22: ffff800082d93ed0 x21: ffff800081a24418 x20: 3eff800082ea7bb0 x19: efff800000000000 x18: 0000000000000000 x17: 00000000000000ff x16: 0000000000000047 x15: 206b63617473206e x14: 0000000000000018 x13: ffff800082ea7780 x12: 0ffff800082ea78e x11: 0ffff800082ea790 x10: 0ffff800082ea79d x9 : 34d77febe173e800 x8 : 34d77febe173e800 x7 : 0000000000000001 x6 : 0000000000000001 x5 : feff800082ea74b8 x4 : ffff800082870a90 x3 : ffff80008018d3c4 x2 : 0000000000000001 x1 : ffff800082858810 x0 : 0000000000000050 Call trace: __debug_object_init+0x330/0x364 debug_object_init_on_stack+0x30/0x3c schedule_hrtimeout_range_clock+0xac/0x26c schedule_hrtimeout+0x1c/0x30 wait_task_inactive+0x1d4/0x25c kthread_bind_mask+0x28/0x98 init_rescuer+0x1e8/0x280 workqueue_init+0x1a0/0x3cc kernel_init_freeable+0x118/0x200 kernel_init+0x28/0x1f0 ret_from_fork+0x10/0x20 ---[end trace 0000000000000000]--- ODEBUG: object 3eff800082ea7bb0 is NOT on stack ffff800082ea0000, but annotated. ----------[cut here]----------
CVE-2024-41932	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: sched: fix warning in sched_setaffinity Commit 8f9ea86fdf99b added some logic to sched_setaffinity that included a WARN when a per-task affinity assignment races with a cpuset update. Specifically, we can have a race where a cpuset update results in the task affinity no longer being a subset of the cpuset. That's fine; we have a fallback to instead use the cpuset mask. However, we have a WARN set up that will trigger if the cpuset mask has no overlap at all with the requested task affinity. This shouldn't be a warning condition; its trivial to create this condition. Reproduced the warning by the following setup: - $PID inside a cpuset cgroup - another thread repeatedly switching the cpuset cpus from 1-2 to just 1 - another thread repeatedly setting the $PID affinity (via taskset) to 2

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-71102	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: scs: fix a wrong parameter in __scs_magic __scs_magic() needs a 'void *' variable, but a 'struct task_struct *' is given. 'task_scs(tsk)' is the starting address of the task's shadow call stack, and '__scs_magic(task_scs(tsk))' is the end address of the task's shadow call stack. Here should be '__scs_magic(task_scs(tsk))'. The user-visible effect of this bug is that when CONFIG_DEBUG_STACK_USAGE is enabled, the shadow call stack usage checking function (scs_check_usage) would scan an incorrect memory range. This could lead 1. **Inaccurate stack usage reporting**: The function would calculate wrong usage statistics for the shadow call stack, potentially showing incorrect value in kmsg. 2. **Potential kernel crash**: If the value of __scs_magic(tsk)is greater than that of __scs_magic(task_scs(tsk)), the for loop may access unmapped memory, potentially causing a kernel panic. However, this scenario is unlikely because task_struct is allocated via the slab allocator (which typically returns lower addresses), while the shadow call stack returned by task_scs(tsk) is allocated via vmalloc(which typically returns higher addresses). However, since this is purely a debugging feature (CONFIG_DEBUG_STACK_USAGE), normal production systems should be not unaffected. The bug only impacts developers and testers who are actively debugging stack usage with this configuration enabled.
CVE-2024-57804	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: scsi: mpi3mr: Fix corrupt config pages PHY state is switched in sysfs The driver, through the SAS transport, exposes a sysfs interface to enable/disable PHYs in a controller/expander setup. When multiple PHYs are disabled and enabled in rapid succession, the persistent and current config pages related to SAS IO unit/SAS Expander pages could get corrupted. Use separate memory for each config request.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-71235	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: scsi: qla2xxx: Delay module unload while fabric scan in progress System crash seen during load/unload test in a loop. [105954.384919] RBP: ffff914589838dc0 R08: 0000000000000000 R09: 0000000000000086 [105954.384920] R10: 000000000000000f R11: ffffa31240904be5 R12: ffff914605f868e0 [105954.384921] R13: ffff914605f86910 R14: 0000000000008010 R15: 00000000ddb7c000 [105954.384923] FS: 0000000000000000(0000) GS:ffff9163fec40000(0000) knlGS:0000000000000000 [105954.384925] CS: 0010 DS: 0000 ES: 0000 CR0: 0000000080050033 [105954.384926] CR2: 000055d31ce1d6a0 CR3: 0000000119f5e001 CR4: 0000000000770ee0 [105954.384928] PKRU: 55555554 [105954.384929] Call Trace: [105954.384931] [105954.384934] qla24xx_sp_unmap+0x1f3/0x2a0 [qla2xxx] [105954.384962] ? qla_async_scan_sp_done+0x114/0x1f0 [qla2xxx] [105954.384980] ? qla24xx_els_ct_entry+0x4de/0x760 [qla2xxx] [105954.384999] ? __wake_up_common+0x80/0x190 [105954.385004] ? qla24xx_process_response_queue+0xc2/0xaa0 [qla2xxx] [105954.385023] ? qla24xx_msix_rsp_q+0x44/0xb0 [qla2xxx] [105954.385040] ? __handle_irq_event_percpu+0x3d/0x190 [105954.385044] ? handle_irq_event+0x58/0xb0 [105954.385046] ? handle_edge_irq+0x93/0x240 [105954.385050] ? __common_interrupt+0x41/0xa0 [105954.385055] ? common_interrupt+0x3e/0xa0 [105954.385060] ? asm_common_interrupt+0x22/0x40 The root cause of this was that there was a free (dma_free_attrs) in the interrupt context. There was a device discovery/fabric scan in progress. A module unload was issued which set the UNLOADING flag. As part of the discovery, after receiving an interrupt a work queue was scheduled (which involved a work to be queued). Since the UNLOADING flag is set, the work item was not allocated and the mapped memory had to be freed. The free occurred in interrupt context leading to system crash. Delay the driver unload until the fabric scan is complete to avoid the crash.
CVE-2025-37977	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: scsi: ufs: exynos: Disable iocc if dma-coherent property isn't set If dma-coherent property isn't set then descriptors are non-cacheable and the iocc shareability bits should be disabled. Without this UFS can end up in an incompatible configuration and suffer from random cache related stability issues.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-39724	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: serial: 8250: fix panic due to PSLVERR When the PSLVERR_RESP_EN parameter is set to 1, the device generates an error response if an attempt is made to read an empty RBR (Receive Buffer Register) while the FIFO is enabled. In serial8250_do_startup(), calling serial_port_out(port, UART_LCR, UART_LCR_WLEN8) triggers dw8250_check_lcr(), which invokes dw8250_force_idle() and serial8250_clear_and_reinit_fifos(). The latter function enables the FIFO via serial_out(p, UART_FCR, p->fcr). Execution proceeds to the serial_port_in(port, UART_RX). This satisfies the PSLVERR trigger condition. When another CPU (e.g., using printk()) is accessing the UART (UART is busy), the current CPU fails the check (value & ~UART_LCR_SPAR) == (lcr & ~UART_LCR_SPAR) in dw8250_check_lcr(), causing it to enter dw8250_force_idle(). Put serial_port_out(port, UART_LCR, UART_LCR_WLEN8) under the port->lock to fix this issue. Panic backtrace: [0.442336] Oops - unknown exception [#1] [0.442343] epc : dw8250_serial_in32+0x1e/0x4a [0.442351] ra : serial8250_do_startup+0x2c8/0x88e ... [0.442416] console_on_rootfs+0x26/0x70
CVE-2025-71072	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: shmem: fix recovery on rename failures maple_tree insertions can fail if we are seriously short on memory, simple_offset_rename() does not recover well if it runs into that. The same goes for simple_offset_rename_exchange(). Moreover, shmem_whiteout() expects that if it succeeds, the caller will progress to d_move(), i.e. that shmem_rename2() won't fail past the successful call of shmem_whiteout(). Not hard to fix, fortunately - mtree_store() can't fail if the index we are trying to store into is already present in the tree as a singleton. For simple_offset_rename_exchange() that's enough - we just need to be careful about the order of operations. For simple_offset_rename() solution is to preinsert the target into the tree for new_dir; the rest can be done without any potentially failing operations. That preinsertion has to be done in shmem_rename2() rather than in simple_offset_rename() itself - otherwise we'd need to deal with the possibility of failure after successful shmem_whiteout().
CVE-2026-23090	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: slimbus: core: fix device reference leak on report present Slimbus devices can be allocated dynamically upon reception of report-present messages. Make sure to drop the reference taken when looking up already registered devices. Note that this requires taking an extra reference in case the device has not yet been registered and has to be allocated.
CVE-2025-71204	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: smb/server: fix refcount leak in parse_durable_handle_context() When the command is a replay operation and -ENOEXEC is returned, the refcount of ksmbd_file must be released.
CVE-2025-71223	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: smb/server: fix refcount leak in smb2_open() When ksmbd_vfs_getattr() fails, the reference count of ksmbd_file must be released.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38379	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: smb: client: fix warning when reconnecting channel When reconnecting a channel in smb2_reconnect_server(), a dummy tcon is passed down to smb2_reconnect() with ->query_interface uninitialized, so we can't call queue_delayed_work() on it. Fix the following warning by ensuring that we're queueing the delayed worker from correct tcon. WARNING: CPU: 4 PID: 1126 at kernel/workqueue.c:2498 __queue_delayed_work+0x1d2/0x200 Modules linked in: cifs cifs_arc4 nls_ucs2_utils cifs_md4 [last unloaded: cifs] CPU: 4 UID: 0 PID: 1126 Comm: kworker/4:0 Not tainted 6.16.0-rc3 #5 PREEMPT(voluntary) Hardware name: QEMU Standard PC (Q35 + ICH9, 2009), BIOS 1.16.3-4.fc42 04/01/2014 Workqueue: cifsiod smb2_reconnect_server [cifs] RIP: 0010:__queue_delayed_work+0x1d2/0x200 Code: 41 5e 41 5f e9 7f ee ff ff 90 0f 0b 90 e9 5d ff ff ff bf 02 00 00 00 e8 6c f3 07 00 89 c3 eb bd 90 0f 0b 90 e9 57 f> 0b 90 e9 65 fe ff ff 90 0f 0b 90 e9 72 fe ff ff 90 0f 0b 90 e9 RSP: 0018:ffffc900014afad8 EFLAGS: 00010003 RAX: 0000000000000000 RBX: ffff888124d99988 RCX: ffffffff81399cc1 RDX: dffffc0000000000 RSI: ffff888114326e00 RDI: ffff888124d999f0 RBP: 000000000000ea60 R08: 0000000000000001 R09: ffffed10249b3331 R10: ffff888124d9998f R11: 0000000000000004 R12: 0000000000000040 R13: ffff888114326e00 R14: ffff888124d999d8 R15: ffff888114939020 FS: 0000000000000000(0000) GS:ffff88829f7fe000(0000) knlGS:0000000000000000 CS: 0010 DS: 0000 ES: 0000 CR0: 0000000080050033 CR2: 00007ffe7a2b4038 CR3: 0000000120a6f000 CR4: 0000000000750ef0 PKRU: 55555554 Call Trace: queue_delayed_work_on+0xb4/0xc0 smb2_reconnect+0xb22/0xf50 [cifs] smb2_reconnect_server+0x413/0xd40 [cifs] ? __pfx_smb2_reconnect_server+0x10/0x10 [cifs] ? local_clock_noinstr+0xd/0xd0 ? local_clock+0x15/0x30 ? lock_release+0x29b/0x390 process_one_work+0x4c5/0xa10 ? __pfx_process_one_work+0x10/0x10 ? __list_add_valid_or_report+0x37/0x120 worker_thread+0x2f1/0x5a0 ? __kthread_parkme+0xde/0x100 ? __pfx_worker_thread+0x10/0x10 kthread+0x1fe/0x380 ? kthread+0x10f/0x380 ? __pfx_kthread+0x10/0x10 ? local_clock_noinstr+0xd/0xd0 ? ret_from_fork+0x1b/0x1f0 ? local_clock+0x15/0x30 ? lock_release+0x29b/0x390 ? rcu_is_watching+0x20/0x50 ? __pfx_kthread+0x10/0x10 ret_from_fork+0x15b/0x1f0 ? __pfx_kthread+0x10/0x10 ret_from_fork_asm+0x1a/0x30 irq event stamp: 1116206 hardirqs last enabled at (1116205): [] __up_console_sem+0x52/0x60 hardirqs last disabled at (1116206): [] queue_delayed_work_on+0x6e/0xc0 softirqs last enabled at (1116138): [] __smb_send_rqst+0x42d/0x950 [cifs] softirqs last disabled at (1116136): [] release_sock+0x21/0xf0

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-39932	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: smb: client: let smbd_destroy() call disable_work_sync(&info->post_send_credits_work) In smbd_destroy() we may destroy the memory so we better wait until post_send_credits_work is no longer pending and will never be started again. I actually just hit the case using rxe: WARNING: CPU: 0 PID: 138 at drivers/infiniband/sw/rxe/rxe_verbs.c:1032 rxe_post_recv+0x1ee/0x480 [rdma_rxe] ... [5305.686979] [T138] smbd_post_recv+0x445/0xc10 [cifs] [5305.687135] [T138] ? srso_alias_return_thunk+0x5/0xfbef5 [5305.687149] [T138] ? __kasan_check_write+0x14/0x30 [5305.687185] [T138] ? __pfx_smbd_post_recv+0x10/0x10 [cifs] [5305.687329] [T138] ? __pfx__raw_spin_lock_irqsave+0x10/0x10 [5305.687356] [T138] ? srso_alias_return_thunk+0x5/0xfbef5 [5305.687368] [T138] ? srso_alias_return_thunk+0x5/0xfbef5 [5305.687378] [T138] ? _raw_spin_unlock_irqrestore+0x11/0x60 [5305.687389] [T138] ? srso_alias_return_thunk+0x5/0xfbef5 [5305.687399] [T138] ? get_receive_buffer+0x168/0x210 [cifs] [5305.687555] [T138] smbd_post_send_credits+0x382/0x4b0 [cifs] [5305.687701] [T138] ? __pfx_smbd_post_send_credits+0x10/0x10 [cifs] [5305.687855] [T138] ? __pfx___schedule+0x10/0x10 [5305.687865] [T138] ? __pfx__raw_spin_lock_irq+0x10/0x10 [5305.687875] [T138] ? queue_delayed_work_on+0x8e/0xa0 [5305.687889] [T138] process_one_work+0x629/0xf80 [5305.687908] [T138] ? srso_alias_return_thunk+0x5/0xfbef5 [5305.687917] [T138] ? __kasan_check_write+0x14/0x30 [5305.687933] [T138] worker_thread+0x87f/0x1570 ... It means rxe_post_recv was called after rdma_destroy_qp(). This happened because put_receive_buffer() was triggered by ib_drain_qp() and called: queue_work(info->workqueue, &info->post_send_credits_work);
CVE-2026-23230	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: smb: client: split cached_fid bitfields to avoid shared-byte RMW races is_open, has_lease and on_list are stored in the same bitfield byte in struct cached_fid but are updated in different code paths that may run concurrently. Bitfield assignments generate byte read–modify–write operations (e.g. `orb $mask, addr` on x86_64), so updating one flag can restore stale values of the others. A possible interleaving is: CPU1: load old byte (has_lease=1, on_list=1) CPU2: clear both flags (store 0) CPU1: RMW store (old | IS_OPEN) -> reintroduces cleared bits To avoid this class of races, convert these flags to separate bool fields.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-71132	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: smc91x: fix broken irq-context in PREEMPT_RT When smc91x.c is built with PREEMPT_RT, the following splat occurs in FVP_RevC: [13.055000] smc91x LNRO0003:00 eth0: link up, 10Mbps, half-duplex, lpa 0x0000 [13.062137] BUG: workqueue leaked atomic, lock or RCU: kworker/2:1[106] [13.062137] preempt=0x00000000 lock=0->0 RCU=0->1 workfn=mld_ifc_work [13.062266] C ** replaying previous printk message ** [13.062266] CPU: 2 UID: 0 PID: 106 Comm: kworker/2:1 Not tainted 6.18.0-dirty #179 PREEMPT_{RT,(full)} [13.062353] Hardware name: , BIOS [13.062382] Workqueue: mld mld_ifc_work [13.062469] Call trace: [13.062494] show_stack+0x24/0x40 (C) [13.062602] __dump_stack+0x28/0x48 [13.062710] dump_stack_lvl+0x7c/0xb0 [13.062818] dump_stack+0x18/0x34 [13.062926] process_scheduled_works+0x294/0x450 [13.063043] worker_thread+0x260/0x3d8 [13.063124] kthread+0x1c4/0x228 [13.063235] ret_from_fork+0x10/0x20 This happens because smc_special_trylock() disables IRQs even on PREEMPT_RT, but smc_special_unlock() does not restore IRQs on PREEMPT_RT. The reason is that smc_special_unlock() calls spin_unlock_irqrestore(), and rcu_read_unlock_bh() in __dev_queue_xmit() cannot invoke rcu_read_unlock() through __local_bh_enable_ip() when current->softirq_disable_cnt becomes zero. To address this issue, replace smc_special_trylock() with spin_trylock_irqsave().
CVE-2025-39787	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: soc: qcom: mdt_loader: Ensure we don't read past the ELF header When the MDT loader is used in remoteproc, the ELF header is sanitized beforehand, but that's not necessary the case for other clients. Validate the size of the firmware buffer to ensure that we don't read past the end as we iterate over the header. e_phentsize and e_shentsize are validated as well, to ensure that the assumptions about step size in the traversal are valid.
CVE-2025-37842	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: spi: fsl-qspi: use devm function instead of driver remove Driver use devm APIs to manage clk/irq/resources and register the spi controller, but the legacy remove function will be called first during device detach and trigger kernel panic. Drop the remove function and use devm_add_action_or_reset() for driver cleanup to ensure the release sequence. Trigger kernel panic on i.MX8MQ by echo 30bb0000.spi >/sys/bus/platform/drivers/fsl-quadspi/unbind
CVE-2025-38571	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: sunrpc: fix client side handling of tls alerts A security exploit was discovered in NFS over TLS in tls_alert_recv due to its assumption that there is valid data in the msghdr's iterator's kvec. Instead, this patch proposes the rework how control messages are setup and used by sock_recvmsg(). If no control message structure is setup, kTLS layer will read and process TLS data record types. As soon as it encounters a TLS control message, it would return an error. At that point, NFS can setup a kvec backed control buffer and read in the control message such as a TLS alert. Scott found that a msg iterator can advance the kvec pointer as a part of the copy process thus we need to revert the iterator before calling into the tls_alert_recv.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-68340	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: team: Move team device type change at the end of team_port_add Attempting to add a port device that is already up will expectedly fail, but not before modifying the team device header_ops. In the case of the syzbot reproducer the gre0 device is already in state UP when it attempts to add it as a port device of team0, this fails but before that header_ops->create of team0 is changed from eth_header to ipgre_header in the call to team_dev_type_check_change. Later when we end up in ipgre_header() struct ip_tunnel* points to nonsense as the private data of the device still holds a struct team. Example sequence of iproute2 commands to reproduce the hang/BUG(): ip link add dev team0 type team ip link add dev gre0 type gre ip link set dev gre0 up ip link set dev gre0 master team0 ip link set dev team0 up ping -I team0 1.1.1.1 Move team_dev_type_check_change down where all other checks have passed as it changes the dev type with no way to restore it in case one of the checks that follow it fail. Also make sure to preserve the origial mtu assignment: - If port_dev is not the same type as dev, dev takes mtu from port_dev - If port_dev is the same type as dev, port_dev takes mtu from dev This is done by adding a conditional before the call to dev_set_mtu to prevent it from assigning port_dev->mtu = dev->mtu and instead letting team_dev_type_check_change assign dev->mtu = port_dev->mtu. The conditional is needed because the patch moves the call to team_dev_type_check_change past dev_set_mtu. Testing: - team device driver in-tree selftests - Add/remove various devices as slaves of team device - syzbot
CVE-2024-50028	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: thermal: core: Reference count the zone in thermal_zone_get_by_id() There are places in the thermal netlink code where nothing prevents the thermal zone object from going away while being accessed after it has been returned by thermal_zone_get_by_id(). To address this, make thermal_zone_get_by_id() get a reference on the thermal zone device object to be returned with the help of get_device(), under thermal_list_lock, and adjust all of its callers to this change with the help of the cleanup.h infrastructure.
CVE-2025-39946	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: tls: make sure to abort the stream if headers are bogus Normally we wait for the socket to buffer up the whole record before we service it. If the socket has a tiny buffer, however, we read out the data sooner, to prevent connection stalls. Make sure that we abort the connection when we find out late that the record is actually invalid. Retrying the parsing is fine in itself but since we copy some more data each time before we parse we can overflow the allocated skb space. Constructing a scenario in which we're under pressure without enough data in the socket to parse the length upfront is quite hard. syzbot figured out a way to do this by serving us the header in small OOB sends, and then filling in the recvbuf with a large normal send. Make sure that tls_rx_msg_size() aborts strp, if we reach an invalid record there's really no way to recover.
CVE-2025-71077	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: tpm: Cap the number of PCR banks tpm2_get_pcr_allocation() does not cap any upper limit for the number of banks. Cap the limit to eight banks so that out of bounds values coming from external I/O cause on only limited harm.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-39829	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: trace/fgraph: Fix the warning caused by missing unregister notifier This warning was triggered during testing on v6.16: notifier callback ftrace_suspend_notifier_call already registered WARNING: CPU: 2 PID: 86 at kernel/notifier.c:23 notifier_chain_register+0x44/0xb0 ... Call Trace: blocking_notifier_chain_register+0x34/0x60 register_ftrace_graph+0x330/0x410 ftrace_profile_write+0x1e9/0x340 vfs_write+0xf8/0x420 ? filp_flush+0x8a/0xa0 ? filp_close+0x1f/0x30 ? do_dup2+0xaf/0x160 ksys_write+0x65/0xe0 do_syscall_64+0xa4/0x260 entry_SYSCALL_64_after_hwframe+0x77/0x7f When writing to the function_profile_enabled interface, the notifier was not unregistered after start_graph_tracing failed, causing a warning the next time function_profile_enabled was written. Fixed by adding unregister_pm_notifier in the exception path.
CVE-2025-38539	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: tracing: Add down_write(trace_event_sem) when adding trace event When a module is loaded, it adds trace events defined by the module. It may also need to modify the modules trace printk formats to replace enum names with their values. If two modules are loaded at the same time, the adding of the event to the ftrace_events list can corrupt the walking of the list in the code that is modifying the printk format strings and crash the kernel. The addition of the event should take the trace_event_sem for write while it adds the new event. Also add a lockdep_assert_held() on that semaphore in __trace_add_event_dirs() as it iterates the list.
CVE-2026-23096	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: uacce: fix cdev handling in the cleanup path When cdev_device_add fails, it internally releases the cdev memory, and if cdev_device_del is then executed, it will cause a hang error. To fix it, we check the return value of cdev_device_add() and clear uacce->cdev to avoid calling cdev_device_del in the uacce_remove.
CVE-2026-23094	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: uacce: fix isolate sysfs check condition uacce supports the device isolation feature. If the driver implements the isolate_err_threshold_read and isolate_err_threshold_write callback functions, uacce will create sysfs files now. Users can read and configure the isolation policy through sysfs. Currently, sysfs files are created as long as either isolate_err_threshold_read or isolate_err_threshold_write callback functions are present. However, accessing a non-existent callback function may cause the system to crash. Therefore, intercept the creation of sysfs if neither read nor write exists; create sysfs if either is supported, but intercept unsupported operations at the call site.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38376	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: usb: chipidea: udc: disconnect/reconnect from host when do suspend/resume Shawn and John reported a hang issue during system suspend as below: - USB gadget is enabled as Ethernet - There is data transfer over USB Ethernet (scp a big file between host and device) - Device is going in/out suspend (echo mem > /sys/power/state) The root cause is the USB device controller is suspended but the USB bus is still active which caused the USB host continues to transfer data with device and the device continues to queue USB requests (in this case, a delayed TCP ACK packet trigger the issue) after controller is suspended, however the USB controller clock is already gated off. Then if udc driver access registers after that point, the system will hang. The correct way to avoid such issue is to disconnect device from host when the USB bus is not at suspend state. Then the host will receive disconnect event and stop data transfer in time. To continue make USB gadget device work after system resume, this will reconnect device automatically. To make usb wakeup work if USB bus is already at suspend state, this will keep connection for it only when USB device controller has enabled wakeup capability.
CVE-2025-38474	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: usb: net: sierra: check for no status endpoint The driver checks for having three endpoints and having bulk in and out endpoints, but not that the third endpoint is interrupt input. Rectify the omission.
CVE-2025-71108	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: usb: typec: ucsi: Handle incorrect num_connectors capability The UCSI spec states that the num_connectors field is 7 bits, and the 8th bit is reserved and should be set to zero. Some buggy FW has been known to set this bit, and it can lead to a system not booting. Flag that the FW is not behaving correctly, and auto-fix the value so that the system boots correctly. Found on Lenovo P1 G8 during Linux enablement program. The FW will be fixed, but seemed worth addressing in case it hit platforms that aren't officially Linux supported.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-40164	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: usbnet: Fix using smp_processor_id() in preemptible code warnings Syzbot reported the following warning: BUG: using smp_processor_id() in preemptible [00000000] code: dhcpcd/2879 caller is usbnet_skb_return+0x74/0x490 drivers/net/usb/usbnet.c:331 CPU: 1 UID: 0 PID: 2879 Comm: dhcpcd Not tainted 6.15.0-rc4-syzkaller-00098-g615dca38c2ea #0 PREEMPT(voluntary) Call Trace: __dump_stack lib/dump_stack.c:94 [inline] dump_stack_lvl+0x16c/0x1f0 lib/dump_stack.c:120 check_preemption_disabled+0xd0/0xe0 lib/smp_processor_id.c:49 usbnet_skb_return+0x74/0x490 drivers/net/usb/usbnet.c:331 usbnet_resume_rx+0x4b/0x170 drivers/net/usb/usbnet.c:708 usbnet_change_mtu+0x1be/0x220 drivers/net/usb/usbnet.c:417 __dev_set_mtu net/core/dev.c:9443 [inline] netif_set_mtu_ext+0x369/0x5c0 net/core/dev.c:9496 netif_set_mtu+0xb0/0x160 net/core/dev.c:9520 dev_set_mtu+0xae/0x170 net/core/dev_api.c:247 dev_ifsioc+0xa31/0x18d0 net/core/dev_ioctl.c:572 dev_ioctl+0x223/0x10e0 net/core/dev_ioctl.c:821 sock_do_ioctl+0x19d/0x280 net/socket.c:1204 sock_ioctl+0x42f/0x6a0 net/socket.c:1311 vfs_ioctl fs/ioctl.c:51 [inline] __do_sys_ioctl fs/ioctl.c:906 [inline] __se_sys_ioctl fs/ioctl.c:892 [inline] __x64_sys_ioctl+0x190/0x200 fs/ioctl.c:892 do_syscall_x64 arch/x86/entry/syscall_64.c:63 [inline] do_syscall_64+0xcd/0x260 arch/x86/entry/syscall_64.c:94 entry_SYSCALL_64_after_hwframe+0x77/0x7f For historical and portability reasons, the netif_rx() is usually run in the softirq or interrupt context, this commit therefore add local_bh_disable/enable() protection in the usbnet_resume_rx().
CVE-2025-38625	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: vfio/pds: Fix missing detach_ioas op When CONFIG_IOMMUFD is enabled and a device is bound to the pds_vfio_pci driver, the following WARN_ON() trace is seen and probe fails: WARNING: CPU: 0 PID: 5040 at drivers/vfio/vfio_main.c:317 __vfio_register_dev+0x130/0x140 [vfio] <...> pds_vfio_pci 0000:08:00.1: probe with driver pds_vfio_pci failed with error -22 This is because the driver's vfio_device_ops.detach_ioas isn't set. Fix this by using the generic vfio_iommufd_physical_detach_ioas function.
CVE-2024-53219	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: virtiofs: use pages instead of pointer for kernel direct IO When trying to insert a 10MB kernel module kept in a virtio-fs with cache disabled, the following warning was reported: ------------[cut here]------------ WARNING: CPU: 1 PID: 404 at mm/page_alloc.c:4551 ..... Modules linked in: CPU: 1 PID: 404 Comm: insmod Not tainted 6.9.0-rc5+ #123 Hardware name: QEMU Standard PC (i440FX + PIIX, 1996) ..... RIP: 0010:__alloc_pages+0x2bf/0x380 ..... Call Trace: ? __warn+0x8e/0x150 ? __alloc_pages+0x2bf/0x380 __kmalloc_large_node+0x86/0x160 __kmalloc+0x33c/0x480 virtio_fs_enqueue_req+0x240/0x6d0 virtio_fs_wake_pending_and_unlock+0x7f/0x190 queue_request_and_unlock+0x55/0x60 fuse_simple_request+0x152/0x2b0 fuse_direct_io+0x5d2/0x8c0 fuse_file_read_iter+0x121/0x160 __kernel_read+0x151/0x2d0 kernel_read+0x45/0x50 kernel_read_file+0x1a9/0x2a0 init_module_from_file+0x6a/0xe0 idempotent_init_module+0x175/0x230

CVE	Severity	CVSS	CVSS Vector	Component	Description
					__x64_sys_finit_module+0x5d/0xb0 x64_sys_call+0x1c3/0x9e0 do_syscall_64+0x3d/0xc0 entry_SYSCALL_64_after_hwframe+0x4b/0x53 ..... ---[end trace 0000000000000000]--- The warning is triggered as follows: 1) syscall finit_module() handles the module insertion and it invokes kernel_read_file() to read the content of the module first. 2) kernel_read_file() allocates a 10MB buffer by using vmalloc() and passes it to kernel_read(). kernel_read() constructs a kvec iter by using iov_iter_kvec() and passes it to fuse_file_read_iter(). 3) virtio-fs disables the cache, so fuse_file_read_iter() invokes fuse_direct_io(). As for now, the maximal read size for kvec iter is only limited by fc->max_read. For virtio-fs, max_read is UINT_MAX, so fuse_direct_io() doesn't split the 10MB buffer. It saves the address and the size of the 10MB-sized buffer in out_args[0] of a fuse request and passes the fuse request to virtio_fs_wake_pending_and_unlock(). 4) virtio_fs_wake_pending_and_unlock() uses virtio_fs_enqueue_req() to queue the request. Because virtiofs need DMA-able address, so virtio_fs_enqueue_req() uses kmalloc() to allocate a bounce buffer for all fuse args, copies these args into the bounce buffer and passed the physical address of the bounce buffer to virtiofsd. The total length of these fuse args for the passed fuse request is about 10MB, so copy_args_to_argbuf() invokes kmalloc() with a 10MB size parameter and it triggers the warning in __alloc_pages(): if (WARN_ON_ONCE_GFP(order > MAX_PAGE_ORDER, gfp)) return NULL; 5) virtio_fs_enqueue_req() will retry the memory allocation in a kworker, but it won't help, because kmalloc() will always return NULL due to the abnormal size and finit_module() will hang forever. A feasible solution is to limit the value of max_read for virtio-fs, so the length passed to kmalloc() will be limited. However it will affect the maximal read size for normal read. And for virtio-fs write initiated from kernel, it has the similar problem but now there is no way to limit fc->max_write in kernel. So instead of limiting both the values of max_read and max_write in kernel, introducing use_pages_for_kvec_io in fuse_conn and setting it as true in virtiofs. When use_pages_for_kvec_io is enabled, fuse will use pages instead of pointer to pass the KVEC_IO data. After switching to pages for KVEC_IO data, these pages will be used for DMA through virtio-fs. If these pages are backed by vmalloc(), {flush|invalidate}_kernel_vmap_range() are necessary to flush or invalidate the cache before the DMA operation. So add two new fields in fuse_args_pages to record the base address of vmalloc area and the condition indicating whether invalidation is needed. Perform the flush in fuse_get_user_pages() for write operations and the invalidation in fuse_release_user_pages() for read operations. It may seem necessary to introduce another fie ---truncated---
CVE-2025-22106	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: vmxnet3: unregister xdp rxq info in the reset path vmxnet3 does not unregister xdp rxq info in the vmxnet3_reset_work() code path as vmxnet3_rq_destroy() is not invoked in this code path. So, we get below message with a backtrace. Missing unregister, handled but fix driver WARNING: CPU:48 PID: 500 at net/core/xdp.c:182 __xdp_rxq_info_reg+0x93/0xf0 This patch fixes the problem by moving the unregister code of XDP from vmxnet3_rq_destroy() to vmxnet3_rq_cleanup().

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2026-23086	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: vsock/virtio: cap TX credit to local buffer size The virtio transports derives its TX credit directly from peer_buf_alloc, which is set from the remote endpoint's SO_VM_SOCKETS_BUFFER_SIZE value. On the host side this means that the amount of data we are willing to queue for a connection is scaled by a guest-chosen buffer size, rather than the host's own vsock configuration. A malicious guest can advertise a large buffer and read slowly, causing the host to allocate a correspondingly large amount of sk_buff memory. The same thing would happen in the guest with a malicious host, since virtio transports share the same code base. Introduce a small helper, virtio_transport_tx_buf_size(), that returns min(peer_buf_alloc, buf_alloc), and use it wherever we consume peer_buf_alloc. This ensures the effective TX window is bounded by both the peer's advertised buffer and our own buf_alloc (already clamped to buffer_max_size via SO_VM_SOCKETS_BUFFER_MAX_SIZE), so a remote peer cannot force the other to queue more data than allowed by its own vsock settings. On an unpatched Ubuntu 22.04 host (~64 GiB RAM), running a PoC with 32 guest vsock connections advertising 2 GiB each and reading slowly drove Slab/SUnreclaim from ~0.5 GiB to ~57 GiB; the system only recovered after killing the QEMU process. That said, if QEMU memory is limited with cgroups, the maximum memory used will be limited. With this patch applied: Before: MemFree: ~61.6 GiB Slab: ~142 MiB SUnreclaim: ~117 MiB After 32 high-credit connections: MemFree: ~61.5 GiB Slab: ~178 MiB SUnreclaim: ~152 MiB Only ~35 MiB increase in Slab/SUnreclaim, no host OOM, and the guest remains responsive. Compatibility with non-virtio transports: - VMCI uses the AF_VSOCK buffer knobs to size its queue pairs per socket based on the local vsk->buffer_* values; the remote side cannot enlarge those queues beyond what the local endpoint configured. - Hyper-V's vsock transport uses fixed-size VMBus ring buffers and an MTU bound; there is no peer-controlled credit field comparable to peer_buf_alloc, and the remote endpoint cannot drive in-flight kernel memory above those ring sizes. - The loopback path reuses virtio_transport_common.c, so it naturally follows the same semantics as the virtio transport. This change is limited to virtio_transport_common.c and thus affects virtio-vsock, vhost-vsock, and loopback, bringing them in line with the "remote window intersected with local policy" behaviour that VMCI and Hyper-V already effectively have. [Stefano: small adjustments after changing the previous patch] [Stefano: tweak the commit message]
CVE-2026-23133	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: wifi: ath10k: fix dma_free_coherent() pointer dma_alloc_coherent() allocates a DMA mapped buffer and stores the addresses in XXX_unaligned fields. Those should be reused when freeing the buffer rather than the aligned addresses.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-39746	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: wifi: ath10k: shutdown driver when hardware is unreliable In rare cases, ath10k may lose connection with the PCIe bus due to some unknown reasons, which could further lead to system crashes during resuming due to watchdog timeout: ath10k_pci 0000:01:00.0: wmi command 20486 timeout, restarting hardware ath10k_pci 0000:01:00.0: already restarting ath10k_pci 0000:01:00.0: failed to stop WMI vdev 0: -11 ath10k_pci 0000:01:00.0: failed to stop vdev 0: -11 ieee80211 phy0: PM: **** DPM device timeout **** Call Trace: panic+0x125/0x315 dpm_watchdog_set+0x54/0x54 dpm_watchdog_handler+0x57/0x57 call_timer_fn+0x31/0x13c At this point, all WMI commands will timeout and attempt to restart device. So set a threshold for consecutive restart failures. If the threshold is exceeded, consider the hardware is unreliable and all ath10k operations should be skipped to avoid system crash. fail_cont_count and pending_recovery are atomic variables, and do not involve complex conditional logic. Therefore, even if recovery check and reconfig complete are executed concurrently, the recovery mechanism will not be broken. Tested-on: QCA6174 hw3.2 PCI WLAN.RM4.4.1-00288-QCARMSWPZ-1
CVE-2025-23129	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: wifi: ath11k: Clear affinity hint before calling ath11k_pcic_free_irq() in error path If a shared IRQ is used by the driver due to platform limitation, then the IRQ affinity hint is set right after the allocation of IRQ vectors in ath11k_pci_alloc_msi(). This does no harm unless one of the functions requesting the IRQ fails and attempt to free the IRQ. This results in the below warning: WARNING: CPU: 7 PID: 349 at kernel/irq/manage.c:1929 free_irq+0x278/0x29c Call trace: free_irq+0x278/0x29c ath11k_pcic_free_irq+0x70/0x10c [ath11k] ath11k_pci_probe+0x800/0x820 [ath11k_pci] local_pci_probe+0x40/0xbc The warning is due to not clearing the affinity hint before freeing the IRQs. So to fix this issue, clear the IRQ affinity hint before calling ath11k_pcic_free_irq() in the error path. The affinity will be cleared once again further down the error path due to code organization, but that does no harm. Tested-on: QCA6390 hw2.0 PCI WLAN.HST.1.0.1-05266-QCAHSTSWPLZ_V2_TO_X86-1

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2024-58096	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: wifi: ath11k: add srng->lock for ath11k_hal_srng_* in monitor mode ath11k_hal_srng_* should be used with srng->lock to protect srng data. For ath11k_dp_rx_mon_dest_process() and ath11k_dp_full_mon_process_rx(), they use ath11k_hal_srng_* for many times but never call srng->lock. So when running (full) monitor mode, warning will occur: RIP: 0010:ath11k_hal_srng_dst_peek+0x18/0x30 [ath11k] Call Trace: ? ath11k_hal_srng_dst_peek+0x18/0x30 [ath11k] ath11k_dp_rx_process_mon_status+0xc45/0x1190 [ath11k] ? idr_alloc_u32+0x97/0xd0 ath11k_dp_rx_process_mon_rings+0x32a/0x550 [ath11k] ath11k_dp_service_srng+0x289/0x5a0 [ath11k] ath11k_pcic_ext_grp_napi_poll+0x30/0xd0 [ath11k] __napi_poll+0x30/0x1f0 net_rx_action+0x198/0x320 __do_softirq+0xdd/0x319 So add srng->lock for them to avoid such warnings. Inorder to fetch the srng->lock, should change srng's definition from 'void' to 'struct hal_srng'. And initialize them elsewhere to prevent one line of code from being too long. This is consistent with other ring process functions, such as ath11k_dp_process_rx(). Tested-on: WCN6855 hw2.0 PCI WLAN.HSP.1.1-03125-QCAHSPSWPL_V1_V2_SILICONZ_LITE-3.6510.30 Tested-on: QCN9074 hw1.0 PCI WLAN.HK.2.7.0.1-01744-QCAHKSWPL_SILICONZ-1
CVE-2025-39732	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: wifi: ath11k: fix sleeping-in-atomic in ath11k_mac_op_set_bitrate_mask() ath11k_mac_disable_peer_fixed_rate() is passed as the iterator to ieee80211_iterate_stations_atomic(). Note in this case the iterator is required to be atomic, however ath11k_mac_disable_peer_fixed_rate() does not follow it as it might sleep. Consequently below warning is seen: BUG: sleeping function called from invalid context at wmi.c:304 Call Trace: dump_stack_lvl __might_resched.cold ath11k_wmi_cmd_send ath11k_wmi_set_peer_param ath11k_mac_disable_peer_fixed_rate ieee80211_iterate_stations_atomic ath11k_mac_op_set_bitrate_mask.cold Change to ieee80211_iterate_stations_mtx() to fix this issue. Tested-on: WCN6855 hw2.0 PCI WLAN.HSP.1.1-03125-QCAHSPSWPL_V1_V2_SILICONZ_LITE-3.6510.30
CVE-2025-38414	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: wifi: ath12k: fix GCC_GCC_PCIE_HOT_RST definition for WCN7850 GCC_GCC_PCIE_HOT_RST is wrongly defined for WCN7850, causing kernel crash on some specific platforms. Since this register is divergent for WCN7850 and QCN9274, move it to register table to allow different definitions. Then correct the register address for WCN7850 to fix this issue. Note IPQ5332 is not affected as it is not PCIe based device. Tested-on: WCN7850 hw2.0 PCI WLAN.HMT.1.0.c5-00481-QCAHMTSWPL_V1.0_V2.0_SILICONZ-3
CVE-2026-23135	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: wifi: ath12k: fix dma_free_coherent() pointer dma_alloc_coherent() allocates a DMA mapped buffer and stores the addresses in XXX_unaligned fields. Those should be reused when freeing the buffer rather than the aligned addresses.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38406	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: wifi: ath6kl: remove WARN on bad firmware input If the firmware gives bad input, that's nothing to do with the driver's stack at this point etc., so the WARN_ON() doesn't add any value. Additionally, this is one of the top syzbot reports now. Just print a message, and as an added bonus, print the sizes too.
CVE-2025-38096	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: wifi: iwlwifi: don't warn when if there is a FW error iwl_trans_reclaim is warning if it is called when the FW is not alive. But if it is called when there is a pending restart, i.e. after a FW error, there is no need to warn, instead - return silently.
CVE-2025-71127	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: wifi: mac80211: Discard Beacon frames to non-broadcast address Beacon frames are required to be sent to the broadcast address, see IEEE Std 802.11-2020, 11.1.3.1 ("The Address 1 field of the Beacon .. frame shall be set to the broadcast address"). A unicast Beacon frame might be used as a targeted attack to get one of the associated STAs to do something (e.g., using CSA to move it to another channel). As such, it is better have strict filtering for this on the received side and discard all Beacon frames that are sent to an unexpected address. This is even more important for cases where beacon protection is used. The current implementation in mac80211 is correctly discarding unicast Beacon frames if the Protected Frame bit in the Frame Control field is set to 0. However, if that bit is set to 1, the logic used for checking for configured BIGTK(s) does not actually work. If the driver does not have logic for dropping unicast Beacon frames with Protected Frame bit 1, these frames would be accepted in mac80211 processing as valid Beacon frames even though they are not protected. This would allow beacon protection to be bypassed. While the logic for checking beacon protection could be extended to cover this corner case, a more generic check for discard all Beacon frames based on A1=unicast address covers this without needing additional changes. Address all these issues by dropping received Beacon frames if they are sent to a non-broadcast address.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-71229	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: wifi: rtw88: Fix alignment fault in rtw_core_enable_beacon() rtw_core_enable_beacon() reads 4 bytes from an address that is not a multiple of 4. This results in a crash on some systems. Do 1 byte reads/writes instead. Unable to handle kernel paging request at virtual address ffff8000827e0522 Mem abort info: ESR = 0x0000000096000021 EC = 0x25: DABT (current EL), IL = 32 bits SET = 0, FnV = 0 EA = 0, S1PTW = 0 FSC = 0x21: alignment fault Data abort info: ISV = 0, ISS = 0x00000021, ISS2 = 0x00000000 CM = 0, WnR = 0, TnD = 0, TagAccess = 0 GCS = 0, Overlay = 0, DirtyBit = 0, Xs = 0 swapper pgtable: 4k pages, 48-bit VAs, pgdp=0000000005492000 [ffff8000827e0522] pgd=0000000000000000, p4d=10000001021d9403, pud=10000001021da403, pmd=100000011061c403, pte=00780000f3200f13 Internal error: Oops: 0000000096000021 [#1] SMP Modules linked in: [...] rtw88_8822ce rtw88_8822c rtw88_pci rtw88_core [...] CPU: 0 UID: 0 PID: 73 Comm: kworker/u32:2 Tainted: G W 6.17.9 #1-NixOS VOLUNTARY Tainted: [W]=WARN Hardware name: FriendlyElec NanoPC-T6 LTS (DT) Workqueue: phy0 rtw_c2h_work [rtw88_core] pstate: 60400009 (nZCv daif +PAN -UAO -TCO -DIT -SSBS BTYPE=--) pc : rtw_pci_read32+0x18/0x40 [rtw88_pci] lr : rtw_core_enable_beacon+0xe0/0x148 [rtw88_core] sp : ffff800080cc3ca0 x29: ffff800080cc3ca0 x28: ffff0001031fc240 x27: ffff000102100828 x26: ffffd2cb7c9b4088 x25: ffff0001031fc2c0 x24: ffff000112fdef00 x23: ffff000112fdef18 x22: ffff000111c29970 x21: 0000000000000001 x20: 0000000000000001 x19: ffff000111c22040 x18: 0000000000000000 x17: 0000000000000000 x16: 0000000000000000 x15: 0000000000000000 x14: 0000000000000000 x13: 0000000000000000 x12: 0000000000000000 x11: 0000000000000000 x10: 0000000000000000 x9 : ffffd2cb6507c090 x8 : 0000000000000000 x7 : 0000000000000000 x6 : 0000000000000000 x5 : 0000000000000000 x4 : 0000000000000000 x3 : 0000000000000000 x2 : 0000000000007f10 x1 : 0000000000000522 x0 : ffff8000827e0522 Call trace: rtw_pci_read32+0x18/0x40 [rtw88_pci] (P) rtw_hw_scan_chan_switch+0x124/0x1a8 [rtw88_core] rtw_fw_c2h_cmd_handle+0x254/0x290 [rtw88_core] rtw_c2h_work+0x50/0x98 [rtw88_core] process_one_work+0x178/0x3f8 worker_thread+0x208/0x418 kthread+0x120/0x220 ret_from_fork+0x10/0x20 Code: d28fe202 8b020000 f9524400 8b214000 (b9400000) ---[end trace 0000000000000000]---
CVE-2025-71222	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: wifi: wlcore: ensure skb headroom before skb_push This avoids occasional skb_under_panic Oops from wl1271_tx_work. In this case, headroom is less than needed (typically 110 - 94 = 16 bytes).

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-39681	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: x86/cpu/hygon: Add missing resctrl_cpu_detect() in bsp_init helper Since 923f3a2b48bd ("x86/resctrl: Query LLC monitoring properties once during boot") resctrl_cpu_detect() has been moved from common CPU initialization code to the vendor-specific BSP init helper, while Hygon didn't put that call in their code. This triggers a division by zero fault during early booting stage on our machines with X86_FEATURE_CQM* supported, where get_rdt_mon_resources() tries to calculate mon_l3_config with uninitialized boot_cpu_data.x86_cache_occ_scale. Add the missing resctrl_cpu_detect() in the Hygon BSP init helper. [bp: Massage commit message.]
CVE-2026-23005	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: x86/fpu: Clear XSTATE_BV[i] in guest XSAVE state whenever XFD[i]=1 When loading guest XSAVE state via KVM_SET_XSAVE, and when updating XFD in response to a guest WRMSR, clear XFD-disabled features in the saved (or to be restored) XSTATE_BV to ensure KVM doesn't attempt to load state for features that are disabled via the guest's XFD. Because the kernel executes XRSTOR with the guest's XFD, saving XSTATE_BV[i]=1 with XFD[i]=1 will cause XRSTOR to #NM and panic the kernel. E.g. if fpu_update_guest_xfd() sets XFD without clearing XSTATE_BV: ------------[cut here]------------ WARNING: arch/x86/kernel/traps.c:1524 at exc_device_not_available+0x101/0x110, CPU#29: amx_test/848 Modules linked in: kvm_intel kvm irqbypass CPU: 29 UID: 1000 PID: 848 Comm: amx_test Not tainted 6.19.0-rc2-ffa07f7fd437-x86_amx_nm_xfd_non_init-vm #171 NONE Hardware name: QEMU Standard PC (Q35 + ICH9, 2009), BIOS 0.0.0 02/06/2015 RIP: 0010:exc_device_not_available+0x101/0x110 Call Trace: asm_exc_device_not_available+0x1a/0x20 RIP: 0010:restore_fpregs_from_fpstate+0x36/0x90 switch_fpu_return+0x4a/0xb0 kvm_arch_vcpu_ioctl_run+0x1245/0x1e40 [kvm] kvm_vcpu_ioctl+0x2c3/0x8f0 [kvm] __x64_sys_ioctl+0x8f/0xd0 do_syscall_64+0x62/0x940 entry_SYSCALL_64_after_hwframe+0x4b/0x53 ---[end trace 0000000000000000]--- This can happen if the guest executes WRMSR(MSR_IA32_XFD) to set XFD[18] = 1, and a host IRQ triggers kernel_fpu_begin() prior to the vmexit handler's call to fpu_update_guest_xfd(). and if userspace stuffs XSTATE_BV[i]=1 via KVM_SET_XSAVE: ------------[cut here]------------ WARNING: arch/x86/kernel/traps.c:1524 at exc_device_not_available+0x101/0x110, CPU#14: amx_test/867 Modules linked in: kvm_intel kvm irqbypass CPU: 14 UID: 1000 PID: 867 Comm: amx_test Not tainted 6.19.0-rc2-2dace9faccd6-x86_amx_nm_xfd_non_init-vm #168 NONE Hardware name: QEMU Standard PC (Q35 + ICH9, 2009), BIOS 0.0.0 02/06/2015 RIP: 0010:exc_device_not_available+0x101/0x110 Call Trace: asm_exc_device_not_available+0x1a/0x20 RIP: 0010:restore_fpregs_from_fpstate+0x36/0x90 fpu_swap_kvm_fpstate+0x6b/0x120 kvm_load_guest_fpu+0x30/0x80 [kvm] kvm_arch_vcpu_ioctl_run+0x85/0x1e40 [kvm] kvm_vcpu_ioctl+0x2c3/0x8f0 [kvm] __x64_sys_ioctl+0x8f/0xd0 do_syscall_64+0x62/0x940 entry_SYSCALL_64_after_hwframe+0x4b/0x53 ---[end trace 0000000000000000]--- The new behavior is consistent with the AMX architecture. Per Intel's SDM, XSAVE saves XSTATE_BV as '0' for components that are disabled via XFD (and non-compacted XSAVE saves the initial configuration of the state component): If XSAVE, XSAVEC, XSAVEOPT, or XSAVES is saving the

CVE	Severity	CVSS	CVSS Vector	Component	Description
					state component i, the instruction does not generate #NM when XCR0[i] = IA32_XFD[i] = 1; instead, it operates as if XINUSE[i] = 0 (and the state component was in its initial state): it saves bit i of XSTATE_BV field of the XSAVE header as 0; in addition, XSAVE saves the initial configuration of the state component (the other instructions do not save state component i). Alternatively, KVM could always do XRSTOR with XFD=0, e.g. by using a constant XFD based on the set of enabled features when XSAVEing for a struct fpu_guest. However, having XSTATE_BV[i]=1 for XFD-disabled features can only happen in the above interrupt case, or in similar scenarios involving preemption on preemptible kernels, because fpu_swap_kvm_fpstate()'s call to save_fpregs_to_fpstate() saves the outgoing FPU state with the current XFD; and that is (on all but the first WRMSR to XFD) the guest XFD. Therefore, XFD can only go out of sync with XSTATE_BV in the above interrupt case, or in similar scenarios involving preemption on preemptible kernels, and it we can consider it (de facto) part of KVM ABI that KVM_GET_XSAVE returns XSTATE_BV[i]=0 for XFD-disabled features. [Move clea ---truncated---
CVE-2025-38560	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: x86/sev: Evict cache lines during SNP memory validation An SNP cache coherency vulnerability requires a cache line eviction mitigation when validating memory after a page state change to private. The specific mitigation is to touch the first and last byte of each 4K page that is being validated. There is no need to perform the mitigation when performing a page state change to shared and rescinding validation. CPUID bit Fn8000001F_EBX[31] defines the COHERENCY_SFW_NO CPUID bit that, when set, indicates that the software mitigation for this vulnerability is not needed. Implement the mitigation and invoke it when validating memory (making it private) and the COHERENCY_SFW_NO bit is not set, indicating the SNP guest is vulnerable.
CVE-2025-40300	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: x86/vmscape: Add conditional IBPB mitigation VMSCAPE is a vulnerability that exploits insufficient branch predictor isolation between a guest and a userspace hypervisor (like QEMU). Existing mitigations already protect kernel/KVM from a malicious guest. Userspace can additionally be protected by flushing the branch predictors after a VMexit. Since it is the userspace that consumes the poisoned branch predictors, conditionally issue an IBPB after a VMexit and before returning to userspace. Workloads that frequently switch between hypervisor and userspace will incur the most overhead from the new IBPB. This new IBPB is not integrated with the existing IBPB sites. For instance, a task can use the existing speculation control prctl() to get an IBPB at context switch time. With this implementation, the IBPB is doubled up: one at context switch and another before running userspace. The intent is to integrate and optimize these cases post-embargo. [dhansen: elaborate on suboptimal IBPB solution]

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2024-50090	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: drm/xe/oa: Fix overflow in oa batch buffer By default xe_bb_create_job() appends a MI_BATCH_BUFFER_END to batch buffer, this is not a problem if batch buffer is only used once but oa reuses the batch buffer for the same metric and at each call it appends a MI_BATCH_BUFFER_END, printing the warning below and then overflowing. [381.072016] ----------[cut here]---------- [381.072019] xe 0000:00:02.0: [drm] Assertion `bb->len * 4 + bb_prefetch(q->gt) <= size` failed! platform: LUNARLAKE subplatform: 1 graphics: Xe2_LPG / Xe2_HPG 20.04 step B0 media: Xe2_LPM / Xe2_HPM 20.00 step B0 tile: 0 VRAM 0 B GT: 0 type 1 So here checking if batch buffer already have MI_BATCH_BUFFER_END if not append it. v2: - simply fix, suggestion from Ashutosh (cherry picked from commit 9ba0e0f30ca42a98af3689460063edfb6315718a)
CVE-2024-26785	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: iommufd: Fix protection fault in iommufd_test_syz_conv_iova Syzkaller reported the following bug: general protection fault, probably for non-canonical address 0xdfffffc0000000038: 0000 [#1] SMP KASAN KASAN: null-ptr-deref in range [0x0000000000001c0-0x0000000000001c7] Call Trace: lock_acquire lock_acquire+0x1ce/0x4f0 down_read+0x93/0x4a0 iommufd_test_syz_conv_iova+0x56/0x1f0 iommufd_test_access_rw.isra.0+0x2ec/0x390 iommufd_test+0x1058/0x1e30 iommufd_fops_ioctl+0x381/0x510 vfs_ioctl __do_sys_ioctl __se_sys_ioctl __x64_sys_ioctl+0x170/0x1e0 do_syscall_x64 do_syscall_64+0x71/0x140 This is because the new iommufd_access_change_ioas() sets access->ioas to NULL during its process, so the lock might be gone in a concurrent racing context. Fix this by doing the same access->ioas sanity as iommufd_access_rw() and iommufd_access_pin_pages() functions do.
CVE-2025-38391	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: usb: typec: altmodes/displayport: do not index invalid pin_assignments A poorly implemented DisplayPort Alt Mode port partner can indicate that its pin assignment capabilities are greater than the maximum value, DP_PIN_ASSIGN_F. In this case, calls to pin_assignment_show will cause a BRK exception due to an out of bounds array access. Prevent for loop in pin_assignment_show from accessing invalid values in pin_assignments by adding DP_PIN_ASSIGN_MAX value in typec_dp.h and using i < DP_PIN_ASSIGN_MAX as a loop condition.
CVE-2024-49994	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: block: fix integer overflow in BLKSECDISCARD I independently rediscovered commit 22d24a544b0d49bbcbd61c8c0eaf77d3c9297155 block: fix overflow in blk_ioctl_discard() but for secure erase. Same problem: uint64_t r[2] = {512, 18446744073709551104ULL}; ioctl(fd, BLKSECDISCARD, r); will enter near infinite loop inside blkdev_issue_secure_erase(): a.out: attempt to access beyond end of device loop0: rw=5, sector=3399043073, nr_sectors = 1024 limit=2048 bio_check_eod: 3286214 callbacks suppressed

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-37984	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: crypto: ecdsa - Harden against integer overflows in DIV_ROUND_UP() Herbert notes that DIV_ROUND_UP() may overflow unnecessarily if an ecdsa implementation's ->key_size() callback returns an unusually large value. Herbert instead suggests (for a division by 8): X / 8 + !!(X & 7) Based on this formula, introduce a generic DIV_ROUND_UP_POW2() macro and use it in lieu of DIV_ROUND_UP() for ->key_size() return values. Additionally, use the macro in ecc_digits_from_bytes(), whose "nbytes" parameter is a ->key_size() return value in some instances, or a user-specified ASN.1 length in the case of ecdsa_get_signature_rs().
CVE-2025-39940	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: dm-stripe: fix a possible integer overflow There's a possible integer overflow in stripe_io_hints if we have too large chunk size. Test if the overflow happened, and if it did, don't set limits->io_min and limits->io_opt;
CVE-2024-42066	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: drm/xe: Fix potential integer overflow in page size calculation Explicitly cast tbo->page_alignment to u64 before bit-shifting to prevent overflow when assigning to min_page_size.
CVE-2024-53187	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: io_uring: check for overflows in io_pin_pages WARNING: CPU: 0 PID: 5834 at io_uring/memmap.c:144 io_pin_pages+0x149/0x180 io_uring/memmap.c:144 CPU: 0 UID: 0 PID: 5834 Comm: syz-executor825 Not tainted 6.12.0-next-20241118-syzkaller #0 Call Trace: __io_uaddr_map+0xfb/0x2d0 io_uring/memmap.c:183 io_rings_map io_uring/io_uring.c:2611 [inline] io_allocate_scq_urings+0x1c0/0x650 io_uring/io_uring.c:3470 io_uring_create+0x5b5/0xc00 io_uring/io_uring.c:3692 io_uring_setup io_uring/io_uring.c:3781 [inline] ... io_pin_pages()'s uaddr parameter came directly from the user and can be garbage. Don't just add size to it as it can overflow.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38463	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: tcp: Correct signedness in skb remaining space calculation Syzkaller reported a bug [1] where sk->sk_forward_alloc can overflow. When we send data, if an skb exists at the tail of the write queue, the kernel will attempt to append the new data to that skb. However, the code that checks for available space in the skb is flawed: "' copy = size_goal - skb->len "' The types of the variables involved are: "' copy: ssize_t (s64 on 64-bit systems) size_goal: int skb->len: unsigned int "' Due to C's type promotion rules, the signed size_goal is converted to an unsigned int to match skb->len before the subtraction. The result is an unsigned int. When this unsigned int result is then assigned to the s64 copy variable, it is zero-extended, preserving its non-negative value. Consequently, copy is always >= 0. Assume we are sending 2GB of data and size_goal has been adjusted to a value smaller than skb->len. The subtraction will result in copy holding a very large positive integer. In the subsequent logic, this large value is used to update sk->sk_forward_alloc, which can easily cause it to overflow. The syzkaller reproducer uses TCP_REPAIR to reliably create this condition. However, this can also occur in real-world scenarios. The tcp_bound_to_half_wnd() function can also reduce size_goal to a small value. This would cause the subsequent tcp_wmem_schedule() to set sk->sk_forward_alloc to a value close to INT_MAX. Further memory allocation requests would then cause sk_forward_alloc to wrap around and become negative. [1]: https://syzkaller.appspot.com/bug?extid=de6565462ab540f50e47
CVE-2026-23069	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: vsock/virtio: fix potential underflow in virtio_transport_get_credit() The credit calculation in virtio_transport_get_credit() uses unsigned arithmetic: ret = vvs->peer_buf_alloc - (vvs->tx_cnt - vvs->peer_fwd_cnt); If the peer shrinks its advertised buffer (peer_buf_alloc) while bytes are in flight, the subtraction can underflow and produce a large positive value, potentially allowing more data to be queued than the peer can handle. Reuse virtio_transport_has_space() which already handles this case and add a comment to make it clear why we are doing that. [Stefano: use virtio_transport_has_space() instead of duplicating the code] [Stefano: tweak the commit message]

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-71087	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: iavf: fix off-by-one issues in iavf_config_rss_reg() There are off-by-one bugs when configuring RSS hash key and lookup table, causing out-of-bounds reads to memory [1] and out-of-bounds writes to device registers. Before commit 43a3d9ba34c9 ("i40evf: Allow PF driver to configure RSS"), the loop upper bounds were: i <= I40E_VFQF_{HKEY,HLUT}_MAX_INDEX which is safe since the value is the last valid index. That commit changed the bounds to: i <= adapter->rss_{key,lut}_size / 4 where `rss_{key,lut}_size / 4` is the number of dwords, so the last valid index is `(rss_{key,lut}_size / 4) - 1`. Therefore, using `<=` accesses one element past the end. Fix the issues by using `<` instead of `<=`, ensuring we do not exceed the bounds. [1] KASAN splat about rss_key_size off-by-one BUG: KASAN: slab-out-of-bounds in iavf_config_rss+0x619/0x800 Read of size 4 at addr ffff888102c50134 by task kworker/u8:6/63 CPU: 0 UID: 0 PID: 63 Comm: kworker/u8:6 Not tainted 6.18.0-rc2-enjuk-tnguy-00378-g3005f5b77652-dirty #156 PREEMPT(voluntary) Hardware name: QEMU Standard PC (Q35 + ICH9, 2009), BIOS 1.16.3-debian-1.16.3-2 04/01/2014 Workqueue: iavf iavf_watchdog_task Call Trace: dump_stack_lvl+0x6f/0xb0 print_report+0x170/0x4f3 kasan_report+0xe1/0x1a0 iavf_config_rss+0x619/0x800 iavf_watchdog_task+0x2be7/0x3230 process_one_work+0x7fd/0x1420 worker_thread+0x4d1/0xd40 kthread+0x344/0x660 ret_from_fork+0x249/0x320 ret_from_fork_asm+0x1a/0x30 Allocated by task 63: kasan_save_stack+0x30/0x50 kasan_save_track+0x14/0x30 __kasan_kmalloc+0x7f/0x90 __kmalloc_noprof+0x246/0x6f0 iavf_watchdog_task+0x28fc/0x3230 process_one_work+0x7fd/0x1420 worker_thread+0x4d1/0xd40 kthread+0x344/0x660 ret_from_fork+0x249/0x320 ret_from_fork_asm+0x1a/0x30 The buggy address belongs to the object at ffff888102c50100 which belongs to the cache kmalloc-64 of size 64 The buggy address is located 0 bytes to the right of allocated 52-byte region [ffff888102c50100, ffff888102c50134) The buggy address belongs to the physical page: page: refcount:0 mapcount:0 mapping:0000000000000000 index:0x0 pfn:0x102c50 flags: 0x200000000000000(node=0|zone=2) page_type: f5(slab) raw: 0200000000000000 ffff8881000418c0 dead000000000122 0000000000000000 raw: 0000000000000000 0000000080200020 00000000f5000000 0000000000000000 page dumped because: kasan: bad access detected Memory state around the buggy address: ffff888102c50000: 00 00 00 00 00 00 00 00 fc fc fc fc fc fc fc fc ffff888102c50080: 00 00 00 00 00 00 00 00 fc fc fc fc fc fc fc fc >ffff888102c50100: 00 00 00 00 00 00 00 04 fc fc fc fc fc fc fc fc ^ ffff888102c50180: 00 00 00 00 00 00 00 00 fc fc fc fc fc fc fc fc ffff888102c50200: fc fc fc fc fc fc fc fc fc fc fc fc fc fc fc fc
CVE-2025-38602	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: iwlwifi: Add missing check for alloc_ordered_workqueue Add check for the return value of alloc_ordered_workqueue since it may return NULL pointer.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-22026	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: nfsd: don't ignore the return code of svc_proc_register() Currently, nfsd_proc_stat_init() ignores the return value of svc_proc_register(). If the procfile creation fails, then the kernel will WARN when it tries to remove the entry later. Fix nfsd_proc_stat_init() to return the same type of pointer as svc_proc_register(), and fix up nfsd_net_init() to check that and fail the nfsd_net construction if it occurs. svc_proc_register() can fail if the dentry can't be allocated, or if an identical dentry already exists. The second case is pretty unlikely in the nfsd_net construction codepath, so if this happens, return -ENOMEM.
CVE-2025-39742	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: RDMA: hfi1: fix possible divide-by-zero in find_hw_thread_mask() The function divides number of online CPUs by num_core_siblings, and later checks the divider by zero. This implies a possibility to get and divide-by-zero runtime error. Fix it by moving the check prior to division. This also helps to save one indentation level.
CVE-2025-39909	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: mm/damon/lru_sort: avoid divide-by-zero in damon_lru_sort_apply_parameters() Patch series "mm/damon: avoid divide-by-zero in DAMON module's parameters application". DAMON's RECLAIM and LRU_SORT modules perform no validation on user-configured parameters during application, which may lead to division-by-zero errors. Avoid the divide-by-zero by adding validation checks when DAMON modules attempt to apply the parameters. This patch (of 2): During the calculation of 'hot_thres' and 'cold_thres', either 'sample_interval' or 'aggr_interval' is used as the divisor, which may lead to division-by-zero errors. Fix it by directly returning -EINVAL when such a case occurs. Additionally, since 'aggr_interval' is already required to be set no smaller than 'sample_interval' in damon_set_attrs(), only the case where 'sample_interval' is zero needs to be checked.
CVE-2025-39916	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: mm/damon/reclaim: avoid divide-by-zero in damon_reclaim_apply_parameters() When creating a new scheme of DAMON_RECLAIM, the calculation of 'min_age_region' uses 'aggr_interval' as the divisor, which may lead to division-by-zero errors. Fix it by directly returning -EINVAL when such a case occurs.
CVE-2024-25740	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	A memory leak flaw was found in the UBI driver in drivers/mtd/ubi/attach.c in the Linux kernel through 6.7.4 for UBI_IOCATT, because kobj->name is not released.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38438	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ASoC: SOF: Intel: hda: Use devm_kstrdup() to avoid memleak. sof_pdata->tplg_filename can have address allocated by kstrdup() and can be overwritten. Memory leak was detected with kmemleak: unreferenced object 0xffff88812391ff60 (size 16): comm "kworker/4:1", pid 161, jiffies 4294802931 hex dump (first 16 bytes): 73 6f 66 2d 68 64 61 2d 67 65 6e 65 72 69 63 00 sof-hda-generic. backtrace (crc 4bf1675c): __kmalloc_node_track_caller_noprof+0x49c/0x6b0 kstrdup+0x46/0xc0 hda_machine_select.cold+0x1de/0x12cf [snd_sof_intel_hda_generic] sof_init_environment+0x16f/0xb50 [snd_sof] sof_probe_continue+0x45/0x7c0 [snd_sof] sof_probe_work+0x1e/0x40 [snd_sof] process_one_work+0x894/0x14b0 worker_thread+0x5e5/0xfb0 kthread+0x39d/0x760 ret_from_fork+0x31/0x70 ret_from_fork_asm+0x1a/0x30
CVE-2026-23190	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ASoC: amd: fix memory leak in acp3x pdm dma ops
CVE-2025-71081	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ASoC: stm32: sai: fix OF node leak on probe The reference taken to the sync provider OF node when probing the platform device is currently only dropped if the set_sync() callback fails during DAI probe. Make sure to drop the reference on platform probe failures (e.g. probe deferral) and on driver unbind. This also avoids a potential use-after-free in case the DAI is ever reprobed without first rebinding the platform driver.
CVE-2025-71147	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: KEYS: trusted: Fix a memory leak in tpm2_load_cmd 'tpm2_load_cmd' allocates a tempoary blob indirectly via 'tpm2_key_decode' but it is not freed in the failure paths. Address this by wrapping the blob into with a cleanup helper.
CVE-2025-38546	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: atm: clip: Fix memory leak of struct clip_vcc. ioctl(ATMARP_MKIP) allocates struct clip_vcc and set it to vcc->user_back. The code assumes that vcc_destroy_socket() passes NULL skb to vcc->push() when the socket is close()d, and then clip_push() frees clip_vcc. However, ioctl(ATMARPD_CTRL) sets NULL to vcc->push() in atm_init_atmarp(), resulting in memory leak. Let's serialise two ioctl() by lock_sock() and check vcc->push() in atm_init_atmarp() to prevent memleak.
CVE-2025-39848	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ax25: properly unshare skbs in ax25_kiss_rcv() Bernard Pidoux reported a regression apparently caused by commit c353e8983e0d ("net: introduce per netns packet chains"). skb->dev becomes NULL and we crash in __netif_receive_skb_core(). Before above commit, different kind of bugs or corruptions could happen without a major crash. But the root cause is that ax25_kiss_rcv() can queue/mangle input skb without checking if this skb is shared or not. Many thanks to Bernard Pidoux for his help, diagnosis and tests. We had a similar issue years ago fixed with commit 7aaed57c5c28 ("phonet: properly unshare skbs in phonet_rcv()").

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-37807	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: bpf: Fix kmemleak warning for percpu hashmap Vlad Poenaru reported the following kmemleak issue: unreferenced object 0x606fd7c44ac8 (size 32): backtrace (crc 0): pcpu_alloc_noprof+0x730/0xeb0 bpf_map_alloc_percpu+0x69/0xc0 prealloc_init+0x9d/0x1b0 htab_map_alloc+0x363/0x510 map_create+0x215/0x3a0 __sys_bpf+0x16b/0x3e0 __x64_sys_bpf+0x18/0x20 do_syscall_64+0x7b/0x150 entry_SYSCALL_64_after_hwframe+0x4b/0x53 Further investigation shows the reason is due to not 8-byte aligned store of percpu pointer in htab_elem_set_ptr(): *(void __percpu **)(l->key + key_size) = pptr; Note that the whole htab_elem alignment is 8 (for x86_64). If the key_size is 4, that means pptr is stored in a location which is 4 byte aligned but not 8 byte aligned. In mm/kmemleak.c, scan_block() scans the memory based on 8 byte stride, so it won't detect above pptr, hence reporting the memory leak. In htab_map_alloc(), we already have htab->elem_size = sizeof(struct htab_elem) + round_up(htab->map.key_size, 8); if (percpu) htab->elem_size += sizeof(void *); else htab->elem_size += round_up(htab->map.value_size, 8); So storing pptr with 8-byte alignment won't cause any problem and can fix kmemleak too. The issue can be reproduced with bpf selftest as well: 1. Enable CONFIG_DEBUG_KMEMLEAK config 2. Add a getchar() before skel destroy in test_hash_map() in prog_tests/for_each.c. The purpose is to keep map available so kmemleak can be detected. 3. run './test_progs -t for_each/hash_map &' and a kmemleak should be reported.
CVE-2026-23075	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: can: esd_usb: esd_usb_read_bulk_callback(): fix URB memory leak Fix similar memory leak as in commit 7352e1d5932a ("can: gs_usb: gs_usb_receive_bulk_callback(): fix URB memory leak"). In esd_usb_open(), the URBs for USB-in transfers are allocated, added to the dev->rx_submitted anchor and submitted. In the complete callback esd_usb_read_bulk_callback(), the URBs are processed and resubmitted. In esd_usb_close() the URBs are freed by calling usb_kill_anchored_urbs(&dev->rx_submitted). However, this does not take into account that the USB framework unanchors the URB before the complete function is called. This means that once an in-URB has been completed, it is no longer anchored and is ultimately not released in esd_usb_close(). Fix the memory leak by anchoring the URB in the esd_usb_read_bulk_callback() to the dev->rx_submitted anchor.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2026-23061	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: can: kvaser_usb: kvaser_usb_read_bulk_callback(): fix URB memory leak Fix similar memory leak as in commit 7352e1d5932a ("can: gs_usb: gs_usb_receive_bulk_callback(): fix URB memory leak"). In kvaser_usb_set_{,data_}bittiming() -> kvaser_usb_setup_rx_urbs(), the URBs for USB-in transfers are allocated, added to the dev->rx_submitted anchor and submitted. In the complete callback kvaser_usb_read_bulk_callback(), the URBs are processed and resubmitted. In kvaser_usb_remove_interfaces() the URBs are freed by calling usb_kill_anchored_urbs(&dev->rx_submitted). However, this does not take into account that the USB framework unanchors the URB before the complete function is called. This means that once an in-URB has been completed, it is no longer anchored and is ultimately not released in usb_kill_anchored_urbs(). Fix the memory leak by anchoring the URB in the kvaser_usb_read_bulk_callback() to the dev->rx_submitted anchor.
CVE-2026-23080	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: can: mcba_usb: mcba_usb_read_bulk_callback(): fix URB memory leak Fix similar memory leak as in commit 7352e1d5932a ("can: gs_usb: gs_usb_receive_bulk_callback(): fix URB memory leak"). In mcba_usb_probe() -> mcba_usb_start(), the URBs for USB-in transfers are allocated, added to the priv->rx_submitted anchor and submitted. In the complete callback mcba_usb_read_bulk_callback(), the URBs are processed and resubmitted. In mcba_usb_close() -> mcba_urb_unlink() the URBs are freed by calling usb_kill_anchored_urbs(&priv->rx_submitted). However, this does not take into account that the USB framework unanchors the URB before the complete function is called. This means that once an in-URB has been completed, it is no longer anchored and is ultimately not released in usb_kill_anchored_urbs(). Fix the memory leak by anchoring the URB in the mcba_usb_read_bulk_callback()to the priv->rx_submitted anchor.
CVE-2026-23108	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: can: usb_8dev: usb_8dev_read_bulk_callback(): fix URB memory leak Fix similar memory leak as in commit 7352e1d5932a ("can: gs_usb: gs_usb_receive_bulk_callback(): fix URB memory leak"). In usb_8dev_open() -> usb_8dev_start(), the URBs for USB-in transfers are allocated, added to the priv->rx_submitted anchor and submitted. In the complete callback usb_8dev_read_bulk_callback(), the URBs are processed and resubmitted. In usb_8dev_close() -> unlink_all_urbs() the URBs are freed by calling usb_kill_anchored_urbs(&priv->rx_submitted). However, this does not take into account that the USB framework unanchors the URB before the complete function is called. This means that once an in-URB has been completed, it is no longer anchored and is ultimately not released in usb_kill_anchored_urbs(). Fix the memory leak by anchoring the URB in the usb_8dev_read_bulk_callback() to the priv->rx_submitted anchor.
CVE-2025-71189	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: dmaengine: dw: dmamux: fix OF node leak on route allocation failure Make sure to drop the reference taken to the DMA master OF node also on late route allocation failures.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-71163	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: dmaengine: idxd: fix device leaks on compat bind and unbind Make sure to drop the reference taken when looking up the idxd device as part of the compat bind and unbind sysfs interface.
CVE-2025-71188	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: dmaengine: lpc18xx-dmamux: fix device leak on route allocation Make sure to drop the reference taken when looking up the DMA mux platform device during route allocation. Note that holding a reference to a device does not prevent its driver data from going away so there is no point in keeping the reference.
CVE-2026-23026	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: dmaengine: qcom: gpi: Fix memory leak in gpi_peripheral_config() Fix a memory leak in gpi_peripheral_config() where the original memory pointed to by gchan->config could be lost if krealloc() fails. The issue occurs when: 1. gchan->config points to previously allocated memory 2. krealloc() fails and returns NULL 3. The function directly assigns NULL to gchan->config, losing the reference to the original memory 4. The original memory becomes unreachable and cannot be freed Fix this by using a temporary variable to hold the krealloc() result and only updating gchan->config when the allocation succeeds. Found via static analysis and code review.
CVE-2025-71186	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: dmaengine: stm32: dmamux: fix device leak on route allocation Make sure to drop the reference taken when looking up the DMA mux platform device during route allocation. Note that holding a reference to a device does not prevent its driver data from going away so there is no point in keeping the reference.
CVE-2025-71185	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: dmaengine: ti: dma-crossbar: fix device leak on am335x route allocation Make sure to drop the reference taken when looking up the crossbar platform device during am335x route allocation.
CVE-2025-38011	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: drm/amdgpu: csa unmap use uninterruptible lock After process exit to unmap csa and free GPU vm, if signal is accepted and then waiting to take vm lock is interrupted and return, it causes memory leaking and below warning backtrace. Change to use uninterruptible wait lock fix the issue. WARNING: CPU: 69 PID: 167800 at amd/amdgpu/amdgpu_kms.c:1525 amdgpu_driver_postclose_kms+0x294/0x2a0 [amdgpu] Call Trace: drm_file_free.part.0+0x1da/0x230 [drm] drm_close_helper.isra.0+0x65/0x70 [drm] drm_release+0x6a/0x120 [drm] amdgpu_drm_release+0x51/0x60 [amdgpu] __fput+0x9f/0x280 ____fput+0xe/0x20 task_work_run+0x67/0xa0 do_exit+0x217/0x3c0 do_group_exit+0x3b/0xb0 get_signal+0x14a/0x8d0 arch_do_signal_or_restart+0xde/0x100 exit_to_user_mode_loop+0xc1/0x1a0 exit_to_user_mode_prepare+0xf4/0x100 syscall_exit_to_user_mode+0x17/0x40 do_syscall_64+0x69/0xc0 (cherry picked from commit 7dbbfb3c171a6f63b01165958629c9c26abf38ab)

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2024-53084	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: drm/imagination: Break an object reference loop When remaining resources are being cleaned up on driver close, outstanding VM mappings may result in resources being leaked, due to an object reference loop, as shown below, with each object (or set of objects) referencing the object below it: PVR GEM Object GPU scheduler "finished" fence GPU scheduler “scheduled” fence PVR driver “done” fence PVR Context PVR VM Context PVR VM Mappings PVR GEM Object The reference that the PVR VM Context has on the VM mappings is a soft one, in the sense that the freeing of outstanding VM mappings is done as part of VM context destruction; no reference counts are involved, as is the case for all the other references in the loop. To break the reference loop during cleanup, free the outstanding VM mappings before destroying the PVR Context associated with the VM context.
CVE-2026-23170	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: drm/imx/tve: fix probe device leak Make sure to drop the reference taken to the DDC device during probe on probe failure (e.g. probe deferral) and on driver unbind.
CVE-2025-38410	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: drm/msm: Fix a fence leak in submit error path In error paths, we could unref the submit without calling drm_sched_entity_push_job(), so msm_job_free() will never get called. Since drm_sched_job_cleanup() will NULL out the s_fence, we can use that to detect this case. Patchwork: https://patchwork.freedesktop.org/patch/653584/
CVE-2025-38409	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: drm/msm: Fix another leak in the submit error path put_unused_fd() doesn't free the installed file, if we've already done fd_install(). So we need to also free the sync_file. Patchwork: https://patchwork.freedesktop.org/patch/653583/
CVE-2025-39679	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: drm/nouveau/nvif: Fix potential memory leak in nvif_vmm_ctor(). When the nvif_vmm_type is invalid, we will return error directly without freeing the args in nvif_vmm_ctor(), which leading a memory leak. Fix it by setting the ret -EINVAL and goto done.
CVE-2025-38057	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: espintcp: fix skb leaks A few error paths are missing a kfree_skb.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-39756	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: fs: Prevent file descriptor table allocations exceeding INT_MAX When sysctl_nr_open is set to a very high value (for example, 1073741816 as set by systemd), processes attempting to use file descriptors near the limit can trigger massive memory allocation attempts that exceed INT_MAX, resulting in a WARNING in mm/slub.c: WARNING: CPU: 0 PID: 44 at mm/slub.c:5027 __kvmalloc_node_noprof+0x21a/0x288 This happens because kvmalloc_array() and kvmalloc() check if the requested size exceeds INT_MAX and emit a warning when the allocation is not flagged with __GFP_NOWARN. Specifically, when nr_open is set to 1073741816 (0x3ffffff8) and a process calls dup2(oldfd, 1073741880), the kernel attempts to allocate: - File descriptor array: 1073741880 * 8 bytes = 8,589,935,040 bytes - Multiple bitmaps: ~400MB - Total allocation size: > 8GB (exceeding INT_MAX = 2,147,483,647) Reproducer: 1. Set /proc/sys/fs/nr_open to 1073741816: # echo 1073741816 > /proc/sys/fs/nr_open 2. Run a program that uses a high file descriptor: #include #include int main() { struct rlimit rlim = {1073741824, 1073741824}; setrlimit(RLIMIT_NOFILE, &rlim); dup2(2, 1073741880); // Triggers the warning return 0; } 3. Observe WARNING in dmesg at mm/slub.c:5027 systemd commit a8b627a introduced automatic bumping of fs.nr_open to the maximum possible value. The rationale was that systems with memory control groups (memcg) no longer need separate file descriptor limits since memory is properly accounted. However, this change overlooked that: 1. The kernel's allocation functions still enforce INT_MAX as a maximum size regardless of memcg accounting 2. Programs and tests that legitimately test file descriptor limits can inadvertently trigger massive allocations 3. The resulting allocations (>8GB) are impractical and will always fail systemd's algorithm starts with INT_MAX and keeps halving the value until the kernel accepts it. On most systems, this results in nr_open being set to 1073741816 (0x3ffffff8), which is just under 1GB of file descriptors. While processes rarely use file descriptors near this limit in normal operation, certain selftests (like tools/testing/selftests/core/unshare_test.c) and programs that test file descriptor limits can trigger this issue. Fix this by adding a check in alloc_fdtable() to ensure the requested allocation size does not exceed INT_MAX. This causes the operation to fail with -EMFILE instead of triggering a kernel warning and avoids the impractical >8GB memory allocation request.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2026-23095	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: gue: Fix skb memleak with inner IP protocol 0. syzbot reported skb memleak below. [0] The repro generated a GUE packet with its inner protocol 0. gue_udp_recv() returns -guehdr->proto_ctype for "resubmit" in ip_protocol_deliver_rcu(), but this only works with non-zero protocol number. Let's drop such packets. Note that 0 is a valid number (IPv6 Hop-by-Hop Option). I think it is not practical to encap HOPOPT in GUE, so once someone starts to complain, we could pass down a resubmit flag pointer to distinguish two zeros from the upper layer: * no error * resubmit HOPOPT [0] BUG: memory leak unreferenced object 0xffff888109695a00 (size 240): comm "syz.0.17", pid 6088, jiffies 4294943096 hex dump (first 32 bytes): 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ..... 00 40 c2 10 81 88 ff ff 00 00 00 00 00 00 00 00 .@..... backtrace (crc a84b336f): kmemleak_alloc_recursive include/linux/kmemleak.h:44 [inline] slab_post_alloc_hook mm/slub.c:4958 [inline] slab_alloc_node mm/slub.c:5263 [inline] kmem_cache_alloc_noprof+0x3b4/0x590 mm/slub.c:5270 __build_skb+0x23/0x60 net/core/skbuff.c:474 build_skb+0x20/0x190 net/core/skbuff.c:490 __tun_build_skb drivers/net/tun.c:1541 [inline] tun_build_skb+0x4a1/0xa40 drivers/net/tun.c:1636 tun_get_user+0xc12/0x2030 drivers/net/tun.c:1770 tun_chr_write_iter+0x71/0x120 drivers/net/tun.c:1999 new_sync_write fs/read_write.c:593 [inline] vfs_write+0x45d/0x710 fs/read_write.c:686 ksys_write+0xa7/0x170 fs/read_write.c:738 do_syscall_x64 arch/x86/entry/syscall_64.c:63 [inline] do_syscall_64+0xa4/0xf80 arch/x86/entry/syscall_64.c:94 entry_SYSCALL_64_after_hwframe+0x77/0x7f

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-39948	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ice: fix Rx page leak on multi-buffer frames The ice_put_rx_mbuf() function handles calling ice_put_rx_buf() for each buffer in the current frame. This function was introduced as part of handling multi-buffer XDP support in the ice driver. It works by iterating over the buffers from first_desc up to 1 plus the total number of fragments in the frame, cached from before the XDP program was executed. If the hardware posts a descriptor with a size of 0, the logic used in ice_put_rx_mbuf() breaks. Such descriptors get skipped and don't get added as fragments in ice_add_xdp_frag. Since the buffer isn't counted as a fragment, we do not iterate over it in ice_put_rx_mbuf(), and thus we don't call ice_put_rx_buf(). Because we don't call ice_put_rx_buf(), we don't attempt to re-use the page or free it. This leaves a stale page in the ring, as we don't increment next_to_alloc. The ice_reuse_rx_page() assumes that the next_to_alloc has been incremented properly, and that it always points to a buffer with a NULL page. Since this function doesn't check, it will happily recycle a page over the top of the next_to_alloc buffer, losing track of the old page. Note that this leak only occurs for multi-buffer frames. The ice_put_rx_mbuf() function always handles at least one buffer, so a single-buffer frame will always get handled correctly. It is not clear precisely why the hardware hands us descriptors with a size of 0 sometimes, but it happens somewhat regularly with "jumbo frames" used by 9K MTU. To fix ice_put_rx_mbuf(), we need to make sure to call ice_put_rx_buf() on all buffers between first_desc and next_to_clean. Borrow the logic of a similar function in i40e used for this same purpose. Use the same logic also in ice_get_pgcnts(). Instead of iterating over just the number of fragments, use a loop which iterates until the current index reaches to the next_to_clean element just past the current frame. Unlike i40e, the ice_put_rx_mbuf() function does call ice_put_rx_buf() on the last buffer of the frame indicating the end of packet. For non-linear (multi-buffer) frames, we need to take care when adjusting the pagecnt_bias. An XDP program might release fragments from the tail of the frame, in which case that fragment page is already released. Only update the pagecnt_bias for the first descriptor and fragments still remaining post-XDP program. Take care to only access the shared info for fragmented buffers, as this avoids a significant cache miss. The xdp_xmit value only needs to be updated if an XDP program is run, and only once per packet. Drop the xdp_xmit pointer argument from ice_put_rx_mbuf(). Instead, set xdp_xmit in the ice_clean_rx_irq() function directly. This avoids needing to pass the argument and avoids an extra bit-wise OR for each buffer in the frame. Move the increment of the ntc local variable to ensure its updated *before* all calls to ice_get_pgcnts() or ice_put_rx_mbuf(), as the loop logic requires the index of the element just after the current frame. Now that we use an index pointer in the ring to identify the packet, we no longer need to track or cache the number of fragments in the rx_ring.
CVE-2026-23091	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: intel_th: fix device leak on output open() Make sure to drop the reference taken when looking up the th device during output device open() on errors and on close(). Note that a recent commit fixed the leak in a couple of open() error paths but not all of them, and the reference is still leaking on successful open().

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-71153	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ksmbd: Fix memory leak in get_file_all_info() In get_file_all_info(), if vfs_getattr() fails, the function returns immediately without freeing the allocated filename, leading to a memory leak. Fix this by freeing the filename before returning in this error case.
CVE-2025-39737	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: mm/kmemleak: avoid soft lockup in __kmemleak_do_cleanup() A soft lockup warning was observed on a relative small system x86-64 system with 16 GB of memory when running a debug kernel with kmemleak enabled. watchdog: BUG: soft lockup - CPU#8 stuck for 33s! [kworker/8:1:134] The test system was running a workload with hot unplug happening in parallel. Then kemleak decided to disable itself due to its inability to allocate more kmemleak objects. The debug kernel has its CONFIG_DEBUG_KMEMLEAK_MEM_POOL_SIZE set to 40,000. The soft lockup happened in kmemleak_do_cleanup() when the existing kmemleak objects were being removed and deleted one-by-one in a loop via a workqueue. In this particular case, there are at least 40,000 objects that need to be processed and given the slowness of a debug kernel and the fact that a raw_spinlock has to be acquired and released in __delete_object(), it could take a while to properly handle all these objects. As kmemleak has been disabled in this case, the object removal and deletion process can be further optimized as locking isn't really needed. However, it is probably not worth the effort to optimize for such an edge case that should rarely happen. So the simple solution is to call cond_resched() at periodic interval in the iteration loop to avoid soft lockup.
CVE-2025-38384	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: mtd: spinand: fix memory leak of ECC engine conf Memory allocated for the ECC engine conf is not released during spinand cleanup. Below kmemleak trace is seen for this memory leak: unreferenced object 0xffffff80064f00e0 (size 8): comm "swapper/0", pid 1, jiffies 4294937458 hex dump (first 8 bytes): 00 00 00 00 00 00 00 00 .... backtrace (crc 0): kmemleak_alloc+0x30/0x40 __kmalloc_cache_noprof+0x208/0x3c0 spinand_ondie_ecc_init_ctx+0x114/0x200 nand_ecc_init_ctx+0x70/0xa8 nanddev_ecc_engine_init+0xec/0x27c spinand_probe+0xa2c/0x1620 spi_mem_probe+0x130/0x21c spi_probe+0xf0/0x170 really_probe+0x17c/0x6e8 __driver_probe_device+0x17c/0x21c driver_probe_device+0x58/0x180 __device_attach_driver+0x15c/0x1f8 bus_for_each_drv+0xec/0x150 __device_attach+0x188/0x24c device_initial_probe+0x10/0x20 bus_probe_device+0x11c/0x160 Fix the leak by calling nanddev_ecc_engine_cleanup() inside spinand_cleanup().

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38590	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net/mlx5e: Remove skb secpath if xfrm state is not found Hardware returns a unique identifier for a decrypted packet's xfrm state, this state is looked up in an xarray. However, the state might have been freed by the time of this lookup. Currently, if the state is not found, only a counter is incremented. The secpath (sp) extension on the skb is not removed, resulting in sp->len becoming 0. Subsequently, functions like __xfrm_policy_check() attempt to access fields such as xfrm_input_state(skb)->xso.type (which dereferences sp->xvec[sp->len - 1]) without first validating sp->len. This leads to a crash when dereferencing an invalid state pointer. This patch prevents the crash by explicitly removing the secpath extension from the skb if the xfrm state is not found after hardware decryption. This ensures downstream functions do not operate on a zero-length secpath. BUG: unable to handle page fault for address: ffffffff000002c8 #PF: supervisor read access in kernel mode #PF: error_code(0x0000) - not-present page PGD 282e067 P4D 282e067 PUD 0 Oops: Oops: 0000 [#1] SMP CPU: 12 UID: 0 PID: 0 Comm: swapper/12 Not tainted 6.15.0-rc7_for_upstream_min_debug_2025_05_27_22_44 #1 NONE Hardware name: QEMU Standard PC (Q35 + ICH9, 2009), BIOS rel-1.13.0-0-gf21b5a4aeb02-prebuilt.qemu.org 04/01/2014 RIP: 0010:__xfrm_policy_check+0x61a/0xa30 Code: b6 77 7f 83 e6 02 74 14 4d 8b af d8 00 00 00 41 0f b6 45 05 c1 e0 03 48 98 49 01 c5 41 8b 45 00 83 e8 01 48 98 49 8b 44 c5 10 <0f> b6 80 c8 02 00 00 83 e0 0c 3c 04 0f 84 0c 02 00 00 31 ff 80 fa RSP: 0018:ffff88885fb04918 EFLAGS: 00010297 RAX: ffffffff00000000 RBX: 0000000000000002 RCX: 0000000000000000 RDX: 0000000000000002 RSI: 0000000000000002 RDI: 0000000000000000 RBP: ffffffff8311af80 R08: 0000000000000020 R09: 00000000c2eda353 R10: ffff88812be2bbc8 R11: 000000001faab533 R12: ffff88885fb049c8 R13: ffff88812be2bbc8 R14: 0000000000000000 R15: ffff88811896ae00 FS: 0000000000000000(0000) GS:ffff8888dca82000(0000) knlGS:0000000000000000 CS: 0010 DS: 0000 ES: 0000 CR0: 0000000080050033 CR2: ffffffff000002c8 CR3: 0000000243050002 CR4: 0000000000372eb0 DR0: 0000000000000000 DR1: 0000000000000000 DR2: 0000000000000000 DR3: 0000000000000000 DR6: 00000000fffe0ff0 DR7: 0000000000000400 Call Trace: ? try_to_wake_up+0x108/0x4c0 ? udp4_lib_lookup2+0xbe/0x150 ? udp_lib_lport_inuse+0x100/0x100 ? __udp4_lib_lookup+0x2b0/0x410 __xfrm_policy_check2.constprop.0+0x11e/0x130 udp_queue_rcv_one_skb+0x1d/0x530 udp_unicast_rcv_skb+0x76/0x90 __udp4_lib_rcv+0xa64/0xe90 ip_protocol_deliver_rcu+0x20/0x130 ip_local_deliver_finish+0x75/0xa0 ip_local_deliver+0xc1/0xd0 ? ip_protocol_deliver_rcu+0x130/0x130 ip_sublist_rcv+0x1f9/0x240 ? ip_rcv_finish_core+0x430/0x430 ip_list_rcv+0xfc/0x130 __netif_receive_skb_list_core+0x181/0x1e0 netif_receive_skb_list_internal+0x200/0x360 ? mlx5e_build_rx_skb+0x1bc/0xda0 [mlx5_core] gro_receive_skb+0xfd/0x210 mlx5e_handle_rx_cqe_mpwrq+0x141/0x280 [mlx5_core] mlx5e_poll_rx_cq+0xcc/0x8e0 [mlx5_core] ? mlx5e_handle_rx_dim+0x91/0xd0 [mlx5_core] mlx5e_napi_poll+0x114/0xab0 [mlx5_core] __napi_poll+0x25/0x170 net_rx_action+0x32d/0x3a0 ? mlx5_eq_comp_int+0x8d/0x280 [mlx5_core] ? notifier_call_chain+0x33/0xa0 handle_softirqs+0xda/0x250 irq_exit_rcu+0x6d/0xc0

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2026-22979	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	common_interrupt+0x81/0xa0 In the Linux kernel, the following vulnerability has been resolved: net: fix memory leak in skb_segment_list for GRO packets When skb_segment_list() is called during packet forwarding, it handles packets that were aggregated by the GRO engine. Historically, the segmentation logic in skb_segment_list assumes that individual segments are split from a parent SKB and may need to carry their own socket memory accounting. Accordingly, the code transfers truesize from the parent to the newly created segments. Prior to commit ed4cccef64c1 ("gro: fix ownership transfer"), this truesize subtraction in skb_segment_list() was valid because fragments still carry a reference to the original socket. However, commit ed4cccef64c1 ("gro: fix ownership transfer") changed this behavior by ensuring that fraglist entries are explicitly orphaned (skb->sk = NULL) to prevent illegal orphaning later in the stack. This change meant that the entire socket memory charge remained with the head SKB, but the corresponding accounting logic in skb_segment_list() was never updated. As a result, the current code unconditionally adds each fragment's truesize to delta_truesize and subtracts it from the parent SKB. Since the fragments are no longer charged to the socket, this subtraction results in an effective under-count of memory when the head is freed. This causes sk_wmem_alloc to remain non-zero, preventing socket destruction and leading to a persistent memory leak. The leak can be observed via KMEMLEAK when tearing down the networking environment: unreferenced object 0xffff8881e6eb9100 (size 2048): comm "ping", pid 6720, jiffies 4295492526 backtrace: kmem_cache_alloc_noprof+0x5c6/0x800 sk_prot_alloc+0x5b/0x220 sk_alloc+0x35/0xa00 inet6_create.part.0+0x303/0x10d0 __sock_create+0x248/0x640 __sys_socket+0x11b/0x1d0 Since skb_segment_list() is exclusively used for SKB_GSO_FRAGLIST packets constructed by GRO, the truesize adjustment is removed. The call to skb_release_head_state() must be preserved. As documented in commit cf673ed0e057 ("net: fix fraglist segmentation reference count leak"), it is still required to correctly drop references to SKB extensions that may be overwritten during __copy_skb_header().
CVE-2025-21768	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net: ipv6: fix dst ref loops in rpl, seg6 and ioam6 lwtunnels Some lwtunnels have a dst cache for post-transformation dst. If the packet destination did not change we may end up recording a reference to the lwtunnel in its own cache, and the lwtunnel state will never be freed. Discovered by the ioam6.sh test, kmemleak was recently fixed to catch per-cpu memory leaks. I'm not sure if rpl and seg6 can actually hit this, but in principle I don't see why not.
CVE-2025-21972	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net: mctp: unshare packets when reassembling Ensure that the frag_list used for reassembly isn't shared with other packets. This avoids incorrect reassembly when packets are cloned, and prevents a memory leak due to circular references between fragments and their skb_shared_info. The upcoming MCTP-over-USB driver uses skb_clone which can trigger the problem - other MCTP drivers don't share SKBs. A kunit test is added to reproduce the issue.
CVE-2026-23021	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net: usb: pegasus: fix memory leak in update_eth_regs_async() When asynchronously writing to the device registers and if usb_submit_urb() fail, the code fail to release allocated to this point resources.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-71154	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net: usb: rtl8150: fix memory leak on usb_submit_urb() failure In async_set_registers(), when usb_submit_urb() fails, the allocated async_req structure and URB are not freed, causing a memory leak. The completion callback async_set_reg_cb() is responsible for freeing these allocations, but it is only called after the URB is successfully submitted and completes (successfully or with error). If submission fails, the callback never runs and the memory is leaked. Fix this by freeing both the URB and the request structure in the error path when usb_submit_urb() fails.
CVE-2026-23172	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net: wwan: t7xx: fix potential skb->frags overflow in RX path When receiving data in the DPMAIF RX path, the t7xx_dpmaif_set_frag_to_skb() function adds page fragments to an skb without checking if the number of fragments has exceeded MAX_SKB_FRAGS. This could lead to a buffer overflow in skb_shinfo(skb)->frags[] array, corrupting adjacent memory and potentially causing kernel crashes or other undefined behavior. This issue was identified through static code analysis by comparing with a similar vulnerability fixed in the mt76 driver commit b102f0c522cf ("mt76: fix array overflow on receiving too many fragments for a packet"). The vulnerability could be triggered if the modem firmware sends packets with excessive fragments. While under normal protocol conditions (MTU 3080 bytes, BAT buffer 3584 bytes), a single packet should not require additional fragments, the kernel should not blindly trust firmware behavior. Malicious, buggy, or compromised firmware could potentially craft packets with more fragments than the kernel expects. Fix this by adding a bounds check before calling skb_add_rx_frag() to ensure nr_frags does not exceed MAX_SKB_FRAGS. The check must be performed before unmapping to avoid a page leak and double DMA unmap during device teardown.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2024-27011	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: netfilter: nf_tables: fix memleak in map from abort path The delete set command does not rely on the transaction object for element removal, therefore, a combination of delete element + delete set from the abort path could result in restoring twice the refcount of the mapping. Check for inactive element in the next generation for the delete element command in the abort path, skip restoring state if next generation bit has been already cleared. This is similar to the activate logic using the set walk iterator. [6170.286929] ------------[cut here]------------ [6170.286939] WARNING: CPU: 6 PID: 790302 at net/netfilter/nf_tables_api.c:2086 nf_tables_chain_destroy+0x1f7/0x220 [nf_tables] [6170.287071] Modules linked in: [...] [6170.287633] CPU: 6 PID: 790302 Comm: kworker/6:2 Not tainted 6.9.0-rc3+ #365 [6170.287768] RIP: 0010:nf_tables_chain_destroy+0x1f7/0x220 [nf_tables] [6170.287886] Code: df 48 8d 7d 58 e8 69 2e 3b df 48 8b 7d 58 e8 80 1b 37 df 48 8d 7d 68 e8 57 2e 3b df 48 8b 7d 68 e8 6e 1b 37 df 48 89 ef eb c4 <0f> 0b 48 83 c4 08 5b 5d 41 5c 41 5d 41 5e 41 5f c3 cc cc cc cc 0f [6170.287895] RSP: 0018:ffff888134b8fd08 EFLAGS: 00010202 [6170.287904] RAX: 0000000000000001 RBX: ffff888125bffb28 RCX: dffffc0000000000 [6170.287912] RDX: 0000000000000003 RSI: ffffffffa20298ab RDI: ffff88811ebe4750 [6170.287919] RBP: ffff88811ebe4700 R08: ffff88838e812650 R09: fffffbfff0623a55 [6170.287926] R10: ffffffff8311d2af R11: 0000000000000001 R12: ffff888125bffb10 [6170.287933] R13: ffff888125bffb10 R14: dead000000000122 R15: dead000000000100 [6170.287940] FS: 0000000000000000(0000) GS:ffff888390b00000(0000) knlGS:0000000000000000 [6170.287948] CS: 0010 DS: 0000 ES: 0000 CR0: 0000000080050033 [6170.287955] CR2: 00007fd31fc00710 CR3: 0000000133f60004 CR4: 00000000001706f0 [6170.287962] Call Trace: [6170.287967] [6170.287973] ? __warn+0x9f/0x1a0 [6170.287986] ? nf_tables_chain_destroy+0x1f7/0x220 [nf_tables] [6170.288092] ? report_bug+0x1b1/0x1e0 [6170.287986] ? nf_tables_chain_destroy+0x1f7/0x220 [nf_tables] [6170.288092] ? report_bug+0x1b1/0x1e0 [6170.288104] ? handle_bug+0x3c/0x70 [6170.288112] ? exc_invalid_op+0x17/0x40 [6170.288120] ? asm_exc_invalid_op+0x1a/0x20 [6170.288132] ? nf_tables_chain_destroy+0x2b/0x220 [nf_tables] [6170.288243] ? nf_tables_chain_destroy+0x1f7/0x220 [nf_tables] [6170.288366] ? nf_tables_chain_destroy+0x2b/0x220 [nf_tables] [6170.288483] nf_tables_trans_destroy_work+0x588/0x590 [nf_tables]

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2024-27012	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: netfilter: nf_tables: restore set elements when delete set fails From abort path, nft_mapelem_activate() needs to restore refcounters to the original state. Currently, it uses the set->ops->walk() to iterate over these set elements. The existing set iterator skips inactive elements in the next generation, this does not work from the abort path to restore the original state since it has to skip active elements instead (not inactive ones). This patch moves the check for inactive elements to the set iterator callback, then it reverses the logic for the .activate case which needs to skip active elements. Toggle next generation bit for elements when delete set command is invoked and call nft_clear() from .activate (abort) path to restore the next generation bit. The splat below shows an object in mappings memleak: [43929.457523] ------------ [cut here]------------ [43929.457532] WARNING: CPU: 0 PID: 1139 at include/net/netfilter/nf_tables.h:1237 nft_setelem_data_deactivate+0xe4/0xf0 [nf_tables] [...] [43929.458014] RIP: 0010:nft_setelem_data_deactivate+0xe4/0xf0 [nf_tables] [43929.458076] Code: 83 f8 01 77 ab 49 8d 7c 24 08 e8 37 5e d0 de 49 8b 6c 24 08 48 8d 7d 50 e8 e9 5c d0 de 8b 45 50 8d 50 ff 89 55 50 85 c0 75 86 <0f> 0b eb 82 0f 0b eb b3 0f 1f 40 00 90 90 90 90 90 90 90 90 90 90 [43929.458081] RSP: 0018:ffff888140f9f4b0 EFLAGS: 00010246 [43929.458086] RAX: 0000000000000000 RBX: ffff8881434f5288 RCX: dffffc0000000000 [43929.458090] RDX: 00000000ffffffff RSI: ffffffffa26d28a7 RDI: ffff88810ecc9550 [43929.458093] RBP: ffff88810ecc9500 R08: 0000000000000001 R09: ffffed10281f3e8f [43929.458096] R10: 0000000000000003 R11: ffff0000ffff0000 R12: ffff8881434f52a0 [43929.458100] R13: ffff888140f9f5f4 R14: ffff888151c7a800 R15: 0000000000000002 [43929.458103] FS: 00007f0c687c4740(0000) GS:ffff888390800000(0000) knlGS:0000000000000000 [43929.458107] CS: 0010 DS: 0000 ES: 0000 CR0: 0000000080050033 [43929.458111] CR2: 00007f58dbe5b008 CR3: 0000000123602005 CR4: 00000000001706f0 [43929.458114] Call Trace: [43929.458118] [43929.458121] ? __warn+0x9f/0x1a0 [43929.458127] ? nft_setelem_data_deactivate+0xe4/0xf0 [nf_tables] [43929.458188] ? report_bug+0x1b1/0x1e0 [43929.458196] ? handle_bug+0x3c/0x70 [43929.458200] ? exc_invalid_op+0x17/0x40 [43929.458211] ? nft_setelem_data_deactivate+0xd7/0xf0 [nf_tables] [43929.458271] ? nft_setelem_data_deactivate+0xe4/0xf0 [nf_tables] [43929.458332] nft_mapelem_deactivate+0x24/0x30 [nf_tables] [43929.458392] nft_rhash_walk+0xdd/0x180 [nf_tables] [43929.458453] ? __pfx_nft_rhash_walk+0x10/0x10 [nf_tables] [43929.458512] ? rb_insert_color+0x2e/0x280 [43929.458520] nft_map_deactivate+0xdc/0x1e0 [nf_tables] [43929.458582] ? __pfx_nft_map_deactivate+0x10/0x10 [nf_tables] [43929.458642] ? __pfx_nft_mapelem_deactivate+0x10/0x10 [nf_tables] [43929.458701] ? __rcu_read_unlock+0x46/0x70 [43929.458709] nft_delset+0xff/0x110 [nf_tables] [43929.458769] nft_flush_table+0x16f/0x460 [nf_tables] [43929.458830] nf_tables_deltable+0x501/0x580 [nf_tables]

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38465	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: netlink: Fix wraparounds of sk->sk_rmem_alloc. Netlink has this pattern in some places if (atomic_read(&sk->sk_rmem_alloc) > sk->sk_rcvbuf) atomic_add(skb->truesize, &sk->sk_rmem_alloc); , which has the same problem fixed by commit 5a465a0da13e ("udp: Fix multiple wraparounds of sk->sk_rmem_alloc."). For example, if we set INT_MAX to SO_RCVBUFFORCE, the condition is always false as the two operands are of int. Then, a single socket can eat as many skb as possible until OOM happens, and we can see multiple wraparounds of sk->sk_rmem_alloc. Let's fix it by using atomic_add_return() and comparing the two variables as unsigned int. Before: [root@fedora ~]# ss -f netlink Recv-Q Send-Q Local Address:Port Peer Address:Port - 1668710080 0 rtnl:nl_wraparound/293 * After: [root@fedora ~]# ss -f netlink Recv-Q Send-Q Local Address:Port Peer Address:Port 2147483072 0 rtnl:nl_wraparound/290 * ^ `--- INT_MAX - 576

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2026-23150	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: nfc: llcp: Fix memleak in nfc_llcp_send_ui_frame(). syzbot reported various memory leaks related to NFC, struct nfc_llcp_sock, sk_buff, nfc_dev, etc. [0] The leading log hinted that nfc_llcp_send_ui_frame() failed to allocate skb due to sock_error(sk) being -ENXIO. ENXIO is set by nfc_llcp_socket_release() when struct nfc_llcp_local is destroyed by local_cleanup(). The problem is that there is no synchronisation between nfc_llcp_send_ui_frame() and local_cleanup(), and skb could be put into local->tx_queue after it was purged in local_cleanup(): CPU1 CPU2 ---- ---- nfc_llcp_send_ui_frame() local_cleanup() |- do { ' |- pdu = nfc_alloc_send_skb(..., &err) | . | |- nfc_llcp_socket_release(local, false, ENXIO); | |- skb_queue_purge(&local->tx_queue); | | ' | |- skb_queue_tail(&local->tx_queue, pdu); | ... | |- pdu = nfc_alloc_send_skb(..., &err) | ^.__________________________________.' local_cleanup() is called for struct nfc_llcp_local only after nfc_llcp_remove_local() unlinks it from llcp_devices. If we hold local->tx_queue.lock then, we can synchronise the thread and nfc_llcp_send_ui_frame(). Let's do that and check list_empty(&local->list) before queuing skb to local->tx_queue in nfc_llcp_send_ui_frame(). [0]: [56.074943][T6096] llcp: nfc_llcp_send_ui_frame: Could not allocate PDU (error=-6) [64.318868][T5813] kmemleak: 6 new suspected memory leaks (see /sys/kernel/debug/kmemleak) BUG: memory leak unreferenced object 0xffff8881272f6800 (size 1024): comm "syz.0.17", pid 6096, jiffies 4294942766 hex dump (first 32 bytes): 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ..... 27 00 03 40 00 00 00 00 00 00 00 00 00 00 00 00 '..@..... backtrace (crc da58d84d): kmemleak_alloc_recursive include/linux/kmemleak.h:44 [inline] slab_post_alloc_hook mm/slub.c:4979 [inline] slab_alloc_node mm/slub.c:5284 [inline] __do_kmalloc_node mm/slub.c:5645 [inline] __kmalloc_noprof+0x3e3/0x6b0 mm/slub.c:5658 kmalloc_noprof include/linux/slab.h:961 [inline] sk_prot_alloc+0x11a/0x1b0 net/core/sock.c:2239 sk_alloc+0x36/0x360 net/core/sock.c:2295 nfc_llcp_sock_alloc+0x37/0x130 net/nfc/llcp_sock.c:979 llcp_sock_create+0x71/0xd0 net/nfc/llcp_sock.c:1044 nfc_sock_create+0xc9/0xf0 net/nfc/af_nfc.c:31 __sock_create+0x1a9/0x340 net/socket.c:1605 sock_create net/socket.c:1663 [inline] __sys_socket_create net/socket.c:1700 [inline] __sys_socket+0xb9/0x1a0 net/socket.c:1747 __do_sys_socket net/socket.c:1761 [inline] __se_sys_socket net/socket.c:1759 [inline] __x64_sys_socket+0x1b/0x30 net/socket.c:1759 do_syscall_x64 arch/x86/entry/syscall_64.c:63 [inline] do_syscall_64+0xa4/0xfa0 arch/x86/entry/syscall_64.c:94 entry_SYSCALL_64_after_hwframe+0x77/0x7f BUG: memory leak unreferenced object 0xffff88810fbd9800 (size 240): comm "syz.0.17", pid 6096, jiffies 4294942850 hex dump (first 32 bytes): 68 f0 ff 08 81 88 ff ff 68 f0 ff 08 81 88 ff ff h.....h..... 00 00 00 00 00 00 00 00 00 00 68 2f 27 81 88 ff ff .....h/'.... backtrace (crc 6cc652b1): kmemleak_alloc_recursive include/linux/kmemleak.h:44 [inline] slab_post_alloc_hook mm/slub.c:4979 [inline] slab_alloc_node mm/slub.c:5284 [inline] kmem_cache_alloc_node_noprof+0x36f/0x5e0 mm/slub.c:5336 __alloc_skb+0x203/0x240 net/core/skbuff.c:660 alloc_skb include/linux/skbuff.h:1383 [inline] alloc_skb_with_frags+0x69/0x3f0 net/core/sk ---truncated---

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2024-43913	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: nvme: apple: fix device reference counting Drivers must call nvme_uninit_ctrl after a successful nvme_init_ctrl. Split the allocation side out to make the error handling boundary easier to navigate. The apple driver had been doing this wrong, leaking the controller device memory on a tagset failure.
CVE-2026-23160	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: octeon_ep: Fix memory leak in octep_device_setup() In octep_device_setup(), if octep_ctrl_net_init() fails, the function returns directly without unmapping the mapped resources and freeing the allocated configuration memory. Fix this by jumping to the unsupported_dev label, which performs the necessary cleanup. This aligns with the error handling logic of other paths in this function. Compile tested only. Issue found using a prototype static analysis tool and code review.
CVE-2025-39847	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ppp: fix memory leak in pad_compress_skb If alloc_skb() fails in pad_compress_skb(), it returns NULL without releasing the old skb. The caller does: skb = pad_compress_skb(ppp, skb); if (!skb) goto drop; drop: kfree_skb(skb); When pad_compress_skb() returns NULL, the reference to the old skb is lost and kfree_skb(skb) ends up doing nothing, leading to a memory leak. Align pad_compress_skb() semantics with realloc(): only free the old skb if allocation and compression succeed. At the call site, use the new_skb variable so the original skb is not lost when pad_compress_skb() fails.
CVE-2025-38444	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: raid10: cleanup memleak at raid10_make_request If raid10_read_request or raid10_write_request registers a new request and the REQ_NOWAIT flag is set, the code does not free the malloc from the mempool. unreferenced object 0xffff8884802c3200 (size 192): comm "fio", pid 9197, jiffies 4298078271 hex dump (first 32 bytes): 00 00 00 00 00 00 00 00 88 41 02 00 00 00 00 00 .....A..... 08 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ..... backtrace (crc c1a049a2): __kmalloc+0x2bb/0x450 mempool_alloc+0x11b/0x320 raid10_make_request+0x19e/0x650 [raid10] md_handle_request+0x3b3/0x9e0 __submit_bio+0x394/0x560 __submit_bio_noacct+0x145/0x530 submit_bio_noacct_nocheck+0x682/0x830 __blkdev_direct_IO_async+0x4dc/0x6b0 blkdev_read_iter+0x1e5/0x3b0 __io_read+0x230/0x1110 io_read+0x13/0x30 io_issue_sqe+0x134/0x1180 io_submit_sqes+0x48c/0xe90 __do_sys_io_uring_enter+0x574/0x8b0 do_syscall_64+0x5c/0xe0 entry_SYSCALL_64_after_hwframe+0x76/0x7e V4: changing backing tree to see if CKI tests will pass. The patch code has not changed between any versions.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2026-23164	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: rocker: fix memory leak in rocker_world_port_post_fini() In rocker_world_port_pre_init(), rocker_port->wpriv is allocated with kzalloc(wops->port_priv_size, GFP_KERNEL). However, in rocker_world_port_post_fini(), the memory is only freed when wops->port_post_fini callback is set: if (!wops->port_post_fini) return; wops->port_post_fini(rocker_port); kfree(rocker_port->wpriv); Since rocker_ofdpa_ops does not implement port_post_fini callback (it is NULL), the wpriv memory allocated for each port is never freed when ports are removed. This leads to a memory leak of sizeof(struct ofdpa_port) bytes per port on every device removal. Fix this by always calling kfree(rocker_port->wpriv) regardless of whether the port_post_fini callback exists.
CVE-2024-41023	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: sched/deadline: Fix task_struct reference leak During the execution of the following stress test with linux-rt: stress-ng --cyclic 30 --timeout 30 --minimize --quiet kmemleak frequently reported a memory leak concerning the task_struct: unreferenced object 0xffff8881305b8000 (size 16136): comm "stress-ng", pid 614, jiffies 4294883961 (age 286.412s) object hex dump (first 32 bytes): 02 40 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .@.............. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ..... debug hex dump (first 16 bytes): 53 09 00 00 00 00 00 00 00 00 00 00 00 00 00 00 S..... backtrace: [<00000000046b6790>] dup_task_struct+0x30/0x540 [<00000000c5ca0f0b>] copy_process+0x3d9/0x50e0 [<00000000ced59777>] kernel_clone+0xb0/0x770 [<00000000a50befdc>] __do_sys_clone+0xb6/0xf0 [<000000001dbf2008>] do_syscall_64+0x5d/0xf0 [<00000000552900ff>] entry_SYSCALL_64_after_hwframe+0x6e/0x76 The issue occurs in start_dl_timer(), which increments the task_struct reference count and sets a timer. The timer callback, dl_task_timer, is supposed to decrement the reference count upon expiration. However, if enqueue_task_dl() is called before the timer expires and cancels it, the reference count is not decremented, leading to the leak. This patch fixes the reference leak by ensuring the task_struct reference count is properly decremented when the timer is canceled.
CVE-2024-57872	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: scsi: ufs: pltfrm: Dellocate HBA during ufshcd_pltfrm_remove() This will ensure that the scsi host is cleaned up properly using scsi_host_dev_release(). Otherwise, it may lead to memory leaks.
CVE-2026-23087	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: scsi: xen: scsiback: Fix potential memory leak in scsiback_remove() Memory allocated for struct vscsiblk_info in scsiback_probe() is not freed in scsiback_remove() leading to potential memory leaks on remove, as well as in the scsiback_probe() error paths. Fix that by freeing it in scsiback_remove().

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2026-23205	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: smb/client: fix memory leak in smb2_open_file() Reproducer: 1. server: directories are exported read-only 2. client: mount -t cifs //${server_ip}/export /mnt 3. client: dd if=/dev/zero of=/mnt/file bs=512 count=1000 oflag=direct 4. client: umount /mnt 5. client: sleep 1 6. client: modprobe -r cifs The error message is as follows: ========== = BUG cifs_small_rq (Not tainted): Objects remaining on __kmem_cache_shutdown() ---------- -- Object 0x00000000d47521be @offset=14336 ... WARNING: mm/slub.c:1251 at __kmem_cache_shutdown+0x34e/0x440, CPU#0: modprobe/1577 ... Call Trace: kmem_cache_destroy+0x94/0x190 cifs_destroy_request_bufs+0x3e/0x50 [cifs] cleanup_module+0x4e/0x540 [cifs] __se_sys_delete_module+0x278/0x400 __x64_sys_delete_module+0x5f/0x70 x64_sys_call+0x2299/0x2ff0 do_syscall_64+0x89/0x350 entry_SYSCALL_64_after_hwframe+0x76/0x7e ... kmem_cache_destroy cifs_small_rq: Slab cache still has objects when called from cifs_destroy_request_bufs+0x3e/0x50 [cifs] WARNING: mm/slab_common.c:532 at kmem_cache_destroy+0x16b/0x190, CPU#0: modprobe/1577
CVE-2025-39929	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: smb: client: fix smbdirect_recv_io leak in smbd_negotiate() error path During tests of another unrelated patch I was able to trigger this error: Objects remaining on __kmem_cache_shutdown()
CVE-2026-23228	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: smb: server: fix leak of active_num_conn in ksmbd_tcp_new_connection() On kthread_run() failure in ksmbd_tcp_new_connection(), the transport is freed via free_transport(), which does not decrement active_num_conn, leaking this counter. Replace free_transport() with ksmbd_tcp_disconnect().
CVE-2025-38612	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: staging: fbtft: fix potential memory leak in fbtft_framebuffer_alloc() In the error paths after fb_info structure is successfully allocated, the memory allocated in fb_deferred_io_init() for info->pagerefs is not freed. Fix that by adding the cleanup function on the error path.
CVE-2024-56712	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: udmabuf: fix memory leak on last export_udmabuf() error path In export_udmabuf(), if dma_buf_fd() fails because the FD table is full, a dma_buf owning the udmabuf has already been created; but the error handling in udmabuf_create() will tear down the udmabuf without doing anything about the containing dma_buf. This leaves a dma_buf in memory that contains a dangling pointer; though that doesn't seem to lead to anything bad except a memory leak. Fix it by moving the dma_buf_fd() call out of export_udmabuf() so that we can give it different error handling. Note that the shape of this code changed a lot in commit 5e72b2b41a21 ("udmabuf: convert udmabuf driver to use folios"); but the memory leak seems to have existed since the introduction of udmabuf.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-71114	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: via_wdt: fix critical boot hang due to unnamed resource allocation The VIA watchdog driver uses allocate_resource() to reserve a MMIO region for the watchdog control register. However, the allocated resource was not given a name, which causes the kernel resource tree to contain an entry marked as "" under /proc/iomem on x86 platforms. During boot, this unnamed resource can lead to a critical hang because subsequent resource lookups and conflict checks fail to handle the invalid entry properly.
CVE-2025-37743	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: wifi: ath12k: Avoid memory leak while enabling statistics Driver uses monitor destination rings for extended statistics mode and standalone monitor mode. In extended statistics mode, TLVs are parsed from the buffer received from the monitor destination ring and assigned to the ppdu_info structure to update per-packet statistics. In standalone monitor mode, along with per-packet statistics, the packet data (payload) is captured, and the driver updates per MSDU to mac80211. When the AP interface is enabled, only extended statistics mode is activated. As part of enabling monitor rings for collecting statistics, the driver subscribes to HAL_RX_MPDU_START TLV in the filter configuration. This TLV is received from the monitor destination ring, and kzalloc for the mon_mpdu object occurs, which is not freed, leading to a memory leak. The kzalloc for the mon_mpdu object is only required while enabling the standalone monitor interface. This causes a memory leak while enabling extended statistics mode in the driver. Fix this memory leak by removing the kzalloc for the mon_mpdu object in the HAL_RX_MPDU_START TLV handling. Additionally, remove the standalone monitor mode handlings in the HAL_MON_BUF_ADDR and HAL_RX_MSDU_END TLVs. These TLV tags will be handled properly when enabling standalone monitor mode in the future. Tested-on: QCN9274 hw2.0 PCI WLAN.WBE.1.3.1-00173-QCAHKSWPL_SILICONZ-1 Tested-on: WCN7850 hw2.0 PCI WLAN.HMT.1.0.c5-00481-QCAHMTSWPL_V1.0_V2.0_SILICONZ-3
CVE-2025-38199	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: wifi: ath12k: Fix memory leak due to multiple rx_stats allocation rx_stats for each arsta is allocated when adding a station. arsta->rx_stats will be freed when a station is removed. Redundant allocations are occurring when the same station is added multiple times. This causes ath12k_mac_station_add() to be called multiple times, and rx_stats is allocated each time. As a result there is memory leaks. Prevent multiple allocations of rx_stats when ath12k_mac_station_add() is called repeatedly by checking if rx_stats is already allocated before allocating again. Allocate arsta->rx_stats if arsta->rx_stats is NULL respectively. Tested-on: QCN9274 hw2.0 PCI WLAN.WBE.1.3.1-00173-QCAHKSWPL_SILICONZ-1 Tested-on: WCN7850 hw2.0 PCI WLAN.HMT.1.0.c5-00481-QCAHMTSWPL_V1.0_V2.0_SILICONZ-3

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-39845	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: x86/mm/64: define ARCH_PAGE_TABLE_SYNC_MASK and arch_sync_kernel_mappings() Define ARCH_PAGE_TABLE_SYNC_MASK and arch_sync_kernel_mappings() to ensure page tables are properly synchronized when calling p*d_populate_kernel(). For 5-level paging, synchronization is performed via pgd_populate_kernel(). In 4-level paging, pgd_populate() is a no-op, so synchronization is instead performed at the P4D level via p4d_populate_kernel(). This fixes intermittent boot failures on systems using 4-level paging and a large amount of persistent memory: BUG: unable to handle page fault for address: ffffe70000000034 #PF: supervisor write access in kernel mode #PF: error_code(0x0002) - not-present page PGD 0 P4D 0 Oops: 0002 [#1] SMP NOPTI RIP: 0010:__init_single_page+0x9/0x6d Call Trace: __init_zone_device_page+0x17/0x5d memmap_init_zone_device+0x154/0x1bb pagemap_range+0x2e0/0x40f memremap_pages+0x10b/0x2f0 devm_memremap_pages+0x1e/0x60 dev_dax_probe+0xce/0x2ec [device_dax] dax_bus_probe+0x6d/0xc9 [... snip ...] It also fixes a crash in vmemmap_set_pmd() caused by accessing vmemmap before sync_global_pgds() [1]: BUG: unable to handle page fault for address: ffffeb3ff1200000 #PF: supervisor write access in kernel mode #PF: error_code(0x0002) - not-present page PGD 0 P4D 0 Oops: Oops: 0002 [#1] PREEMPT SMP NOPTI Tainted: [W]=WARN RIP: 0010:vmemmap_set_pmd+0xff/0x230 vmemmap_populate_hugepages+0x176/0x180 vmemmap_populate+0x34/0x80 __populate_section_memmap+0x41/0x90 sparse_add_section+0x121/0x3e0 __add_pages+0xba/0x150 add_pages+0x1d/0x70 memremap_pages+0x3dc/0x810 devm_memremap_pages+0x1c/0x60 xe_devm_add+0x8b/0x100 [xe] xe_tile_init_noalloc+0x6a/0x70 [xe] xe_device_probe+0x48c/0x740 [xe] [... snip ...]
CVE-2024-40979	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: wifi: ath12k: fix kernel crash during resume Currently during resume, QMI target memory is not properly handled, resulting in kernel crash in case DMA remap is not supported: BUG: Bad page state in process kworker/u16:54 pfn:36e80 page: refcount:1 mapcount:0 mapping:0000000000000000 index0x0 pfn:0x36e80 page dumped because: nonzero _refcount Call Trace: bad_page free_page_is_bad_report __free_pages_ok __free_pages dma_direct_free dma_free_attrs ath12k_qmi_free_target_mem_chunk ath12k_qmi_msg_mem_request_cb The reason is: Once ath12k module is loaded, firmware sends memory request to host. In case DMA remap not supported, ath12k refuses the first request due to failure in allocating with large segment size: ath12k_pci 0000:04:00.0: qmi firmware request memory request ath12k_pci 0000:04:00.0: qmi mem seg type 1 size 7077888 ath12k_pci 0000:04:00.0: qmi mem seg type 4 size 8454144 ath12k_pci 0000:04:00.0: qmi dma allocation failed (7077888 B type 1), will try later with small size ath12k_pci 0000:04:00.0: qmi delays mem_request 2 ath12k_pci 0000:04:00.0: qmi firmware request memory request Later firmware comes back with more but small segments and allocation succeeds: ath12k_pci 0000:04:00.0: qmi mem seg type 1 size 524288 ath12k_pci 0000:04:00.0: qmi mem seg type 1 size 524288 ath12k_pci 0000:04:00.0: qmi mem seg type 1 size 524288 ath12k_pci 0000:04:00.0: qmi mem seg type 1 size 524288 ath12k_pci 0000:04:00.0: qmi mem seg type 1 size 524288 ath12k_pci 0000:04:00.0: qmi mem seg type 1 size 524288 ath12k_pci 0000:04:00.0: qmi mem seg type 1 size 524288 ath12k_pci 0000:04:00.0: qmi mem seg type 1 size 524288 ath12k_pci 0000:04:00.0:

CVE	Severity	CVSS	CVSS Vector	Component	Description
					qmi mem seg type 1 size 262144 ath12k_pci 0000:04:00.0: qmi mem seg type 1 size 524288 ath12k_pci 0000:04:00.0: qmi mem seg type 1 size 524288 ath12k_pci 0000:04:00.0: qmi mem seg type 1 size 524288 ath12k_pci 0000:04:00.0: qmi mem seg type 1 size 524288 ath12k_pci 0000:04:00.0: qmi mem seg type 1 size 524288 ath12k_pci 0000:04:00.0: qmi mem seg type 4

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-39914	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: tracing: Silence warning when chunk allocation fails in trace_pid_write Syzkaller trigger a fault injection warning: WARNING: CPU: 1 PID: 12326 at tracepoint_add_func+0xbfc/0xeb0 Modules linked in: CPU: 1 UID: 0 PID: 12326 Comm: syz.6.10325 Tainted: G U 6.14.0-rc5-syzkaller #0 Tainted: [U]=USER Hardware name: Google Compute Engine/Google Compute Engine RIP: 0010:tracepoint_add_func+0xbfc/0xeb0 kernel/tracepoint.c:294 Code: 09 fe ff 90 0f 0b 90 0f b6 74 24 43 31 ff 41 bc ea ff ff ff RSP: 0018:ffffc9000414fb48 EFLAGS: 00010283 RAX: 00000000000012a1 RBX: ffffffff8e240ae0 RCX: ffffc90014b78000 RDX: 0000000000080000 RSI: ffffffff81bbd78b RDI: 0000000000000001 RBP: 0000000000000000 R08: 0000000000000001 R09: 0000000000000000 R10: 0000000000000001 R11: 0000000000000001 R12: ffffffffffffffef R13: 0000000000000000 R14: dffffc0000000000 R15: ffffffff81c264f0 FS: 00007f27217f66c0(0000) GS:ffff8880b8700000(0000) knlGS:0000000000000000 CS: 0010 DS: 0000 ES: 0000 CR0: 0000000080050033 CR2: 0000001b2e80dff8 CR3: 00000000268f8000 CR4: 00000000003526f0 DR0: 0000000000000000 DR1: 0000000000000000 DR2: 0000000000000000 DR3: 0000000000000000 DR6: 00000000fffe0ff0 DR7: 0000000000000400 Call Trace: tracepoint_probe_register_prio+0xc0/0x110 kernel/tracepoint.c:464 register_trace_prio_sched_switch include/trace/events/sched.h:222 [inline] register_pid_events kernel/trace/trace_events.c:2354 [inline] event_pid_write.isra.0+0x439/0x7a0 kernel/trace/trace_events.c:2425 vfs_write+0x24c/0x1150 fs/read_write.c:677 ksys_write+0x12b/0x250 fs/read_write.c:731 do_syscall_x64 arch/x86/entry/common.c:52 [inline] do_syscall_64+0xcd/0x250 arch/x86/entry/common.c:83 entry_SYSCALL_64_after_hwframe+0x77/0x7f We can reproduce the warning by following the steps below: 1. echo 8 >> set_event_notrace_pid. Let tr->filtered_pids owns one pid and register sched_switch tracepoint. 2. echo '' >> set_event_pid, and perform fault injection during chunk allocation of trace_pid_list_alloc. Let pid_list with no pid and assign to tr->filtered_pids. 3. echo '' >> set_event_pid. Let pid_list is NULL and assign to tr->filtered_pids. 4. echo 9 >> set_event_pid, will trigger the double register sched_switch tracepoint warning. The reason is that syzkaller injects a fault into the chunk allocation in trace_pid_list_alloc, causing a failure in trace_pid_list_set, which may trigger double register of the same tracepoint. This only occurs when the system is about to crash, but to suppress this warning, let's add failure handling logic to trace_pid_list_set.
CVE-2024-23848	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel through 6.7.1, there is a use-after-free in cec_queue_msg_fh, related to drivers/media/cec/core/cec-adap.c and drivers/media/cec/core/cec-api.c.
CVE-2025-38386	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ACPICA: Refuse to evaluate a method if arguments are missing As reported in [1], a platform firmware update that increased the number of method parameters and forgot to update a least one of its callers, caused ACPICA to crash due to use-after-free. Since this a result of a clear AML issue that arguably cannot be fixed up by the interpreter (it cannot produce missing data out of thin air), address it by making ACPICA refuse to evaluate a method if the caller attempts to pass fewer arguments than expected to it.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2024-57875	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: block: RCU protect disk->conv_zones_bitmap Ensure that a disk revalidation changing the conventional zones bitmap of a disk does not cause invalid memory references when using the disk_zone_is_conv() helper by RCU protecting the disk->conv_zones_bitmap pointer. disk_zone_is_conv() is modified to operate under the RCU read lock and the function disk_set_conv_zones_bitmap() is added to update a disk conv_zones_bitmap pointer using rcu_replace_pointer() with the disk zone_wplugs_lock spinlock held. disk_free_zone_resources() is modified to call disk_update_zone_resources() with a NULL bitmap pointer to free the disk conv_zones_bitmap. disk_set_conv_zones_bitmap() is also used in disk_update_zone_resources() to set the new (revalidated) bitmap and free the old one.
CVE-2024-42075	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: bpf: Fix remap of arena. The bpf arena logic didn't account for mremap operation. Add a refcnt for multiple mmap events to prevent use-after-free in arena_vm_close.
CVE-2025-39721	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: crypto: qat - flush misc workqueue during device shutdown Repeated loading and unloading of a device specific QAT driver, for example qat_4xxx, in a tight loop can lead to a crash due to a use-after-free scenario. This occurs when a power management (PM) interrupt triggers just before the device-specific driver (e.g., qat_4xxx.ko) is unloaded, while the core driver (intel_qat.ko) remains loaded. Since the driver uses a shared workqueue (`qat_misc_wq`) across all devices and owned by intel_qat.ko, a deferred routine from the device-specific driver may still be pending in the queue. If this routine executes after the driver is unloaded, it can dereference freed memory, resulting in a page fault and kernel crash like the following: BUG: unable to handle page fault for address: ffa000002e50a01c #PF: supervisor read access in kernel mode RIP: 0010:pm_bh_handler+0x1d2/0x250 [intel_qat] Call Trace: pm_bh_handler+0x1d2/0x250 [intel_qat] process_one_work+0x171/0x340 worker_thread+0x277/0x3a0 kthread+0xf0/0x120 ret_from_fork+0x2d/0x50 To prevent this, flush the misc workqueue during device shutdown to ensure that all pending work items are completed before the driver is unloaded. Note: This approach may slightly increase shutdown latency if the workqueue contains jobs from other devices, but it ensures correctness and stability.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38449	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: drm/gem: Acquire references on GEM handles for framebuffers A GEM handle can be released while the GEM buffer object is attached to a DRM framebuffer. This leads to the release of the dma-buf backing the buffer object, if any. [1] Trying to use the framebuffer in further mode-setting operations leads to a segmentation fault. Most easily happens with driver that use shadow planes for vmap-ing the dma-buf during a page flip. An example is shown below. [156.791968] ------------[cut here]------------ [156.796830] WARNING: CPU: 2 PID: 2255 at drivers/dma-buf/dma-buf.c:1527 dma_buf_vmap+0x224/0x430 [...] [156.942028] RIP: 0010:dma_buf_vmap+0x224/0x430 [157.043420] Call Trace: [157.045898] [157.048030] ? show_trace_log_lvl+0x1af/0x2c0 [157.052436] ? show_trace_log_lvl+0x1af/0x2c0 [157.056836] ? show_trace_log_lvl+0x1af/0x2c0 [157.061253] ? drm_gem_shmem_vmap+0x74/0x710 [157.065567] ? dma_buf_vmap+0x224/0x430 [157.069446] ? __warn.cold+0x58/0xe4 [157.073061] ? dma_buf_vmap+0x224/0x430 [157.077111] ? report_bug+0x1dd/0x390 [157.080842] ? handle_bug+0x5e/0xa0 [157.084389] ? exc_invalid_op+0x14/0x50 [157.088291] ? asm_exc_invalid_op+0x16/0x20 [157.092548] ? dma_buf_vmap+0x224/0x430 [157.096663] ? dma_resv_get_singleton+0x6d/0x230 [157.101341] ? __pfx_dma_buf_vmap+0x10/0x10 [157.105588] ? __pfx_dma_resv_get_singleton+0x10/0x10 [157.110697] drm_gem_shmem_vmap+0x74/0x710 [157.114866] drm_gem_vmap+0xa9/0x1b0 [157.118763] drm_gem_vmap_unlocked+0x46/0xa0 [157.123086] drm_gem_fb_vmap+0xab/0x300 [157.126979] drm_atomic_helper_prepare_planes.part.0+0x487/0xb10 [157.133032] ? lockdep_init_map_type+0x19d/0x880 [157.137701] drm_atomic_helper_commit+0x13d/0x2e0 [157.142671] ? drm_atomic_nonblocking_commit+0xa0/0x180 [157.147988] drm_mode_atomic_ioctl+0x766/0xe40 [...] [157.346424] ---[end trace 0000000000000000]--- Acquiring GEM handles for the framebuffer's GEM buffer objects prevents this from happening. The framebuffer's cleanup later puts the handle references. Commit 1a148af06000 ("drm/gem-shmem: Use dma_buf from GEM object instance") triggers the segmentation fault easily by using the dma-buf field more widely. The underlying issue with reference counting has been present before. v2: - acquire the handle instead of the BO (Christian) - fix comment style (Christian) - drop the Fixes tag (Christian) - rename err_ gotos - add missing Link tag
CVE-2025-38578	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: f2fs: fix to avoid UAF in f2fs_sync_inode_meta() syzbot reported an UAF issue as below: [1] [2] [1] https://syzkaller.appspot.com/text?tag=CrashReport&x=16594c60580000 == BUG: KASAN: use-after-free in __list_del_entry_valid+0xa6/0x130 lib/list_debug.c:62 Read of size 8 at addr ffff888100567dc8 by task kworker/u4:0/8 CPU: 1 PID: 8 Comm: kworker/u4:0 Tainted: G W 6.1.129-syzkaller-00017-g642656a36791 #0 Hardware name: Google Google Compute Engine/Google Compute Engine, BIOS Google 02/12/2025 Workqueue: writeback wb_workfn (flush-7:0) Call Trace: __dump_stack lib/dump_stack.c:88 [inline]

CVE	Severity	CVSS	CVSS Vector	Component	Description
					dump_stack_lvl+0x151/0x1b7 lib/dump_stack.c:106 print_address_description mm/kasan/report.c:316 [inline] print_report+0x158/0x4e0 mm/kasan/report.c:427 kasan_report+0x13c/0x170 mm/kasan/report.c:531 __asan_report_load8_noabort+0x14/0x20 mm/kasan/report_generic.c:351 __list_del_entry_valid+0xa6/0x130 lib/list_debug.c:62 __list_del_entry include/linux/list.h:134 [inline] list_del_init include/linux/list.h:206 [inline] f2fs_inode_synced+0x100/0x2e0 fs/f2fs/super.c:1553 f2fs_update_inode+0x72/0x1c40 fs/f2fs/inode.c:588 f2fs_update_inode_page+0x135/0x170 fs/f2fs/inode.c:706 f2fs_write_inode+0x416/0x790 fs/f2fs/inode.c:734 write_inode fs/fs-writeback.c:1460 [inline] __writeback_single_inode+0x4cf/0xb80 fs/fs-writeback.c:1677 writeback_sb_inodes+0xb32/0x1910 fs/fs-writeback.c:1903 __writeback_inodes_wb+0x118/0x3f0 fs/fs-writeback.c:1974 wb_writeback+0x3da/0xa00 fs/fs-writeback.c:2081 wb_check_background_flush fs/fs-writeback.c:2151 [inline] wb_do_writeback fs/fs-writeback.c:2239 [inline] wb_workfn+0xbba/0x1030 fs/fs-writeback.c:2266 process_one_work+0x73d/0xcb0 kernel/workqueue.c:2299 worker_thread+0xa60/0x1260 kernel/workqueue.c:2446 kthread+0x26d/0x300 kernel/kthread.c:386 ret_from_fork+0x1f/0x30 arch/x86/entry/entry_64.S:295 Allocated by task 298: kasan_save_stack mm/kasan/common.c:45 [inline] kasan_set_track+0x4b/0x70 mm/kasan/common.c:52 kasan_save_alloc_info+0x1f/0x30 mm/kasan/generic.c:505 __kasan_slab_alloc+0x6c/0x80 mm/kasan/common.c:333 kasan_slab_alloc include/linux/kasan.h:202 [inline] slab_post_alloc_hook+0x53/0x2c0 mm/slab.h:768 slab_alloc_node mm/slub.c:3421 [inline] slab_alloc mm/slub.c:3431 [inline] __kmem_cache_alloc_lru mm/slub.c:3438 [inline] kmem_cache_alloc_lru+0x102/0x270 mm/slub.c:3454 alloc_inode_sb include/linux/fs.h:3255 [inline] f2fs_alloc_inode+0x2d/0x350 fs/f2fs/super.c:1437 alloc_inode fs/inode.c:261 [inline] iget_locked+0x18c/0x7e0 fs/inode.c:1373 f2fs_iget+0x55/0x4ca0 fs/f2fs/inode.c:486 f2fs_lookup+0x3c1/0xb50 fs/f2fs/namei.c:484 __lookup_slow+0x2b9/0x3e0 fs/namei.c:1689 lookup_slow+0x5a/0x80 fs/namei.c:1706 walk_component+0x2e7/0x410 fs/namei.c:1997 lookup_last fs/namei.c:2454 [inline] path_lookupat+0x16d/0x450 fs/namei.c:2478 filename_lookup+0x251/0x600 fs/namei.c:2507 vfs_statx+0x107/0x4b0 fs/stat.c:229 vfs_fstatat fs/stat.c:267 [inline] vfs_lstat include/linux/fs.h:3434 [inline] __do_sys_newlstat fs/stat.c:423 [inline] __se_sys_newlstat+0xda/0x7c0 fs/stat.c:417 __x64_sys_newlstat+0x5b/0x70 fs/stat.c:417 x64_sys_call+0x52/0x9a0 arch/x86/include/generated/asm/syscalls_64.h:7 do_syscall_x64 arch/x86/entry/common.c:51 [inline] do_syscall_64+0x3b/0x80 arch/x86/entry/common.c:81 entry_SYSCALL_64_after_hwframe+0x68/0xd2 Freed by task 0: kasan_save_stack mm/kasan/common.c:45 [inline] kasan_set_track+0x4b/0x70 mm/kasan/common.c:52 kasan_save_free_info+0x2b/0x40 mm/kasan/generic.c:516 ____kasan_slab_free+0x131/0x180 mm/kasan/common.c:241 __kasan_slab_free+0x11/0x20

CVE	Severity	CVSS	CVSS Vector	Component	Description
					mm/kasan/common.c:249 kasan_slab_free include/linux/kasan.h:178 [inline] slab_free_hook mm/slub.c:1745 [inline] slab_free_freelist_hook mm/slub.c:1771 [inline] slab_free mm/slub.c:3686 [inline] kmem_cache_free+0x--- truncated---
CVE-2025-38577	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: f2fs: fix to avoid panic in f2fs_evict_inode As syzbot [1] reported as below: R10: 0000000000000100 R11: 0000000000000206 R12: 00007ffe17473450 R13: 00007f28b1c10854 R14: 000000000000dae5 R15: 00007ffe17474520 ---[end trace 0000000000000000]--- == BUG: KASAN: use-after-free in __list_del_entry_valid+0xa6/0x130 lib/list_debug.c:62 Read of size 8 at addr ffff88812d962278 by task syz-executor/564 CPU: 1 PID: 564 Comm: syz-executor Tainted: G W 6.1.129-syzkaller #0 Hardware name: Google Google Compute Engine/Google Compute Engine, BIOS Google 02/12/2025 Call Trace: __dump_stack+0x21/0x24 lib/dump_stack.c:88 dump_stack_lvl+0xee/0x158 lib/dump_stack.c:106 print_address_description+0x71/0x210 mm/kasan/report.c:316 print_report+0x4a/0x60 mm/kasan/report.c:427 kasan_report+0x122/0x150 mm/kasan/report.c:531 __asan_report_load8_noabort+0x14/0x20 mm/kasan/report_generic.c:351 __list_del_entry_valid+0xa6/0x130 lib/list_debug.c:62 __list_del_entry include/linux/list.h:134 [inline] list_del_init include/linux/list.h:206 [inline] f2fs_inode_synced+0xf7/0x2e0 fs/f2fs/super.c:1531 f2fs_update_inode+0x74/0x1c40 fs/f2fs/inode.c:585 f2fs_update_inode_page+0x137/0x170 fs/f2fs/inode.c:703 f2fs_write_inode+0x4ec/0x770 fs/f2fs/inode.c:731 write_inode fs/fs-writeback.c:1460 [inline] __writeback_single_inode+0x4a0/0xab0 fs/fs-writeback.c:1677 writeback_single_inode+0x221/0x8b0 fs/fs-writeback.c:1733 sync_inode_metadata+0xb6/0x110 fs/fs-writeback.c:2789 f2fs_sync_inode_meta+0x16d/0x2a0 fs/f2fs/checkpoint.c:1159 block_operations fs/f2fs/checkpoint.c:1269 [inline] f2fs_write_checkpoint+0xca3/0x2100 fs/f2fs/checkpoint.c:1658 kill_f2fs_super+0x231/0x390 fs/f2fs/super.c:4668 deactivate_locked_super+0x98/0x100 fs/super.c:332 deactivate_super+0xaf/0xe0 fs/super.c:363 cleanup_mnt+0x45f/0x4e0 fs/namespace.c:1186 __cleanup_mnt+0x19/0x20 fs/namespace.c:1193 task_work_run+0x1c6/0x230 kernel/task_work.c:203 exit_task_work include/linux/task_work.h:39 [inline] do_exit+0x9fb/0x2410 kernel/exit.c:871 do_group_exit+0x210/0x2d0 kernel/exit.c:1021 __do_sys_exit_group kernel/exit.c:1032 [inline] __se_sys_exit_group kernel/exit.c:1030 [inline] __x64_sys_exit_group+0x3f/0x40 kernel/exit.c:1030 x64_sys_call+0x7b4/0x9a0 arch/x86/include/generated/asm/syscalls_64.h:232 do_syscall_x64 arch/x86/entry/common.c:51 [inline] do_syscall_64+0x4c/0xa0 arch/x86/entry/common.c:81 entry_SYSCALL_64_after_hwframe+0x68/0xd2 RIP: 0033:0x7f28b1b8e169 Code: Unable to access opcode bytes at 0x7f28b1b8e13f. RSP: 002b:00007ffe174710a8 EFLAGS: 00000246 ORIG_RAX: 00000000000000e7 RAX: ffffffffffffffda RBX: 00007f28b1c10879 RCX: 00007f28b1b8e169 RDX: 0000000000000000 RSI: 0000000000000000 RDI: 0000000000000001 RBP: 0000000000000002

CVE	Severity	CVSS	CVSS Vector	Component	Description
					R08: 00007ffe1746ee47 R09: 00007ffe17472360 R10: 0000000000000009 R11: 0000000000000246 R12: 00007ffe17472360 R13: 00007f28b1c10854 R14: 000000000000dae5 R15: 00007ffe17474520 Allocated by task 569: kasan_save_stack mm/kasan/common.c:45 [inline] kasan_set_track+0x4b/0x70 mm/kasan/common.c:52 kasan_save_alloc_info+0x25/0x30 mm/kasan/generic.c:505 __kasan_slab_alloc+0x72/0x80 mm/kasan/common.c:328 kasan_slab_alloc include/linux/kasan.h:201 [inline] slab_post_alloc_hook+0x4f/0x2c0 mm/slab.h:737 slab_alloc_node mm/slub.c:3398 [inline] slab_alloc mm/slub.c:3406 [inline] __kmem_cache_alloc_lru mm/slub.c:3413 [inline] kmem_cache_alloc_lru+0x104/0x220 mm/slub.c:3429 alloc_inode_sb include/linux/fs.h:3245 [inline] f2fs_alloc_inode+0x2d/0x340 fs/f2fs/super.c:1419 alloc_inode fs/inode.c:261 [inline] iget_locked+0x186/0x880 fs/inode.c:1373 f2fs_iget+0x55/0x4c60 fs/f2fs/inode.c:483 f2fs_lookup+0x366/0xab0 fs/f2fs/namei.c:487 __lookup_slow+0x2a3/0x3d0 fs/namei.c:1690 lookup_slow+0x57/0x70 fs/namei.c:1707 walk_component+0x2e6/0x410 fs/namei ---truncated---
CVE-2025-21861	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: mm/migrate_device: don't add folio to be freed to LRU in migrate_device_finalize() If migration succeeded, we called folio_migrate_flags()->mem_cgroup_migrate() to migrate the memcg from the old to the new folio. This will set memcg_data of the old folio to 0. Similarly, if migration failed, memcg_data of the dst folio is left unset. If we call folio_putback_lru() on such folios (memcg_data == 0), we will add the folio to be freed to the LRU, making memcg code unhappy. Running the hmm selftests: # ./hmm-tests ... # RUN hmm.hmm_device_private.migrate ... [102.078007][T14893] page: refcount:1 mapcount:0 mapping:0000000000000000 index:0x7ff27d200 pfn:0x13cc00 [102.079974][T14893] anon flags: 0x17ff00000020018(uptodate|dirty|swapbacked|node=0|zone=2|lastcpupid=0x7ff) [102.082037][T14893] raw: 017ff00000020018 dead000000000100 dead000000000122 ffff8881353896c9 [102.083687][T14893] raw: 00000007ff27d200 0000000000000000 00000001ffffffff 0000000000000000 [102.085331][T14893] page dumped because: VM_WARN_ON_ONCE_FOLIO(!memcg && !mem_cgroup_disabled()) [102.087230][T14893] ------------[cut here]------------ [102.088279][T14893] WARNING: CPU: 0 PID: 14893 at ./include/linux/memcontrol.h:726 folio_lruvec_lock_irqsave+0x10e/0x170 [102.090478][T14893] Modules linked in: [102.091244][T14893] CPU: 0 UID: 0 PID: 14893 Comm: hmm-tests Not tainted 6.13.0-09623-g6c216bc522fd #151 [102.093089][T14893] Hardware name: QEMU Standard PC (Q35 + ICH9, 2009), BIOS 1.16.3-2.fc40 04/01/2014 [102.094848][T14893] RIP: 0010:folio_lruvec_lock_irqsave+0x10e/0x170 [102.096104][T14893] Code: ... [102.099908][T14893] RSP: 0018:ffffc900236c37b0 EFLAGS: 00010293 [102.101152][T14893] RAX: 0000000000000000 RBX: ffffea0004f30000 RCX: ffffffff8183f426 [102.102684][T14893] RDX: ffff8881063cb880 RSI: ffffffff81b8117f RDI: ffff8881063cb880 [102.104227][T14893] RBP: 0000000000000000 R08: 0000000000000005 R09: 0000000000000000 [102.105757][T14893] R10: 0000000000000001 R11: 0000000000000002 R12: ffffc900236c37d8 [102.107296][T14893] R13: ffff888277a2bcb0 R14: 000000000000001f R15: 0000000000000000 [102.108830][T14893] FS:

CVE	Severity	CVSS	CVSS Vector	Component	Description
					00007ff27dbdd740(0000) GS:ffff888277a00000(0000) knlGS:0000000000000000 [102.110643] [T14893] CS: 0010 DS: 0000 ES: 0000 CR0: 0000000080050033 [102.111924][T14893] CR2: 00007ff27d400000 CR3: 000000010866e000 CR4: 0000000000750ef0 [102.113478][T14893] PKRU: 55555554 [102.114172][T14893] Call Trace: [102.114805][T14893] [102.115397] [T14893] ? folio_lruvec_lock_irqsave+0x10e/0x170 [102.116547][T14893] ? __warn.cold+0x110/0x210 [102.117461][T14893] ? folio_lruvec_lock_irqsave+0x10e/0x170 [102.118667][T14893] ? report_bug+0x1b9/0x320 [102.119571][T14893] ? handle_bug+0x54/0x90 [102.120494][T14893] ? exc_invalid_op+0x17/0x50 [102.121433][T14893] ? asm_exc_invalid_op+0x1a/0x20 [102.122435] [T14893] ? __wake_up_klogd.part.0+0x76/0xd0 [102.123506][T14893] ? dump_page+0x4f/0x60 [102.124352][T14893] ? folio_lruvec_lock_irqsave+0x10e/0x170 [102.125500][T14893] folio_batch_move_lru+0xd4/0x200 [102.126577] [T14893] ? __pfx_lru_add+0x10/0x10 [102.127505][T14893] __folio_batch_add_and_move+0x391/0x720 [102.128633][T14893] ? __pfx_lru_add+0x10/0x10 [102.129550][T14893] folio_putback_lru+0x16/0x80 [102.130564] [T14893] migrate_device_finalize+0x9b/0x530 [102.131640][T14893] dmirror_migrate_to_device.constprop.0+0x7c5/0xad0 [102.133047][T14893] dmirror_fops_unlocked_ioctl+0x89b/0xc80 Likely, nothing else goes wrong: putting the last folio reference will remove the folio from the LRU again. So besides memcg complaining, adding the folio to be freed to the LRU is just an unnecessary step. The new flow resembles what we have in migrate_folio_move(): add the dst to the lru, rem ---truncated---
CVE-2024-50027	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: thermal: core: Free tzp copy along with the thermal zone The object pointed to by tz->tzp may still be accessed after being freed in thermal_zone_device_unregister(), so move the freeing of it to the point after the removal completion has been completed at which it cannot be accessed any more.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2024-57976	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: btrfs: do proper folio cleanup when cow_file_range() failed [BUG] When testing with COW fixup marked as BUG_ON() (this is involved with the new pin_user_pages*() change, which should not result new out-of-band dirty pages), I hit a crash triggered by the BUG_ON() from hitting COW fixup path. This BUG_ON() happens just after a failed btrfs_run_delalloc_range(): BTRFS error (device dm-2): failed to run delalloc range, root 348 ino 405 folio 65536 submit_bitmap 6-15 start 90112 len 106496: -28 ------------[cut here]------------ kernel BUG at fs/btrfs/extent_io.c:1444! Internal error: Oops - BUG: 00000000f2000800 [#1] SMP CPU: 0 UID: 0 PID: 434621 Comm: kworker/u24:8 Tainted: G OE 6.12.0-rc7-custom+ #86 Hardware name: QEMU KVM Virtual Machine, BIOS unknown 2/2/2022 Workqueue: events_unbound btrfs_async_reclaim_data_space [btrfs] pc : extent_writepage_io+0x2d4/0x308 [btrfs] lr : extent_writepage_io+0x2d4/0x308 [btrfs] Call trace: extent_writepage_io+0x2d4/0x308 [btrfs] extent_writepage+0x218/0x330 [btrfs] extent_write_cache_pages+0x1d4/0x4b0 [btrfs] btrfs_writepages+0x94/0x150 [btrfs] do_writepages+0x74/0x190 filemap_fdatawrite_wbc+0x88/0xc8 start_delalloc_inodes+0x180/0x3b0 [btrfs] btrfs_start_delalloc_roots+0x174/0x280 [btrfs] shrink_delalloc+0x114/0x280 [btrfs] flush_space+0x250/0x2f8 [btrfs] btrfs_async_reclaim_data_space+0x180/0x228 [btrfs] process_one_work+0x164/0x408 worker_thread+0x25c/0x388 kthread+0x100/0x118 ret_from_fork+0x10/0x20 Code: aa1403e1 9402f3ef aa1403e0 9402f36f (d4210000) ---[end trace 0000000000000000]--- [CAUSE] That failure is mostly from cow_file_range(), where we can hit -ENOSPC. Although the -ENOSPC is already a bug related to our space reservation code, let's just focus on the error handling. For example, we have the following dirty range [0, 64K) of an inode, with 4K sector size and 4K page size: 0 16K 32K 48K 64K |////////////////////////////////////| |####################################### #| Where |///| means page are still dirty, and |###| means the extent io tree has EXTENT_DELALLOC flag. - Enter extent_writepage() for page 0 - Enter btrfs_run_delalloc_range() for range [0, 64K) - Enter cow_file_range() for range [0, 64K) - Function btrfs_reserve_extent() only reserved one 16K extent So we created extent map and ordered extent for range [0, 16K) 0 16K 32K 48K 64K |///////|////////////////////////////////| |<- OE ->|################################| And range [0, 16K) has its delalloc flag cleared. But since we haven't yet submit any bio, involved 4 pages are still dirty. - Function btrfs_reserve_extent() returns with -ENOSPC Now we have to run error cleanup, which will clear all EXTENT_DELALLOC* flags and clear the dirty flags for the remaining ranges: 0 16K 32K 48K 64K |////////| | | | Note that range [0, 16K) still has its pages dirty. - Some time later, writeback is triggered again for the range [0, 16K) since the page range still has dirty flags. - btrfs_run_delalloc_range() will do nothing because there is no EXTENT_DELALLOC flag. - extent_writepage_io() finds page 0 has no ordered flag Which falls into the COW fixup path, triggering the BUG_ON(). Unfortunately this error handling bug dates back to the introduction of btrfs. Thankfully with the abuse of COW fixup, at least it won't crash the kernel. [FIX] Instead of immediately unlocking the extent and folios, we keep the extent and folios locked until either erroring out or the whole delalloc range finished. When the whole delalloc range finished without error, we just unlock the whole range with PAGE_SET_ORDERED (and PAGE_UNLOCK for

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2024-57975	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	!keep_locked cases) ---truncated--- In the Linux kernel, the following vulnerability has been resolved: btrfs: do proper folio cleanup when run_delalloc_nocow() failed [BUG] With CONFIG_DEBUG_VM set, test case generic/476 has some chance to crash with the following VM_BUG_ON_FOLIO(): BTRFS error (device dm-3): cow_file_range failed, start 1146880 end 1253375 len 106496 ret -28 BTRFS error (device dm-3): run_delalloc_nocow failed, start 1146880 end 1253375 len 106496 ret -28 page: refcount:4 mapcount:0 mapping:00000000592787cc index:0x12 pfn:0x10664 aops:btrfs_aops [btrfs] ino:101 dentry name(?):"f1774" flags: 0x2fffff80004028(uptodate|lru|private|node=0|zone=2|lastcpupid=0xfffff) page dumped because: VM_BUG_ON_FOLIO(!folio_test_locked(folio)) ------------[cut here]------------ kernel BUG at mm/page-writeback.c:2992! Internal error: Oops - BUG: 00000000f2000800 [#1] SMP CPU: 2 UID: 0 PID: 3943513 Comm: kworker/u24:15 Tainted: G OE 6.12.0-rc7-custom+ #87 Tainted: [O]=OOT_MODULE, [E]=UNSIGNED_MODULE Hardware name: QEMU KVM Virtual Machine, BIOS unknown 2/2/2022 Workqueue: events_unbound btrfs_async_reclaim_data_space [btrfs] pc : folio_clear_dirty_for_io+0x128/0x258 lr : folio_clear_dirty_for_io+0x128/0x258 Call trace: folio_clear_dirty_for_io+0x128/0x258 btrfs_folio_clamp_clear_dirty+0x80/0xd0 [btrfs] __process_folios_contig+0x154/0x268 [btrfs] extent_clear_unlock_delalloc+0x5c/0x80 [btrfs] run_delalloc_nocow+0x5f8/0x760 [btrfs] btrfs_run_delalloc_range+0xa8/0x220 [btrfs] writepage_delalloc+0x230/0x4c8 [btrfs] extent_writepage+0xb8/0x358 [btrfs] extent_write_cache_pages+0x21c/0x4e8 [btrfs] btrfs_writepages+0x94/0x150 [btrfs] do_writepages+0x74/0x190 filemap_fdatawrite_wbc+0x88/0xc8 start_delalloc_inodes+0x178/0x3a8 [btrfs] btrfs_start_delalloc_roots+0x174/0x280 [btrfs] shrink_delalloc+0x114/0x280 [btrfs] flush_space+0x250/0x2f8 [btrfs] btrfs_async_reclaim_data_space+0x180/0x228 [btrfs] process_one_work+0x164/0x408 worker_thread+0x25c/0x388 kthread+0x100/0x118 ret_from_fork+0x10/0x20 Code: 910a8021 a90363f7 a9046bf9 94012379 (d4210000) ---[end trace 0000000000000000]--- [CAUSE] The first two lines of extra debug messages show the problem is caused by the error handling of run_delalloc_nocow(). E.g. we have the following dirtied range (4K blocksize 4K page size): 0 16K 32K |////////////////////////////////////| | Pre-allocated | And the range [0, 16K) has a preallocated extent. - Enter run_delalloc_nocow() for range [0, 16K) Which found range [0, 16K) is preallocated, can do the proper NOCOW write. - Enter fallback_to_cow() for range [16K, 32K) Since the range [16K, 32K) is not backed by preallocated extent, we have to go COW. - cow_file_range() failed for range [16K, 32K) So cow_file_range() will do the clean up by clearing folio dirty, unlock the folios. Now the folios in range [16K, 32K) is unlocked. - Enter extent_clear_unlock_delalloc() from run_delalloc_nocow() Which is called with PAGE_START_WRITEBACK to start page writeback. But folios can only be marked writeback when it's properly locked, thus this triggered the VM_BUG_ON_FOLIO(). Furthermore there is another hidden but common bug that run_delalloc_nocow() is not clearing the folio dirty flags in its error handling path. This is the common bug shared between run_delalloc_nocow() and cow_file_range(). [FIX] - Clear folio dirty for range [@start, @cur_offset) Introduce a helper, cleanup_dirty_folios(), which will find and lock the folio in the range, clear the dirty flag and start/end the writeback, with the extra handling for the @locked_folio. - Introduce a helper to clear folio dirty, start and end writeback - Introduce a helper to record the last failed COW range end This is to trace which range we should

CVE	Severity	CVSS	CVSS Vector	Component	Description
					skip, to avoid double unlocking. - Skip the failed COW range for the e ---truncated---
CVE-2022-38096	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	A NULL pointer dereference vulnerability was found in vmwgfx driver in drivers/gpu/vmxgfx/vmxgfx_execbuf.c in GPU component of Linux kernel with device file '/dev/dri/renderD128 (or Dxxx)'. This flaw allows a local attacker with a user account on the system to gain privilege, causing a denial of service(DoS).
CVE-2025-71118	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ACPICA: Avoid walking the Namespace if start_node is NULL Although commit 0c9992315e73 ("ACPICA: Avoid walking the ACPI Namespace if it is not there") fixed the situation when both start_node and acpi_gbl_root_node are NULL, the Linux kernel mainline now still crashed on Honor Magicbook 14 Pro [1]. That happens due to the access to the member of parent_node in acpi_ns_get_next_node(). The NULL pointer dereference will always happen, no matter whether or not the start_node is equal to ACPI_ROOT_OBJECT, so move the check of start_node being NULL out of the if block. Unfortunately, all the attempts to contact Honor have failed, they refused to provide any technical support for Linux. The bad DSDT table's dump could be found on GitHub [2]. DMI: HONOR FMB-P/FMB-P-PCB, BIOS 1.13 05/08/2025 [rjw: Subject adjustment, changelog edits]
CVE-2024-58012	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ASoC: SOF: Intel: hda-dai: Ensure DAI widget is valid during params Each cpu DAI should associate with a widget. However, the topology might not create the right number of DAI widgets for aggregated amps. And it will cause NULL pointer deference. Check that the DAI widget associated with the CPU DAI is valid to prevent NULL pointer deference due to missing DAI widgets in topologies with aggregated amps.
CVE-2025-21870	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ASoC: SOF: ipc4-topology: Harden loops for looking up ALH copiers Other, non DAI copier widgets could have the same stream name (sname) as the ALH copier and in that case the copier->data is NULL, no alh_data is attached, which could lead to NULL pointer dereference. We could check for this NULL pointer in sof_ipc4_prepare_copier_module() and avoid the crash, but a similar loop in sof_ipc4_widget_setup_comp_dai() will miscalculate the ALH device count, causing broken audio. The correct fix is to harden the matching logic by making sure that the 1. widget is a DAI widget - so dai = w->private is valid 2. the dai (and thus the copier) is ALH copier
CVE-2025-38706	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ASoC: core: Check for rtd == NULL in snd_soc_remove_pcm_runtime() snd_soc_remove_pcm_runtime() might be called with rtd == NULL which will leads to null pointer dereference. This was reproduced with topology loading and marking a link as ignore due to missing hardware component on the system. On module removal the soc_tplg_remove_link() would call snd_soc_remove_pcm_runtime() with rtd == NULL since the link was ignored, no runtime was created.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-39938	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ASoC: qcom: q6apm-lpass-dais: Fix NULL pointer dereference if source graph failed If earlier opening of source graph fails (e.g. ADSP rejects due to incorrect audioreach topology), the graph is closed and "dai_data->graph[dai->id]" is assigned NULL. Preparing the DAI for sink graph continues though and next call to q6apm_lpass_dai_prepare() receives dai_data->graph[dai->id]=NULL leading to NULL pointer exception: qcom-apm gprsvc:service:2:1: Error (1) Processing 0x01001002 cmd qcom-apm gprsvc:service:2:1: DSP returned error[1001002] 1 q6apm-lpass-dais 30000000.remoteproc:glink-edge:gpr:service@1:bedais: fail to start APM port 78 q6apm-lpass-dais 30000000.remoteproc:glink-edge:gpr:service@1:bedais: ASoC: error at snd_soc_pcm_dai_prepare on TX_CODEC_DMA_TX_3: -22 Unable to handle kernel NULL pointer dereference at virtual address 00000000000000a8 ... Call trace: q6apm_graph_media_format_pcm+0x48/0x120 (P) q6apm_lpass_dai_prepare+0x110/0x1b4 snd_soc_pcm_dai_prepare+0x74/0x108 __soc_pcm_prepare+0x44/0x160 dpcm_be_dai_prepare+0x124/0x1c0
CVE-2026-23006	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ASoC: tlv320adcx140: fix null pointer The "snd_soc_component" in "adcx140_priv" was only used once but never set. It was only used for reaching "dev" which is already present in "adcx140_priv".

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38473	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: Bluetooth: Fix null-ptr-deref in l2cap_sock_resume_cb() syzbot reported null-ptr-deref in l2cap_sock_resume_cb(). [0] l2cap_sock_resume_cb() has a similar problem that was fixed by commit 1bff51ea59a9 ("Bluetooth: fix use-after-free error in lock_sock_nested()"). Since both l2cap_sock_kill() and l2cap_sock_resume_cb() are executed under l2cap_sock_resume_cb(), we can avoid the issue simply by checking if chan->data is NULL. Let's not access to the killed socket in l2cap_sock_resume_cb(). [0]: BUG: KASAN: null-ptr-deref in instrument_atomic_write include/linux/instrumented.h:82 [inline] BUG: KASAN: null-ptr-deref in clear_bit include/asm-generic/bitops/instrumented-atomic.h:41 [inline] BUG: KASAN: null-ptr-deref in l2cap_sock_resume_cb+0xb4/0x17c net/bluetooth/l2cap_sock.c:1711 Write of size 8 at addr 0000000000000570 by task kworker/u9:0/52 CPU: 1 UID: 0 PID: 52 Comm: kworker/u9:0 Not tainted 6.16.0-rc4-syzkaller-g7482bb149b9f #0 PREEMPT Hardware name: Google Google Compute Engine/Google Compute Engine, BIOS Google 05/07/2025 Workqueue: hci0 hci_rx_work Call trace: show_stack+0x2c/0x3c arch/arm64/kernel/stacktrace.c:501 (C) __dump_stack+0x30/0x40 lib/dump_stack.c:94 dump_stack_lvl+0xd8/0x12c lib/dump_stack.c:120 print_report+0x58/0x84 mm/kasan/report.c:524 kasan_report+0xb0/0x110 mm/kasan/report.c:634 check_region_inline mm/kasan/generic.c:-1 [inline] kasan_check_range+0x264/0x2a4 mm/kasan/generic.c:189 __kasan_check_write+0x20/0x30 mm/kasan/shadow.c:37 instrument_atomic_write include/linux/instrumented.h:82 [inline] clear_bit include/asm-generic/bitops/instrumented-atomic.h:41 [inline] l2cap_sock_resume_cb+0xb4/0x17c net/bluetooth/l2cap_sock.c:1711 l2cap_security_cfm+0x524/0xea0 net/bluetooth/l2cap_core.c:7357 hci_auth_cfm include/net/bluetooth/hci_core.h:2092 [inline] hci_auth_complete_evt+0x2e8/0xa4c net/bluetooth/hci_event.c:3514 hci_event_func net/bluetooth/hci_event.c:7511 [inline] hci_event_packet+0x650/0xe9c net/bluetooth/hci_event.c:7565 hci_rx_work+0x320/0xb18 net/bluetooth/hci_core.c:4070 process_one_work+0x7e8/0x155c kernel/workqueue.c:3238 process_scheduled_works kernel/workqueue.c:3321 [inline] worker_thread+0x958/0xed8 kernel/workqueue.c:3402 kthread+0x5fc/0x75c kernel/kthread.c:464 ret_from_fork+0x10/0x20 arch/arm64/kernel/entry.S:847

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2026-23146	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: Bluetooth: hci_uart: fix null-ptr-deref in hci_uart_write_work hci_uart_set_proto() sets HCI_UART_PROTO_INIT before calling hci_uart_register_dev(), which calls proto->open() to initialize hu->priv. However, if a TTY write wakeup occurs during this window, hci_uart_tx_wakeup() may schedule write_work before hu->priv is initialized, leading to a NULL pointer dereference in hci_uart_write_work() when proto->dequeue() accesses hu->priv. The race condition is: CPU0 CPU1 ---- ---- hci_uart_set_proto() set_bit(HCI_UART_PROTO_INIT) hci_uart_register_dev() tty write wakeup hci_uart_tty_wakeup() hci_uart_tx_wakeup() schedule_work(&hu->write_work) proto->open(hu) // initializes hu->priv hci_uart_write_work() hci_uart_dequeue() proto->dequeue(hu) // accesses hu->priv (NULL!) Fix this by moving set_bit(HCI_UART_PROTO_INIT) after proto->open() succeeds, ensuring hu->priv is initialized before any work can be scheduled.
CVE-2026-23198	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: KVM: Don't clobber irqfd routing type when deassigning irqfd When deassigning a KVM_IRQFD, don't clobber the irqfd's copy of the IRQ's routing entry as doing so breaks kvm_arch_irq_bypass_del_producer() on x86 and arm64, which explicitly look for KVM_IRQ_ROUTING_MSI. Instead, to handle a concurrent routing update, verify that the irqfd is still active before consuming the routing information. As evidenced by the x86 and arm64 bugs, and another bug in kvm_arch_update_irqfd_routing() (see below), clobbering the entry type without notifying arch code is surprising and error prone. As a bonus, checking that the irqfd is active provides a convenient location for documenting _why_ KVM must not consume the routing entry for an irqfd that is in the process of being deassigned: once the irqfd is deleted from the list (which happens *before* the eventfd is detached), it will no longer receive updates via kvm_irq_routing_update(), and so KVM could deliver an event using stale routing information (relative to KVM_SET_GSI_ROUTING returning to userspace). As an even better bonus, explicitly checking for the irqfd being active fixes a similar bug to the one the clobbering is trying to prevent: if an irqfd is deactivated, and then its routing is changed, kvm_irq_routing_update() won't invoke kvm_arch_update_irqfd_routing() (because the irqfd isn't in the list). And so if the irqfd is in bypass mode, IRQs will continue to be posted using the old routing information. As for kvm_arch_irq_bypass_del_producer(), clobbering the routing type results in KVM incorrectly keeping the IRQ in bypass mode, which is especially problematic on AMD as KVM tracks IRQs that are being posted to a vCPU in a list whose lifetime is tied to the irqfd. Without the help of KASAN to detect use-after-free, the most common sympton on AMD is a NULL pointer deref in amd_iommu_update_ga() due to the memory for irqfd structure being re-allocated and zeroed, resulting in irqfd->irq_bypass_data being NULL when read by avic_update_iommu_vcpu_affinity(): BUG: kernel NULL pointer dereference, address: 0000000000000018 #PF: supervisor read access in kernel mode #PF: error_code(0x0000) - not-present page PGD 40cf2b9067 P4D 40cf2b9067 PUD 408362a067 PMD 0 Oops: Oops: 0000 [#1] SMP CPU: 6 UID: 0 PID: 40383 Comm: vfio_irq_test Tainted: G U W O 6.19.0-smp--5dddc257e6b2-irqfd #31 NONE Tainted: [U]=USER, [W]=WARN, [O]=OOT_MODULE Hardware name: Google, Inc. Arcadia_IT_80/Arcadia_IT_80, BIOS 34.78.2-0 09/05/2025 RIP: 0010:amd_iommu_update_ga+0x19/0xe0 Call Trace:

CVE	Severity	CVSS	CVSS Vector	Component	Description
					avic_update_iommu_vcpu_affinity+0x3d/0x90 [kvm_amd] __avic_vcpu_load+0xf4/0x130 [kvm_amd] kvm_arch_vcpu_load+0x89/0x210 [kvm] vcpu_load+0x30/0x40 [kvm] kvm_arch_vcpu_ioctl_run+0x45/0x620 [kvm] kvm_vcpu_ioctl+0x571/0x6a0 [kvm] __se_sys_ioctl+0x6d/0xb0 do_syscall_64+0x6f/0x9d0 entry_SYSCALL_64_after_hwframe+0x4b/0x53 RIP: 0033:0x46893b ---[end trace 0000000000000000]--- If AVIC is inhibited when the irfd is deassigned, the bug will manifest as list corruption, e.g. on the next irqfd assignment. list_add corruption. next->prev should be prev (ffff8d474d5cd588), but was 0000000000000000. (next=ffff8d8658f86530). ------------[cut here]------------ kernel BUG at lib/list_debug.c:31! Oops: invalid opcode: 0000 [#1] SMP CPU: 128 UID: 0 PID: 80818 Comm: vfio_irq_test Tainted: G U W O 6.19.0-smp--f19dc4d680ba-irqfd #28 NONE Tainted: [U]=USER, [W]=WARN, [O]=OOT_MODULE Hardware name: Google, Inc. Arcadia_IT_80/Arcadia_IT_80, BIOS 34.78.2-0 09/05/2025 RIP: 0010:__list_add_valid_or_report+0x97/0xc0 Call Trace: avic_pi_update_irte+0x28e/0x2b0 [kvm_amd] kvm_pi_update_irte+0xbf/0x190 [kvm] kvm_arch_irq_bypass_add_producer+0x72/0x90 [kvm] irq_bypass_register_consumer+0xcd/0x170 [irqbypa ---truncated---
CVE-2025-38455	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: KVM: SVM: Reject SEV{-ES} intra host migration if vCPU creation is in-flight Reject migration of SEV{-ES} state if either the source or destination VM is actively creating a vCPU, i.e. if kvm_vm_ioctl_create_vcpu() is in the section between incrementing created_vcpus and online_vcpus. The bulk of vCPU creation runs _outside_ of kvm->lock to allow creating multiple vCPUs in parallel, and so sev_info.es_active can get toggled from false=>true in the destination VM after (or during) svm_vcpu_create(), resulting in an SEV{-ES} VM effectively having a non-SEV{-ES} vCPU. The issue manifests most visibly as a crash when trying to free a vCPU's NULL VMSA page in an SEV-ES VM, but any number of things can go wrong. BUG: unable to handle page fault for address: ffffebde00000000 #PF: supervisor read access in kernel mode #PF: error_code(0x0000) - not-present page PGD 0 P4D 0 Oops: Oops: 0000 [#1] SMP KASAN NOPTI CPU: 227 UID: 0 PID: 64063 Comm: syz.5.60023 Tainted: G U O 6.15.0-smp-DEV #2 NONE Tainted: [U]=USER, [O]=OOT_MODULE Hardware name: Google, Inc. Arcadia_IT_80/Arcadia_IT_80, BIOS 12.52.0-0 10/28/2024 RIP: 0010:constant_test_bit arch/x86/include/asm/bitops.h:206 [inline] RIP: 0010:arch_test_bit arch/x86/include/asm/bitops.h:238 [inline] RIP: 0010:_test_bit include/asm-generic/bitops/instrumented-non-atomic.h:142 [inline] RIP: 0010:PageHead include/linux/page-flags.h:866 [inline] RIP: 0010:___free_pages+0x3e/0x120 mm/page_alloc.c:5067 Code: <49> f7 06 40 00 00 00 75 05 45 31 ff eb 0c 66 90 4c 89 f0 4c 39 f0 RSP: 0018:ffff8984551978d0 EFLAGS: 00010246 RAX: 0000777f80000001 RBX: 0000000000000000 RCX: ffffffff918aeb98 RDX: 0000000000000000 RSI: 0000000000000008 RDI: ffffebde00000000 RBP: 0000000000000000 R08: ffffebde00000007 R09: 1ffffd7bc0000000 R10: dfffffc0000000000 R11: fffff97bc0000001 R12: dfffffc0000000000 R13: ffff8983e19751a8 R14: ffffebde00000000 R15: 1ffffd7bc0000000 FS: 0000000000000000(0000) GS:ffff89ee661d3000(0000) knlGS:0000000000000000 CS: 0010 DS: 0000 ES: 0000 CR0: 0000000080050033 CR2: ffffebde00000000 CR3: 000000793ceaa000 CR4: 0000000000350ef0 DR0: 0000000000000000 DR1: 0000000000000b5f DR2: 0000000000000000 DR3:

CVE	Severity	CVSS	CVSS Vector	Component	Description
					0000000000000000 DR6: 00000000ffff0ff0 DR7: 0000000000000400 Call Trace: sev_free_vcpu+0x413/0x630 arch/x86/kvm/svm/sev.c:3169 svm_vcpu_free+0x13a/0x2a0 arch/x86/kvm/svm/svm.c:1515 kvm_arch_vcpu_destroy+0x6a/0x1d0 arch/x86/kvm/x86.c:12396 kvm_vcpu_destroy virt/kvm/kvm_main.c:470 [inline] kvm_destroy_vcpus+0xd1/0x300 virt/kvm/kvm_main.c:490 kvm_arch_destroy_vm+0x636/0x820 arch/x86/kvm/x86.c:12895 kvm_put_kvm+0xb8e/0xfb0 virt/kvm/kvm_main.c:1310 kvm_vm_release+0x48/0x60 virt/kvm/kvm_main.c:1369 __fput+0x3e4/0x9e0 fs/file_table.c:465 task_work_run+0x1a9/0x220 kernel/task_work.c:227 exit_task_work include/linux/task_work.h:40 [inline] do_exit+0x7f0/0x25b0 kernel/exit.c:953 do_group_exit+0x203/0x2d0 kernel/exit.c:1102 get_signal+0x1357/0x1480 kernel/signal.c:3034 arch_do_signal_or_restart+0x40/0x690 arch/x86/kernel/signal.c:337 exit_to_user_mode_loop kernel/entry/common.c:111 [inline] exit_to_user_mode_prepare include/linux/entry-common.h:329 [inline] __syscall_exit_to_user_mode_work kernel/entry/common.c:207 [inline] syscall_exit_to_user_mode+0x67/0xb0 kernel/entry/common.c:218 do_syscall_64+0x7c/0x150 arch/x86/entry/syscall_64.c:100 entry_SYSCALL_64_after_hwframe+0x76/0x7e RIP: 0033:0x7f87a898e969 Modules linked in: gq(O) gsmi: Log Shutdown Reason 0x03 CR2: ffffebde00000000 ---[end trace 0000000000000000]--- Deliberately don't check for a NULL VMSA when freeing the vCPU, as crashing the host is likely desirable due to the VMSA being consumed by hardware. E.g. if KVM manages to allow VMRUN on the vCPU, hardware may read/write a bogus VMSA page. Accessing P ---truncated---
CVE-2025-38696	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: MIPS: Don't crash in stack_top() for tasks without ABI or vDSO Not all tasks have an ABI associated or vDSO mapped, for example kthreads never do. If such a task ever ends up calling stack_top(), it will derefence the NULL ABI pointer and crash. This can for example happen when using kunit: mips_stack_top+0x28/0xc0 arch_pick_mmap_layout+0x190/0x220 kunit_vm_mmap_init+0xf8/0x138 __kunit_add_resource+0x40/0xa8 kunit_vm_mmap+0x88/0xd8 usercopy_test_init+0xb8/0x240 kunit_try_run_case+0x5c/0x1a8 kunit_generic_run_threadfn_adapter+0x28/0x50 kthread+0x118/0x240 ret_from_kernel_thread+0x14/0x1c Only dereference the ABI point if it is set. The GIC page is also included as it is specific to the vDSO. Also move the randomization adjustment into the same conditional.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-71233	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: PCI: endpoint: Avoid creating sub-groups asynchronously The asynchronous creation of sub-groups by a delayed work could lead to a NULL pointer dereference when the driver directory is removed before the work completes. The crash can be easily reproduced with the following commands: # cd /sys/kernel/config/pci_ep/functions/pci_epf_test # for i in {1..20}; do mkdir test && rmdir test; done BUG: kernel NULL pointer dereference, address: 0000000000000088 ... Call Trace: configfs_register_group+0x3d/0x190 pci_epf_cfs_work+0x41/0x110 process_one_work+0x18f/0x350 worker_thread+0x25a/0x3a0 Fix this issue by using configfs_add_default_group() API which does not have the deadlock problem as configfs_register_group() and does not require the delayed work handler. [mani: slightly reworded the description and added stable list]
CVE-2025-38609	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: PM / devfreq: Check governor before using governor->name Commit 96ffcdf239de ("PM / devfreq: Remove redundant governor_name from struct devfreq") removes governor_name and uses governor->name to replace it. But devfreq->governor may be NULL and directly using devfreq->governor->name may cause null pointer exception. Move the check of governor to before using governor->name.
CVE-2025-38387	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: RDMA/mlx5: Initialize obj_event->obj_sub_list before xa_insert The obj_event may be loaded immediately after inserted, then if the list_head is not initialized then we may get a poisonous pointer. This fixes the crash below: mlx5_core 0000:03:00.0: MLX5E: StrdRq(1) RqSz(8) StrdSz(2048) RxCqeCmprss(0 enhanced) mlx5_core.sf mlx5_core.sf.4: firmware version: 32.38.3056 mlx5_core 0000:03:00.0 en3f0pf0sf2002: renamed from eth0 mlx5_core.sf mlx5_core.sf.4: Rate limit: 127 rates are supported, range: 0Mbps to 195312Mbps IPv6: ADDRCONF(NETDEV_CHANGE): en3f0pf0sf2002: link becomes ready Unable to handle kernel NULL pointer dereference at virtual address 0000000000000060 Mem abort info: ESR = 0x96000006 EC = 0x25: DABT (current EL), IL = 32 bits SET = 0, FnV = 0 EA = 0, S1PTW = 0 Data abort info: ISV = 0, ISS = 0x00000006 CM = 0, WnR = 0 user pgtable: 4k pages, 48-bit VAs, pgdp=00000007760fb000 [0000000000000060] pgd=000000076f6d7003, p4d=000000076f6d7003, pud=0000000777841003, pmd=0000000000000000 Internal error: Oops: 96000006 [#1] SMP Modules linked in: ipmb_host(OE) act_mirred(E) cls_flower(E) sch_ingress(E) mptcp_diag(E) udp_diag(E) raw_diag(E) unix_diag(E) tcp_diag(E) inet_diag(E) binfmt_misc(E) bonding(OE) rdma_ucm(OE) rdma_cm(OE) iw_cm(OE) ib_ipoib(OE) ib_cm(OE) isofs(E) cdrom(E) mst_pciconf(OE) ib_umad(OE) mlx5_ib(OE) ipmb_dev_int(OE) mlx5_core(OE) kpatch_15237886(OEK) mlxdevm(OE) auxiliary(OE) ib_uverbs(OE) ib_core(OE) psample(E) mlxfw(OE) tls(E) sunrpc(E) vfat(E) fat(E) crct10dif_ce(E) ghash_ce(E) sha1_ce(E) sbsa_gwdt(E) virtio_console(E) ext4(E) mbcache(E) jbd2(E) xfs(E) libcrc32c(E) mmc_block(E) virtio_net(E) net_failover(E) failover(E) sha2_ce(E) sha256_arm64(E) nvme(OE) nvme_core(OE) gpio_mlxbf3(OE) mlx_compat(OE) mlxbf_pmc(OE) i2c_mlxbf(OE) sdhci_of_dwcmshc(OE) pinctrl_mlxbf3(OE) mlxbf_pka(OE) gpio_generic(E) i2c_core(E) mmc_core(E) mlxbf_gige(OE) vitesse(E) pwr_mlxbf(OE) mlxbf_tmfifo(OE) micrel(E) mlxbf_bootctl(OE) virtio_ring(E) virtio(E) ipmi_devintf(E) ipmi_msghandler(E) [last

CVE	Severity	CVSS	CVSS Vector	Component	Description
					unloaded: mst_pci] CPU: 11 PID: 20913 Comm: rte-worker-11 Kdump: loaded Tainted: G OE K 5.10.134-13.1.an8.aarch64 #1 Hardware name: https://www.mellanox.com BlueField-3 SmartNIC Main Card/BlueField-3 SmartNIC Main Card, BIOS 4.2.2.12968 Oct 26 2023 pstate: a0400089 (NzCv daIf +PAN -UAO -TCO BTYPE=--) pc : dispatch_event_fd+0x68/0x300 [mlx5_ib] lr : devx_event_notifier+0xcc/0x228 [mlx5_ib] sp : ffff80001005bcf0 x29: ffff80001005bcf0 x28: 0000000000000001 x27: ffff244e0740a1d8 x26: ffff244e0740a1d0 x25: ffffda56beff5ae0 x24: ffffda56bf911618 x23: ffff244e0596a480 x22: ffff244e0596a480 x21: ffff244d8312ad90 x20: ffff244e0596a480 x19: fffffffffffffff0 x18: 0000000000000000 x17: 0000000000000000 x16: ffffda56be66d620 x15: 0000000000000000 x14: 0000000000000000 x13: 0000000000000000 x12: 0000000000000000 x11: 0000000000000040 x10: ffffda56bfcafb50 x9 : ffffda5655c25f2c x8 : 0000000000000010 x7 : 0000000000000000 x6 : ffff24545a2e24b8 x5 : 0000000000000003 x4 : ffff80001005bd28 x3 : 0000000000000000 x2 : 0000000000000000 x1 : ffff244e0596a480 x0 : ffff244d8312ad90 Call trace: dispatch_event_fd+0x68/0x300 [mlx5_ib] devx_event_notifier+0xcc/0x228 [mlx5_ib] atomic_notifier_call_chain+0x58/0x80 mlx5_eq_async_int+0x148/0x2b0 [mlx5_core] atomic_notifier_call_chain+0x58/0x80 irq_int_handler+0x20/0x30 [mlx5_core] __handle_irq_event_percpu+0x60/0x220 handle_irq_event_percpu+0x3c/0x90 handle_irq_event+0x58/0x158 handle_fasteoi_irq+0xfc/0x188 generic_handle_irq+0x34/0x48 ...
CVE-2025-71120	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: SUNRPC: svcauth_gss: avoid NULL deref on zero length gss_token in gss_read_proxy_verf A zero length gss_token results in pages == 0 and in_token->pages[0] is NULL. The code unconditionally evaluates page_address(in_token->pages[0]) for the initial memcpy, which can dereference NULL even when the copy length is 0. Guard the first memcpy so it only runs when length > 0.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2026-23107	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: arm64/fpsimd: signal: Allocate SSVE storage when restoring ZA The code to restore a ZA context doesn't attempt to allocate the task's sve_state before setting TIF_SME. Consequently, restoring a ZA context can place a task into an invalid state where TIF_SME is set but the task's sve_state is NULL. In legitimate but uncommon cases where the ZA signal context was NOT created by the kernel in the context of the same task (e.g. if the task is saved/restored with something like CRIU), we have no guarantee that sve_state had been allocated previously. In these cases, userspace can enter streaming mode without trapping while sve_state is NULL, causing a later NULL pointer dereference when the kernel attempts to store the register state: | # ./sigreturn-za | Unable to handle kernel NULL pointer dereference at virtual address 0000000000000000 | Mem abort info: | ESR = 0x0000000096000046 | EC = 0x25: DABT (current EL), IL = 32 bits | SET = 0, FnV = 0 | EA = 0, S1PTW = 0 | FSC = 0x06: level 2 translation fault | Data abort info: | ISV = 0, ISS = 0x00000046, ISS2 = 0x00000000 | CM = 0, WnR = 1, TnD = 0, TagAccess = 0 | GCS = 0, Overlay = 0, DirtyBit = 0, Xs = 0 | user pgtable: 4k pages, 52-bit VAs, pgdp=0000000101f47c00 | [0000000000000000] pgd=08000001021d8403, p4d=0800000102274403, pud=0800000102275403, pmd=0000000000000000 | Internal error: Oops: 0000000096000046 [#1] SMP | Modules linked in: | CPU: 0 UID: 0 PID: 153 Comm: sigreturn-za Not tainted 6.19.0-rc1 #1 PREEMPT | Hardware name: linux,dummy-virt (DT) | pstate: 214000c9 (nzCv daIF +PAN -UAO -TCO +DIT -SSBS BTYPE=--) | pc : sve_save_state+0x4/0xf0 | lr : fpsimd_save_user_state+0xb0/0x1c0 | sp : ffff80008070bcc0 | x29: ffff80008070bcc0 x28: fff00000c1ca4c40 x27: 63cfa172fb5cf658 | x26: fff00000c1ca5228 x25: 0000000000000000 x24: 0000000000000000 | x23: 0000000000000000 x22: fff00000c1ca4c40 x21: fff00000c1ca4c40 | x20: 0000000000000020 x19: fff00000ff6900f0 x18: 0000000000000000 | x17: fff05e8e0311f000 x16: 0000000000000000 x15: 028fca8f3bdaf21c | x14: 0000000000000212 x13: fff00000c0209f10 x12: 0000000000000020 | x11: 0000000000200b20 x10: 0000000000000000 x9 : fff00000ff69dcc0 | x8 : 00000000000003f2 x7 : 0000000000000001 x6 : fff00000c1ca5b48 | x5 : fff05e8e0311f000 x4 : 0000000008000000 x3 : 0000000000000000 | x2 : 0000000000000001 x1 : fff00000c1ca5970 x0 : 0000000000000440 | Call trace: | sve_save_state+0x4/0xf0 (P) | fpsimd_thread_switch+0x48/0x198 | __switch_to+0x20/0x1c0 | __schedule+0x36c/0xce0 | schedule+0x34/0x11c | exit_to_user_mode_loop+0x124/0x188 | el0_interrupt+0xc8/0xd8 | __el0_irq_handler_common+0x18/0x24 | el0t_64_irq_handler+0x10/0x1c | el0t_64_irq+0x198/0x19c | Code: 54000040 d51b4408 d65f03c0 d503245f (e5bb5800) | ---[end trace 0000000000000000]--- Fix this by having restore_za_context() ensure that the task's sve_state is allocated, matching what we do when taking an SME trap. Any live SVE/SSVE state (which is restored earlier from a separate signal context) must be preserved, and hence this is not zeroed.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38458	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: atm: clip: Fix NULL pointer dereference in vcc_sendmsg() atmarpd_dev_ops does not implement the send method, which may cause crash as bellow. BUG: kernel NULL pointer dereference, address: 0000000000000000 PGD 0 P4D 0 Oops: Oops: 0010 [#1] SMP KASAN NOPTI CPU: 0 UID: 0 PID: 5324 Comm: syz.0.0 Not tainted 6.15.0-rc6-syzkaller-00346-g5723cc3450bc #0 PREEMPT(full) Hardware name: QEMU Standard PC (Q35 + ICH9, 2009), BIOS 1.16.3-debian-1.16.3-2~bpo12+1 04/01/2014 RIP: 0010:0x0 Code: Unable to access opcode bytes at 0xffffffffffffffd6. RSP: 0018:ffffc9000d3cf778 EFLAGS: 00010246 RAX: 1ffffffff1910dd1 RBX: 00000000000000c0 RCX: dffffc0000000000 RDX: ffffc9000dc82000 RSI: ffff88803e4c4640 RDI: ffff888052cd0000 RBP: ffffc9000d3cf8d0 R08: ffff888052c9143f R09: 1ffff1100a592287 R10: dffffc0000000000 R11: 0000000000000000 R12: 1ffff92001a79f00 R13: ffff888052cd0000 R14: ffff88803e4c4640 R15: ffffffff8c886e88 FS: 00007fbc762566c0(0000) GS:ffff88808d6c2000(0000) knlGS:0000000000000000 CS: 0010 DS: 0000 ES: 0000 CR0: 0000000080050033 CR2: ffffffffffffffd6 CR3: 0000000041f1b000 CR4: 0000000000352ef0 DR0: 0000000000000000 DR1: 0000000000000000 DR2: 0000000000000000 DR3: 0000000000000000 DR6: 00000000fffe0ff0 DR7: 0000000000000400 Call Trace: vcc_sendmsg+0xa10/0xc50 net/atm/common.c:644 sock_sendmsg_nosec net/socket.c:712 [inline] __sock_sendmsg+0x219/0x270 net/socket.c:727 ____sys_sendmsg+0x52d/0x830 net/socket.c:2566 ___sys_sendmsg+0x21f/0x2a0 net/socket.c:2620 __sys_sendmmsg+0x227/0x430 net/socket.c:2709 __do_sys_sendmmsg net/socket.c:2736 [inline] __se_sys_sendmmsg net/socket.c:2733 [inline] __x64_sys_sendmmsg+0xa0/0xc0 net/socket.c:2733 do_syscall_x64 arch/x86/entry/syscall_64.c:63 [inline] do_syscall_64+0xf6/0x210 arch/x86/entry/syscall_64.c:94 entry_SYSCALL_64_after_hwframe+0x77/0x7f
CVE-2025-38460	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: atm: clip: Fix potential null-ptr-deref in to_atmarpd(). atmarpd is protected by RTNL since commit f3a0592b37b8 ("[ATM]: clip causes unregister hang"). However, it is not enough because to_atmarpd() is called without RTNL, especially clip_neigh_solicit() / neigh_ops->solicit() is unsleepable. Also, there is no RTNL dependency around atmarpd. Let's use a private mutex and RCU to protect access to atmarpd in to_atmarpd().
CVE-2026-23084	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: be2net: Fix NULL pointer dereference in be_cmd_get_mac_from_list When the parameter pmac_id_valid argument of be_cmd_get_mac_from_list() is set to false, the driver may request the PMAC_ID from the firmware of the network card, and this function will store that PMAC_ID at the provided address pmac_id. This is the contract of this function. However, there is a location within the driver where both pmac_id_valid == false and pmac_id == NULL are being passed. This could result in dereferencing a NULL pointer. To resolve this issue, it is necessary to pass the address of a stub variable to the function.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38569	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: benet: fix BUG when creating VFs benet crashes as soon as SRIOV VFs are created: kernel BUG at mm/vmalloc.c:3457! Oops: invalid opcode: 0000 [#1] SMP KASAN NOPTI CPU: 4 UID: 0 PID: 7408 Comm: test.sh Kdump: loaded Not tainted 6.16.0+ #1 PREEMPT(voluntary) [...] RIP: 0010:vunmap+0x5f/0x70 [...] Call Trace: __iommu_dma_free+0xe8/0x1c0 be_cmd_set_mac_list+0x3fe/0x640 [be2net] be_cmd_set_mac+0xaf/0x110 [be2net] be_vf_eth_addr_config+0x19f/0x330 [be2net] be_vf_setup+0x4f7/0x990 [be2net] be_pci_sriov_configure+0x3a1/0x470 [be2net] sriov_numvfs_store+0x20b/0x380 kernfs_fop_write_iter+0x354/0x530 vfs_write+0x9b9/0xf60 ksys_write+0xf3/0x1d0 do_syscall_64+0x8c/0x3d0 be_cmd_set_mac_list() calls dma_free_coherent() under a spin_lock_bh. Fix it by freeing only after the lock has been released.
CVE-2024-56702	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: bpf: Mark raw_tp arguments with PTR_MAYBE_NULL Arguments to a raw tracepoint are tagged as trusted, which carries the semantics that the pointer will be non-NULL. However, in certain cases, a raw tracepoint argument may end up being NULL. More context about this issue is available in [0]. Thus, there is a discrepancy between the reality, that raw_tp arguments can actually be NULL, and the verifier's knowledge, that they are never NULL, causing explicit NULL checks to be deleted, and accesses to such pointers potentially crashing the kernel. To fix this, mark raw_tp arguments as PTR_MAYBE_NULL, and then special case the dereference and pointer arithmetic to permit it, and allow passing them into helpers/kfuncs; these exceptions are made for raw_tp programs only. Ensure that we don't do this when ref_obj_id > 0, as in that case this is an acquired object and doesn't need such adjustment. The reason we do mask_raw_tp_trusted_reg logic is because other will recheck in places whether the register is a trusted_reg, and then consider our register as untrusted when detecting the presence of the PTR_MAYBE_NULL flag. To allow safe dereference, we enable PROBE_MEM marking when we see loads into trusted pointers with PTR_MAYBE_NULL. While trusted raw_tp arguments can also be passed into helpers or kfuncs where such broken assumption may cause issues, a future patch set will tackle their case separately, as PTR_TO_BTF_ID (without PTR_TRUSTED) can already be passed into helpers and causes similar problems. Thus, they are left alone for now. It is possible that these checks also permit passing non-raw_tp args that are trusted PTR_TO_BTF_ID with null marking. In such a case, allowing dereference when pointer is NULL expands allowed behavior, so won't regress existing programs, and the case of passing these into helpers is the same as above and will be dealt with later. Also update the failure case in tp_btf_nullable selftest to capture the new behavior, as the verifier will no longer cause an error when directly dereference a raw tracepoint argument marked as __nullable. [0]: https://lore.kernel.org/bpf/ZrCZS6nisraEqehw@jlelli-thinkpadt14gen4.remote.csb

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2024-42151	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: bpf: mark bpf_dummy_struct_ops.test_1 parameter as nullable Test case dummy_st_ops/dummy_init_ret_value passes NULL as the first parameter of the test_1() function. Mark this parameter as nullable to make verifier aware of such possibility. Otherwise, NULL check in the test_1() code: SEC("struct_ops/test_1") int BPF_PROG(test_1, struct bpf_dummy_ops_state *state) { if (!state) return ...; ... access state ... } Might be removed by verifier, thus triggering NULL pointer dereference under certain conditions.
CVE-2023-52920	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: bpf: support non-r10 register spill/fill to/from stack in precision tracking Use instruction (jump) history to record instructions that performed register spill/fill to/from stack, regardless if this was done through read-only r10 register, or any other register after copying r10 into it *and* potentially adjusting offset. To make this work reliably, we push extra per-instruction flags into instruction history, encoding stack slot index (spi) and stack frame number in extra 10 bit flags we take away from prev_idx in instruction history. We don't touch idx field for maximum performance, as it's checked most frequently during backtracking. This change removes basically the last remaining practical limitation of precision backtracking logic in BPF verifier. It fixes known deficiencies, but also opens up new opportunities to reduce number of verified states, explored in the subsequent patches. There are only three differences in selftests' BPF object files according to veristat, all in the positive direction (less states). File Program Insns (A) Insns (B) Insns (DIFF) States (A) States (B) States (DIFF) ---------- ---------- ---------- ---------- ---------- ---------- ---------- ---------- test_cls_redirect_dynptr.bpf.linked3.o cls_redirect 2987 2864 -123 (-4.12%) 240 231 -9 (-3.75%) xdp_synproxy_kern.bpf.linked3.o syncookie_tc 82848 82661 -187 (-0.23%) 5107 5073 -34 (-0.67%) xdp_synproxy_kern.bpf.linked3.o syncookie_xdp 85116 84964 -152 (-0.18%) 5162 5130 -32 (-0.62%) Note, I avoided renaming jmp_history to more generic insn_hist to minimize number of lines changed and potential merge conflicts between bpf and bpf-next trees. Notice also cur_hist_entry pointer reset to NULL at the beginning of instruction verification loop. This pointer avoids the problem of relying on last jump history entry's insn_idx to determine whether we already have entry for current instruction or not. It can happen that we added jump history entry because current instruction is_jmp_point(), but also we need to add instruction flags for stack access. In this case, we don't want to entries, so we need to reuse last added entry, if it is present. Relying on insn_idx comparison has the same ambiguity problem as the one that was fixed recently in [0], so we avoid that. [0] https://patchwork.kernel.org/project/netdevbpf/patch/20231110002638.4168352-3-andrii@kernel.org/
CVE-2025-38269	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: btrfs: exit after state insertion failure at btrfs_convert_extent_bit() If insert_state() state failed it returns an error pointer and we call extent_io_tree_panic() which will trigger a BUG() call. However if CONFIG_BUG is disabled, which is an uncommon and exotic scenario, then we fallthrough and call cache_state() which will dereference the error pointer, resulting in an invalid memory access. So jump to the 'out' label after calling extent_io_tree_panic(), it also makes the code more clear besides dealing with the exotic scenario where CONFIG_BUG is disabled.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-71184	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: btrfs: fix NULL dereference on root when tracing inode eviction When evicting an inode the first thing we do is to setup tracing for it, which implies fetching the root's id. But in btrfs_evict_inode() the root might be NULL, as implied in the next check that we do in btrfs_evict_inode(). Hence, we either should set the ->root_objectid to 0 in case the root is NULL, or we move tracing setup after checking that the root is not NULL. Setting the rootid to 0 at least gives us the possibility to trace this call even in the case when the root is NULL, so that's the solution taken here.
CVE-2025-38665	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: can: netlink: can_changelink(): fix NULL pointer deref of struct can_priv::do_set_mode Andrei Lalaev reported a NULL pointer deref when a CAN device is restarted from Bus Off and the driver does not implement the struct can_priv::do_set_mode callback. There are 2 code path that call struct can_priv::do_set_mode: - directly by a manual restart from the user space, via can_changelink() - delayed automatic restart after bus off (deactivated by default) To prevent the NULL pointer deference, refuse a manual restart or configure the automatic restart delay in can_changelink() and report the error via extack to user space. As an additional safety measure let can_restart() return an error if can_priv::do_set_mode is not set instead of dereferencing it unchecked.
CVE-2025-38635	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: clk: davinci: Add NULL check in davinci_lpsc_clk_register() devm_kasprintf() returns NULL when memory allocation fails. Currently, davinci_lpsc_clk_register() does not check for this case, which results in a NULL pointer dereference. Add NULL check after devm_kasprintf() to prevent this issue and ensuring no resources are left allocated.
CVE-2025-38583	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: clk: xilinx: vcu: unregister pll_post only if registered correctly If registration of pll_post is failed, it will be set to NULL or ERR, unregistering same will fail with following call trace: Unable to handle kernel NULL pointer dereference at virtual address 008 pc : clk_hw_unregister+0xc/0x20 lr : clk_hw_unregister_fixed_factor+0x18/0x30 sp : ffff800011923850 ... Call trace: clk_hw_unregister+0xc/0x20 clk_hw_unregister_fixed_factor+0x18/0x30 xvcu_unregister_clock_provider+0xcc/0xf4 [xlnx_vcu] xvcu_probe+0x2bc/0x53c [xlnx_vcu]
CVE-2024-50009	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: cpufreq: amd-pstate: add check for cpufreq_cpu_get's return value cpufreq_cpu_get may return NULL. To avoid NULL-dereference check it and return in case of error. Found by Linux Verification Center (linuxtesting.org) with SVACE.
CVE-2026-23060	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: crypto: authencesn - reject too-short AAD (assoclen<8) to match ESP/ESN spec authencesn assumes an ESP/ESN-formatted AAD. When assoclen is shorter than the minimum expected length, crypto_authenc_esn_decrypt() can advance past the end of the destination scatterlist and trigger a NULL pointer dereference in scatterwalk_map_and_copy(), leading to a kernel panic (DoS). Add a minimum AAD length check to fail fast on invalid inputs.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38581	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: crypto: ccp - Fix crash when rebind ccp device for ccp.ko When CONFIG_CRYPTO_DEV_CCP_DEBUGFS is enabled, rebinding the ccp device causes the following crash: $ echo '0000:0a:00.2' > /sys/bus/pci/drivers/ccp/unbind $ echo '0000:0a:00.2' > /sys/bus/pci/drivers/ccp/bind [204.976930] BUG: kernel NULL pointer dereference, address: 0000000000000098 [204.978026] #PF: supervisor write access in kernel mode [204.979126] #PF: error_code(0x0002) - not-present page [204.980226] PGD 0 P4D 0 [204.981317] Oops: Oops: 0002 [#1] SMP NOPTI ... [204.997852] Call Trace: [204.999074] [205.000297] start_creating+0x9f/0x1c0 [205.001533] debugfs_create_dir+0x1f/0x170 [205.002769] ? srso_return_thunk+0x5/0x5f [205.004000] ccp5_debugfs_setup+0x87/0x170 [ccp] [205.005241] ccp5_init+0x8b2/0x960 [ccp] [205.006469] ccp_dev_init+0xd4/0x150 [ccp] [205.007709] sp_init+0x5f/0x80 [ccp] [205.008942] sp_pci_probe+0x283/0x2e0 [ccp] [205.010165] ? srso_return_thunk+0x5/0x5f [205.011376] local_pci_probe+0x4f/0xb0 [205.012584] pci_device_probe+0xdb/0x230 [205.013810] really_probe+0xed/0x380 [205.015024] __driver_probe_device+0x7e/0x160 [205.016240] device_driver_attach+0x2f/0x60 [205.017457] bind_store+0x7c/0xb0 [205.018663] drv_attr_store+0x28/0x40 [205.019868] sysfs_kf_write+0x5f/0x70 [205.021065] kernfs_fop_write_iter+0x145/0x1d0 [205.022267] vfs_write+0x308/0x440 [205.023453] ksys_write+0x6d/0xe0 [205.024616] __x64_sys_write+0x1e/0x30 [205.025778] x64_sys_call+0x16ba/0x2150 [205.026942] do_syscall_64+0x56/0x1e0 [205.028108] entry_SYSCALL_64_after_hwframe+0x76/0x7e [205.029276] RIP: 0033:0x7fbc36f10104 [205.030420] Code: 89 02 48 c7 c0 ff ff ff ff c3 66 2e 0f 1f 84 00 00 00 00 00 66 90 48 8d 05 e1 08 2e 00 8b 00 85 c0 75 13 b8 01 00 00 00 0f 05 <48> 3d 00 f0 ff ff 77 54 f3 c3 66 90 41 54 55 49 89 d4 53 48 89 f5 This patch sets ccp_debugfs_dir to NULL after destroying it in ccp5_debugfs_destroy, allowing the directory dentry to be recreated when rebinding the ccp device. Tested on AMD Ryzen 7 1700X.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2024-41085	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: cxl/mem: Fix no cxl_nvd during pmem region auto-assembling When CXL subsystem is auto-assembling a pmem region during cxl endpoint port probing, always hit below calltrace. BUG: kernel NULL pointer dereference, address: 0000000000000078 #PF: supervisor read access in kernel mode #PF: error_code(0x0000) - not-present page RIP: 0010:cxl_pmem_region_probe+0x22e/0x360 [cxl_pmem] Call Trace: ? __die+0x24/0x70 ? page_fault_oops+0x82/0x160 ? do_user_addr_fault+0x65/0x6b0 ? exc_page_fault+0x7d/0x170 ? asm_exc_page_fault+0x26/0x30 ? cxl_pmem_region_probe+0x22e/0x360 [cxl_pmem] ? cxl_pmem_region_probe+0x1ac/0x360 [cxl_pmem] cxl_bus_probe+0x1b/0x60 [cxl_core] really_probe+0x173/0x410 ? __pfx___device_attach_driver+0x10/0x10 __driver_probe_device+0x80/0x170 driver_probe_device+0x1e/0x90 __device_attach_driver+0x90/0x120 bus_for_each_drv+0x84/0xe0 __device_attach+0xbc/0x1f0 bus_probe_device+0x90/0xa0 device_add+0x51c/0x710 devm_cxl_add_pmem_region+0x1b5/0x380 [cxl_core] cxl_bus_probe+0x1b/0x60 [cxl_core] The cxl_nvd of the memdev needs to be available during the pmem region probe. Currently the cxl_nvd is registered after the endpoint port probe. The endpoint probe, in the case of autoassembly of regions, can cause a pmem region probe requiring the not yet available cxl_nvd. Adjust the sequence so this dependency is met. This requires adding a port parameter to cxl_find_nvdimm_bridge() that can be used to query the ancestor root port. The endpoint port is not yet available, but will share a common ancestor with its parent, so start the query from there instead.
CVE-2025-40251	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: devlink: rate: Unset parent pointer in devl_rate_nodes_destroy The function devl_rate_nodes_destroy is documented to "Unset parent for all rate objects". However, it was only calling the driver-specific `rate_leaf_parent_set` or `rate_node_parent_set` ops and decrementing the parent's refcount, without actually setting the `devlink_rate->parent` pointer to NULL. This leaves a dangling pointer in the `devlink_rate` struct, which cause refcount error in netdevsim[1] and mlx5[2]. In addition, this is inconsistent with the behavior of `devlink_nl_rate_parent_node_set`, where the parent pointer is correctly cleared. This patch fixes the issue by explicitly setting `devlink_rate->parent` to NULL after notifying the driver, thus fulfilling the function's documented behavior for all rate objects. [1] repro steps: echo 1 > /sys/bus/netdevsim/new_device devlink dev eswitch set netdevsim/netdevsim1 mode switchdev echo 1 > /sys/bus/netdevsim/devices/netdevsim1/sriov_numvfs devlink port function rate add netdevsim/netdevsim1/test_node devlink port function rate set netdevsim/netdevsim1/128 parent test_node echo 1 > /sys/bus/netdevsim/del_device dmesg: refcount_t: decrement hit 0; leaking memory. WARNING: CPU: 8 PID: 1530 at lib/refcount.c:31 refcount_warn_saturate+0x42/0xe0 CPU: 8 UID: 0 PID: 1530 Comm: bash Not tainted 6.18.0-rc4+ #1 NONE Hardware name: QEMU Standard PC (Q35 + ICH9, 2009), BIOS rel-1.16.0-0-gd239552ce722-prebuilt.qemu.org 04/01/2014 RIP: 0010:refcount_warn_saturate+0x42/0xe0 Call Trace: devl_rate_leaf_destroy+0x8d/0x90 __nsim_dev_port_del+0x6c/0x70 [netdevsim] nsim_dev_reload_destroy+0x11c/0x140 [netdevsim] nsim_drv_remove+0x2b/0xb0

CVE	Severity	CVSS	CVSS Vector	Component	Description
					[netdevsim] device_release_driver_internal+0x194/0x1f0 bus_remove_device+0xc6/0x130 device_del+0x159/0x3c0 device_unregister+0x1a/0x60 del_device_store+0x111/0x170 [netdevsim] kernfs_fop_write_iter+0x12e/0x1e0 vfs_write+0x215/0x3d0 ksys_write+0x5f/0xd0 do_syscall_64+0x55/0x10f0 entry_SYSCALL_64_after_hwframe+0x4b/0x53 [2] devlink dev eswitch set pci/0000:08:00.0 mode switchdev devlink port add pci/0000:08:00.0 flavour pcisf pfnum 0 sfnum 1000 devlink port function rate add pci/0000:08:00.0/group1 devlink port function rate set pci/0000:08:00.0/32768 parent group1 modprobe -r mlx5_ib mlx5_fwctl mlx5_core dmesg: refcount_t: decrement hit 0; leaking memory. WARNING: CPU: 7 PID: 16151 at lib/refcount.c:31 refcount_warn_saturate+0x42/0xe0 CPU: 7 UID: 0 PID: 16151 Comm: bash Not tainted 6.17.0-rc7_for_upstream_min_debug_2025_10_02_12_44 #1 NONE Hardware name: QEMU Standard PC (Q35 + ICH9, 2009), BIOS rel-1.16.3-0-ga6ed6b701f0a-prebuilt.qemu.org 04/01/2014 RIP: 0010:refcount_warn_saturate+0x42/0xe0 Call Trace: devl_rate_leaf_destroy+0x8d/0x90 mlx5_esw_offloads_devlink_port_unregister+0x33/0x60 [mlx5_core] mlx5_esw_offloads_unload_rep+0x3f/0x50 [mlx5_core] mlx5_eswitch_unload_sf_vport+0x40/0x90 [mlx5_core] mlx5_sf_esw_event+0xc4/0x120 [mlx5_core] notifier_call_chain+0x33/0xa0 blocking_notifier_call_chain+0x3b/0x50 mlx5_eswitch_disable_locked+0x50/0x110 [mlx5_core] mlx5_eswitch_disable+0x63/0x90 [mlx5_core] mlx5_unload+0x1d/0x170 [mlx5_core] mlx5_uninit_one+0xa2/0x130 [mlx5_core] remove_one+0x78/0xd0 [mlx5_core] pci_device_remove+0x39/0xa0 device_release_driver_internal+0x194/0x1f0 unbind_store+0x99/0xa0 kernfs_fop_write_iter+0x12e/0x1e0 vfs_write+0x215/0x3d0 ksys_write+0x5f/0xd0 do_syscall_64+0x53/0x1f0 entry_SYSCALL_64_after_hwframe+0x4b/0x53
CVE-2025-23131	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: dlm: prevent NPD when writing a positive value to event_done do_uevent returns the value written to event_done. In case it is a positive value, new_lockspace would undo all the work, and lockspace would not be set. __dlm_new_lockspace, however, would treat that positive value as a success due to commit 8511a2728ab8 ("dlm: fix use count with multiple joins"). Down the line, device_create_lockspace would pass that NULL lockspace to dlm_find_lockspace_local, leading to a NULL pointer dereference. Treating such positive values as successes prevents the problem. Given this has been broken for so long, this is unlikely to break userspace expectations.
CVE-2026-23206	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: dpaa2-switch: prevent ZERO_SIZE_PTR dereference when num_ifs is zero The driver allocates arrays for ports, FDBs, and filter blocks using kcalloc() with ethsw->sw_attr.num_ifs as the element count. When the device reports zero interfaces (either due to hardware configuration or firmware issues), kcalloc(0, ...) returns ZERO_SIZE_PTR (0x10) instead of NULL. Later in dpaa2_switch_probe(), the NAPI initialization unconditionally accesses ethsw->ports[0]->netdev, which attempts to dereference ZERO_SIZE_PTR (address 0x10), resulting in a kernel panic. Add a check to ensure num_ifs is greater than zero after retrieving device attributes. This prevents the zero-sized allocations and subsequent invalid pointer dereference.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2024-26948	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: drm/amd/display: Add a dc_state NULL check in dc_state_release [How] Check wheather state is NULL before releasing it.
CVE-2025-39675	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: drm/amd/display: Add null pointer check in mod_hdcp_hdcp1_create_session() The function mod_hdcp_hdcp1_create_session() calls the function get_first_active_display(), but does not check its return value. The return value is a null pointer if the display list is empty. This will lead to a null pointer dereference. Add a null pointer check for get_first_active_display() and return MOD_HDCP_STATUS_DISPLAY_NOT_FOUND if the function return null. This is similar to the commit c3e9826a2202 ("drm/amd/display: Add null pointer check for get_first_active_display()"). (cherry picked from commit 5e43eb3cd731649c4f8b9134f857be62a416c893)
CVE-2025-39693	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: drm/amd/display: Avoid a NULL pointer dereference [WHY] Although unlikely drm_atomic_get_new_connector_state() or drm_atomic_get_old_connector_state() can return NULL. [HOW] Check returns before dereference. (cherry picked from commit 1e5e8d672fec9f2ab352be121be971877bff2af9)
CVE-2025-39762	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: drm/amd/display: add null check [WHY] Prevents null pointer dereferences to enhance function robustness [HOW] Adds early null check and return false if invalid.
CVE-2025-39705	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: drm/amd/display: fix a Null pointer dereference vulnerability [Why] A null pointer dereference vulnerability exists in the AMD display driver's (DC module) cleanup function dc_destruct(). When display control context (dc->ctx) construction fails (due to memory allocation failure), this pointer remains NULL. During subsequent error handling when dc_destruct() is called, there's no NULL check before dereferencing the perf_trace member (dc->ctx->perf_trace), causing a kernel null pointer dereference crash. [How] Check if dc->ctx is non-NULL before dereferencing. (Updated commit text and removed unnecessary error message) (cherry picked from commit 9dd8e2ba268c636c240a918e0a31e6feaee19404)
CVE-2025-38705	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: drm/amd/pm: fix null pointer access Writing a string without delimiters (' ', '\n', '\0') to the under gpu_od/fan_ctrl sysfs or pp_power_profile_mode for the CUSTOM profile will result in a null pointer dereference.
CVE-2025-39707	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: drm/amdgpu: check if hubbub is NULL in debugfs/amdgpu_dm_capabilities HUBBUB structure is not initialized on DCE hardware, so check if it is NULL to avoid null dereference while accessing amdgpu_dm_capabilities file in debugfs.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2026-23163	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: drm/amdgpu: fix NULL pointer dereference in amdgpu_gmc_filter_faults_remove On APUs such as Raven and Renoir (GC 9.1.0, 9.2.2, 9.3.0), the ih1 and ih2 interrupt ring buffers are not initialized. This is by design, as these secondary IH rings are only available on discrete GPUs. See vega10_ih_sw_init() which explicitly skips ih1/ih2 initialization when AMD_IS_APU is set. However, amdgpu_gmc_filter_faults_remove() unconditionally uses ih1 to get the timestamp of the last interrupt entry. When retry faults are enabled on APUs (noretry=0), this function is called from the SVM page fault recovery path, resulting in a NULL pointer dereference when amdgpu_ih_decode_iv_ts_helper() attempts to access ih->ring[]. The crash manifests as: BUG: kernel NULL pointer dereference, address: 0000000000000004 RIP: 0010:amdgpu_ih_decode_iv_ts_helper+0x22/0x40 [amdgpu] Call Trace: amdgpu_gmc_filter_faults_remove+0x60/0x130 [amdgpu] svm_range_restore_pages+0xae5/0x11c0 [amdgpu] amdgpu_vm_handle_fault+0xc8/0x340 [amdgpu] gmc_v9_0_process_interrupt+0x191/0x220 [amdgpu] amdgpu_irq_dispatch+0xed/0x2c0 [amdgpu] amdgpu_ih_process+0x84/0x100 [amdgpu] This issue was exposed by commit 1446226d32a4 ("drm/amdgpu: Remove GC HW IP 9.3.0 from noretry=1") which changed the default for Renoir APU from noretry=1 to noretry=0, enabling retry fault handling and thus exercising the buggy code path. Fix this by adding a check for ih1.ring_size before attempting to use it. Also restore the soft_ih support from commit dd299441654f ("drm/amdgpu: Rework retry fault removal"). This is needed if the hardware doesn't support secondary HW IH rings. v2: additional updates (Alex) (cherry picked from commit 6ce8d536c80aa1f059e82184f0d1994436b1d526)
CVE-2025-39706	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: drm/amdkfd: Destroy KFD debugfs after destroy KFD wq Since KFD proc content was moved to kernel debugfs, we can't destroy KFD debugfs before kfd_process_destroy_wq. Move kfd_process_destroy_wq prior to kfd_debugfs_fini to fix a kernel NULL pointer problem. It happens when /sys/kernel/debug/kfd was already destroyed in kfd_debugfs_fini but kfd_process_destroy_wq calls kfd_debugfs_remove_process. This line debugfs_remove_recursive(entry->proc_dentry); tries to remove /sys/kernel/debug/kfd/proc/ while /sys/kernel/debug/kfd is already gone. It hangs the kernel by kernel NULL pointer. (cherry picked from commit 0333052d90683d88531558dcfdbf2525cc37c233)
CVE-2025-38467	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: drm/exynos: exynos7_drm_decon: add vblank check in IRQ handling If there's support for another console device (such as a TTY serial), the kernel occasionally panics during boot. The panic message and a relevant snippet of the call stack is as follows: Unable to handle kernel NULL pointer dereference at virtual address 0000000000000000 Call trace: drm_crtc_handle_vblank+0x10/0x30 (P) decon_irq_handler+0x88/0xb4 [...] Otherwise, the panics don't happen. This indicates that it's some sort of race condition. Add a check to validate if the drm device can handle vblanks before calling drm_crtc_handle_vblank() to avoid this.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-39772	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: drm/hisilicon/hibmc: fix the hibmc loaded failed bug When hibmc loaded failed, the driver use hibmc_unload to free the resource, but the mutexes in mode.config are not init, which will access an NULL pointer. Just change goto statement to return, because hibnc_hw_init() doesn't need to free anything.
CVE-2025-71130	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: drm/i915/gem: Zero-initialize the eb.vma array in i915_gem_do_execbuffer Initialize the eb.vma array with values of 0 when the eb structure is first set up. In particular, this sets the eb->vma[i].vma pointers to NULL, simplifying cleanup and getting rid of the bug described below. During the execution of eb_lookup_vmas(), the eb->vma array is successively filled up with struct eb_vma objects. This process includes calling eb_add_vma(), which might fail; however, even in the event of failure, eb->vma[i].vma is set for the currently processed buffer. If eb_add_vma() fails, eb_lookup_vmas() returns with an error, which prompts a call to eb_release_vmas() to clean up the mess. Since eb_lookup_vmas() might fail during processing any (possibly not first) buffer, eb_release_vmas() checks whether a buffer's vma is NULL to know at what point did the lookup function fail. In eb_lookup_vmas(), eb->vma[i].vma is set to NULL if either the helper function eb_lookup_vma() or eb_validate_vma() fails. eb->vma[i+1].vma is set to NULL in case i915_gem_object_userptr_submit_init() fails; the current one needs to be cleaned up by eb_release_vmas() at this point, so the next one is set. If eb_add_vma() fails, neither the current nor the next vma is set to NULL, which is a source of a NULL deref bug described in the issue linked in the Closes tag. When entering eb_lookup_vmas(), the vma pointers are set to the slab poison value, instead of NULL. This doesn't matter for the actual lookup, since it gets overwritten anyway, however the eb_release_vmas() function only recognizes NULL as the stopping value, hence the pointers are being set to NULL as they go in case of intermediate failure. This patch changes the approach to filling them all with NULL at the start instead, rather than handling that manually during failure. (cherry picked from commit 08889b706d4f0b8d2352b7ca29c2d8df4d0787cd)
CVE-2025-71138	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: drm/msm/dpu: Add missing NULL pointer check for pingpong interface It is checked almost always in dpu_encoder_phys_wb_setup_ctl(), but in a single place the check is missing. Also use convenient locals instead of phys_enc->* where available. Patchwork: https://patchwork.freedesktop.org/patch/693860/
CVE-2025-39747	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: drm/msm: Add error handling for krealloc in metadata setup Function msm_ioctl_gem_info_set_metadata() now checks for krealloc failure and returns -ENOMEM, avoiding potential NULL pointer dereference. Explicitly avoids __GFP_NOFAIL due to deadlock risks and allocation constraints. Patchwork: https://patchwork.freedesktop.org/patch/661235/

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38597	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: drm/rockchip: vop2: fail cleanly if missing a primary plane for a video-port Each window of a vop2 is usable by a specific set of video ports, so while binding the vop2, we look through the list of available windows trying to find one designated as primary-plane and usable by that specific port. The code later wants to use drm_crtc_init_with_planes with that found primary plane, but nothing has checked so far if a primary plane was actually found. For whatever reason, the rk3576 vp2 does not have a usable primary window (if vp0 is also in use) which brought the issue to light and ended in a null-pointer dereference further down. As we expect a primary-plane to exist for a video-port, add a check at the end of the window-iteration and fail probing if none was found.
CVE-2025-38543	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: drm/tegra: nvdec: Fix dma_alloc_coherent error check Check for NULL return value with dma_alloc_coherent, in line with Robin's fix for vic.c in 'drm/tegra: vic: Fix DMA API misuse'.
CVE-2025-71083	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: drm/ttm: Avoid NULL pointer deref for evicted BOs It is possible for a BO to exist that is not currently associated with a resource, e.g. because it has been evicted. When devcoredump tries to read the contents of all BOs for dumping, we need to expect this as well -- in this case, ENODATA is recorded instead of the buffer contents.
CVE-2025-38189	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: drm/v3d: Avoid NULL pointer dereference in `v3d_job_update_stats()` The following kernel Oops was recently reported by Mesa CI: [800.139824] Unable to handle kernel NULL pointer dereference at virtual address 0000000000000588 [800.148619] Mem abort info: [800.151402] ESR = 0x0000000096000005 [800.155141] EC = 0x25: DABT (current EL), IL = 32 bits [800.160444] SET = 0, FnV = 0 [800.163488] EA = 0, S1PTW = 0 [800.166619] FSC = 0x05: level 1 translation fault [800.171487] Data abort info: [800.174357] ISV = 0, ISS = 0x00000005, ISS2 = 0x00000000 [800.179832] CM = 0, WnR = 0, TnD = 0, TagAccess = 0 [800.184873] GCS = 0, Overlay = 0, DirtyBit = 0, Xs = 0 [800.190176] user pgtable: 4k pages, 39-bit VAs, pgdp=00000001014c2000 [800.196607] [0000000000000588] pgd=0000000000000000, p4d=0000000000000000, pud=0000000000000000 [800.205305] Internal error: Oops: 0000000096000005 [#1] PREEMPT SMP [800.211564] Modules linked in: vc4 snd_soc_hdmi_codec drm_display_helper v3d cec gpu_sched drm_dma_helper drm_shmem_helper drm_kms_helper drm drm_panel_orientation_quirks snd_soc_core snd_compress snd_pcm_dmaengine snd_pcm i2c_brcmstb snd_timer snd backlight [800.234448] CPU: 0 UID: 0 PID: 0 Comm: swapper/0 Not tainted 6.12.25+rpt-rpi-v8 #1 Debian 1:6.12.25-1+rpt1 [800.244182] Hardware name: Raspberry Pi 4 Model B Rev 1.4 (DT) [800.250005] pstate: 600000c5 (nZCv daIF -PAN -UAO -TCO -DIT -SSBS BTYPE=--) [800.256959] pc : v3d_job_update_stats+0x60/0x130 [v3d] [800.262112] lr : v3d_job_update_stats+0x48/0x130 [v3d] [800.267251] sp : ffffffc080003e60 [800.270555] x29: ffffffc080003e60 x28: ffffffd842784980 x27: 0224012000000000 [800.277687] x26: ffffffd84277f630 x25: ffffff81012fd800 x24: 0000000000000020 [800.284818] x23: ffffff8040238b08 x22: 0000000000000570 x21: 0000000000000158 [800.291948] x20: 0000000000000000 x19: ffffff8040238000 x18: 0000000000000000 [800.299078] x17: ffffffa8c1bd2000 x16: ffffffc080000000 x15: 0000000000000000 [800.306208] x14:

CVE	Severity	CVSS	CVSS Vector	Component	Description
					0000000000000000 x13: 0000000000000000 x12: 0000000000000000 [800.313338] x11: 0000000000000040 x10: 0000000000001a40 x9 : ffffffd83b39757c [800.320468] x8 : ffffffd842786420 x7 : 7fffffffffffffff x6 : 0000000000ef32b0 [800.327598] x5 : 00ffffffffffffff x4 : 0000000000000015 x3 : ffffffd842784980 [800.334728] x2 : 0000000000000004 x1 : 0000000000010002 x0 : 000000ba4c0ca382 [800.341859] Call trace: [800.344294] v3d_job_update_stats+0x60/0x130 [v3d] [800.349086] v3d_irq+0x124/0x2e0 [v3d] [800.352835] __handle_irq_event_percpu+0x58/0x218 [800.357539] handle_irq_event+0x54/0xb8 [800.361369] handle_fasteoi_irq+0xac/0x240 [800.365458] handle_irq_desc+0x48/0x68 [800.369200] generic_handle_domain_irq+0x24/0x38 [800.373810] gic_handle_irq+0x48/0xd8 [800.377464] call_on_irq_stack+0x24/0x58 [800.381379] do_interrupt_handler+0x88/0x98 [800.385554] el1_interrupt+0x34/0x68 [800.389123] el1h_64_irq_handler+0x18/0x28 [800.393211] el1h_64_irq+0x64/0x68 [800.396603] default_idle_call+0x3c/0x168 [800.400606] do_idle+0x1fc/0x230 [800.403827] cpu_startup_entry+0x40/0x50 [800.407742] rest_init+0xe4/0xf0 [800.410962] start_kernel+0x5e8/0x790 [800.414616] __primary_switched+0x80/0x90 [800.418622] Code: 8b170277 8b160296 11000421 b9000861 (b9401ac1) [800.424707] ---[end trace 0000000000000000]--- [800.457313] ---[end Kernel panic - not syncing: Oops: Fatal exception in interrupt]--- This issue happens when the file descriptor is closed before the jobs submitted by it are completed. When the job completes, we update the global GPU stats and the per-fd GPU stats, which are exposed through fdinfo. If the file descriptor was closed, then the struct `v3d_file_priv` and its stats were already freed and we can't update the per-fd stats. Therefore, if the file descriptor was already closed, don't u ---truncated---
CVE-2025-38371	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: drm/v3d: Disable interrupts before resetting the GPU Currently, an interrupt can be triggered during a GPU reset, which can lead to GPU hangs and NULL pointer dereference in an interrupt context as shown in the following trace: [314.035040] Unable to handle kernel NULL pointer dereference at virtual address 00000000000000c0 [314.043822] Mem abort info: [314.046606] ESR = 0x0000000096000005 [314.050347] EC = 0x25: DABT (current EL), IL = 32 bits [314.055651] SET = 0, FnV = 0 [314.058695] EA = 0, S1PTW = 0 [314.061826] FSC = 0x05: level 1 translation fault [314.066694] Data abort info: [314.069564] ISV = 0, ISS = 0x00000005, ISS2 = 0x00000000 [314.075039] CM = 0, WnR = 0, TnD = 0, TagAccess = 0 [314.080080] GCS = 0, Overlay = 0, DirtyBit = 0, Xs = 0 [314.085382] user pgtable: 4k pages, 39-bit VAs, pgdp=0000000102728000 [314.091814] [00000000000000c0] pgd=0000000000000000, p4d=0000000000000000, pud=0000000000000000 [314.100511] Internal error: Oops: 0000000096000005 [#1] PREEMPT SMP [314.106770] Modules linked in: v3d i2c_brcmstb vc4 snd_soc_hdmi_codec gpu_sched drm_shmem_helper drm_display_helper cec drm_dma_helper drm_kms_helper drm drm_panel_orientation_quirks snd_soc_core snd_compress snd_pcm_dmaengine snd_pcm snd_timer snd backlight [314.129654] CPU: 0 UID: 0 PID: 0 Comm: swapper/0 Not tainted 6.12.25+rpt-rpi-v8 #1 Debian 1:6.12.25-1+rpt1 [314.139388] Hardware name: Raspberry Pi 4 Model B Rev 1.4 (DT) [314.145211] pstate: 600000c5 (nZCv daIF -PAN -UAO -TCO -DIT -SSBS BTYPE=--) [314.152165] pc : v3d_irq+0xec/0x2e0 [v3d] [314.156187] lr : v3d_irq+0xe0/0x2e0 [v3d] [314.160198] sp :

CVE	Severity	CVSS	CVSS Vector	Component	Description
					ffffffc080003ea0 [314.163502] x29: ffffffc080003ea0 x28: ffffffec1f184980 x27: 021202b000000000 [314.170633] x26: ffffffec1f17f630 x25: ffffff8101372000 x24: ffffffec1f17d9f0 [314.177764] x23: 000000000000002a x22: 000000000000002a x21: ffffff8103252000 [314.184895] x20: 0000000000000001 x19: 00000000deadbeef x18: 0000000000000000 [314.192026] x17: ffffff94e51d2000 x16: ffffffec1dac3cb0 x15: c306000000000000 [314.199156] x14: 0000000000000000 x13: b2fc982e03cc5168 x12: 0000000000000001 [314.206286] x11: ffffff8103f8bcc0 x10: ffffffec1f196868 x9 : ffffffec1dac3874 [314.213416] x8 : 0000000000000000 x7 : 0000000000042a3a x6 : ffffff810017a180 [314.220547] x5 : ffffffec1ebad400 x4 : ffffffec1ebad320 x3 : 00000000000bebeb [314.227677] x2 : 0000000000000000 x1 : 0000000000000000 x0 : 0000000000000000 [314.234807] Call trace: [314.237243] v3d_irq+0xec/0x2e0 [v3d] [314.240906] __handle_irq_event_percpu+0x58/0x218 [314.245609] handle_irq_event+0x54/0xb8 [314.249439] handle_fasteoi_irq+0xac/0x240 [314.253527] handle_irq_desc+0x48/0x68 [314.257269] generic_handle_domain_irq+0x24/0x38 [314.261879] gic_handle_irq+0x48/0xd8 [314.265533] call_on_irq_stack+0x24/0x58 [314.269448] do_interrupt_handler+0x88/0x98 [314.273624] el1_interrupt+0x34/0x68 [314.277193] el1h_64_irq_handler+0x18/0x28 [314.281281] el1h_64_irq+0x64/0x68 [314.284673] default_idle_call+0x3c/0x168 [314.288675] do_idle+0x1fc/0x230 [314.291895] cpu_startup_entry+0x3c/0x50 [314.295810] rest_init+0xe4/0xf0 [314.299030] start_kernel+0x5e8/0x790 [314.302684] __primary_switched+0x80/0x90 [314.306691] Code: 940029eb 360ffc13 f9442ea0 52800001 (f9406017) [314.312775] ---[end trace 0000000000000000]--- [314.317384] Kernel panic - not syncing: Oops: Fatal exception in interrupt [314.324249] SMP: stopping secondary CPUs [314.328167] Kernel Offset: 0x2b9da00000 from 0xffffffc080000000 [314.334076] PHYS_OFFSET: 0x0 [314.336946] CPU features: 0x08,00002013,c0200000,0200421b [314.342337] Memory Limit: none [314.345382] ---[end Kernel panic - not syncing: Oops: Fatal exception in interrupt]--- Before resetting the G ---truncated---
CVE-2024-42081	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: drm/xe/xe_devcoredump: Check NULL before assignments Assign 'xe_devcoredump_snapshot *' and 'xe_device *' only if 'coredump' is not NULL. v2 - Fix commit messages. v3 - Define variables before code. (Ashutosh/Jose) v4 - Drop return check for coredump_to_xe. (Jose/Rodrigo) v5 - Modify misleading commit message. (Matt)
CVE-2024-42065	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: drm/xe: Add a NULL check in xe_ttm_stolen_mgr_init Add an explicit check to ensure that the mgr is not NULL.
CVE-2024-46705	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: drm/xe: reset mmio mappings with devm Set our various mmio mappings to NULL. This should make it easier to catch something rogue trying to mess with mmio after device removal. For example, we might unmap everything and then start hitting some mmio address which has already been unmamped by us and then remapped by something else, causing all kinds of carnage.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-39934	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: drm: bridge: anx7625: Fix NULL pointer dereference with early IRQ If the interrupt occurs before resource initialization is complete, the interrupt handler/worker may access uninitialized data such as the I2C tcpc_client device, potentially leading to NULL pointer dereference.
CVE-2025-71107	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: f2fs: ensure node page reads complete before f2fs_put_super() finishes Xfstests generic/335, generic/336 sometimes crash with the following message: F2FS-fs (dm-0): detect filesystem reference count leak during umount, type: 9, count: 1 ------------[cut here]------------ kernel BUG at fs/f2fs/super.c:1939! Oops: invalid opcode: 0000 [#1] SMP NOPTI CPU: 1 UID: 0 PID: 609351 Comm: umount Tainted: G W 6.17.0-rc5-xfstests-g9dd1835ecda5 #1 PREEMPT(none) Tainted: [W]=WARN Hardware name: QEMU Standard PC (Q35 + ICH9, 2009), BIOS 1.16.3-debian-1.16.3-2 04/01/2014 RIP: 0010:f2fs_put_super+0x3b3/0x3c0 Call Trace: generic_shutdown_super+0x7e/0x190 kill_block_super+0x1a/0x40 kill_f2fs_super+0x9d/0x190 deactivate_locked_super+0x30/0xb0 cleanup_mnt+0xba/0x150 task_work_run+0x5c/0xa0 exit_to_user_mode_loop+0xb7/0xc0 do_syscall_64+0x1ae/0x1c0 entry_SYSCALL_64_after_hwframe+0x76/0x7e ---[end trace 0000000000000000]--- It appears that sometimes it is possible that f2fs_put_super() is called before all node page reads are completed. Adding a call to f2fs_wait_on_all_pages() for F2FS_RD_NODE fixes the problem.
CVE-2025-38630	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: fbdev: imxfb: Check fb_add_videomode to prevent null-ptr-deref fb_add_videomode() can fail with -ENOMEM when its internal kmalloc() cannot allocate a struct fb_modelist. If that happens, the modelist stays empty but the driver continues to register. Add a check for its return value to prevent poteintial null-ptr-deref, which is similar to the commit 17186f1f90d3 ("fbdev: Fix do_register_framebuffer to prevent null-ptr-deref in fb_videomode_to_var").

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-22070	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: fs/9p: fix NULL pointer dereference on mkdir When a 9p tree was mounted with option 'posixacl', parent directory had a default ACL set for its subdirectories, e.g.: setfacl -m default:group:simpsons:rwx parentdir then creating a subdirectory crashed 9p client, as v9fs_fid_add() call in function v9fs_vfs_mkdir_dotl() sets the passed 'fid' pointer to NULL (since dafbe689736) even though the subsequent v9fs_set_create_acl() call expects a valid non-NULL 'fid' pointer: [37.273191] BUG: kernel NULL pointer dereference, address: 0000000000000000 ... [37.322338] Call Trace: [37.323043] [37.323621] ? __die (arch/x86/kernel/dumpstack.c:421 arch/x86/kernel/dumpstack.c:434) [37.324448] ? page_fault_oops (arch/x86/mm/fault.c:714) [37.325532] ? search_module_extables (kernel/module/main.c:3733) [37.326742] ? p9_client_walk (net/9p/client.c:1165) 9pnet [37.328006] ? search_bpf_extables (kernel/bpf/core.c:804) [37.329142] ? exc_page_fault (./arch/x86/include/asm/paravirt.h:686 arch/x86/mm/fault.c:1488 arch/x86/mm/fault.c:1538) [37.330196] ? asm_exc_page_fault (./arch/x86/include/asm/idtentry.h:574) [37.331330] ? p9_client_walk (net/9p/client.c:1165) 9pnet [37.332562] ? v9fs_fid_xattr_get (fs/9p/xattr.c:30) 9p [37.333824] v9fs_fid_xattr_set (fs/9p/fid.h:23 fs/9p/xattr.c:121) 9p [37.335077] v9fs_set_acl (fs/9p/acl.c:276) 9p [37.336112] v9fs_set_create_acl (fs/9p/acl.c:307) 9p [37.337326] v9fs_vfs_mkdir_dotl (fs/9p/vfs_inode_dotl.c:411) 9p [37.338590] vfs_mkdir (fs/namei.c:4313) [37.339535] do_mkdirat (fs/namei.c:4336) [37.340465] __x64_sys_mkdir (fs/namei.c:4354) [37.341455] do_syscall_64 (arch/x86/entry/common.c:52 arch/x86/entry/common.c:83) [37.342447] entry_SYSCALL_64_after_hwframe (arch/x86/entry/entry_64.S:130) Fix this by simply swapping the sequence of these two calls in v9fs_vfs_mkdir_dotl(), i.e. calling v9fs_set_create_acl() before v9fs_fid_add().

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-37806	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: fs/ntfs3: Keep write operations atomic syzbot reported a NULL pointer dereference in __generic_file_write_iter. [1] Before the write operation is completed, the user executes ioctl[2] to clear the compress flag of the file, which causes the is_compressed() judgment to return 0, further causing the program to enter the wrong process and call the wrong ops ntfs_aops_cmpr, which triggers the null pointer dereference of write_begin. Use inode lock to synchronize ioctl and write to avoid this case. [1] Unable to handle kernel NULL pointer dereference at virtual address 0000000000000000 Mem abort info: ESR = 0x0000000086000006 EC = 0x21: IABT (current EL), IL = 32 bits SET = 0, FnV = 0 EA = 0, S1PTW = 0 FSC = 0x06: level 2 translation fault user pgtable: 4k pages, 48-bit VAs, pgdp=000000011896d000 [0000000000000000] pgd=0800000118b44403, p4d=0800000118b44403, pud=0800000117517403, pmd=0000000000000000 Internal error: Oops: 0000000086000006 [#1] PREEMPT SMP Modules linked in: CPU: 0 UID: 0 PID: 6427 Comm: syz-executor347 Not tainted 6.13.0-rc3-syzkaller-g573067a5a685 #0 Hardware name: Google Google Compute Engine/Google Compute Engine, BIOS Google 09/13/2024 pstate: 80400005 (Nzcv daif +PAN -UAO -TCO -DIT -SSBS BTYPE=--) pc : 0x0 lr : generic_perform_write+0x29c/0x868 mm/filemap.c:4055 sp : ffff80009d4978a0 x29: ffff80009d4979c0 x28: dfff800000000000 x27: ffff80009d497bc8 x26: 0000000000000000 x25: ffff80009d497960 x24: ffff80008ba71c68 x23: 0000000000000000 x22: ffff0000c655dac0 x21: 0000000000001000 x20: 000000000000000c x19: 1ffff00013a92f2c x18: ffff0000e183aa1c x17: 0004060000000014 x16: ffff800083275834 x15: 0000000000000001 x14: 0000000000000000 x13: 0000000000000001 x12: ffff0000c655dac0 x11: 0000000000ff0100 x10: 0000000000ff0100 x9 : 0000000000000000 x8 : 0000000000000000 x7 : 0000000000000000 x6 : 0000000000000000 x5 : ffff80009d497980 x4 : ffff80009d497960 x3 : 0000000000001000 x2 : 0000000000000000 x1 : ffff0000e183a928 x0 : ffff0000d60b0fc0 Call trace: 0x0 (P) __generic_file_write_iter+0xfc/0x204 mm/filemap.c:4156 ntfs_file_write_iter+0x54c/0x630 fs/ntfs3/file.c:1267 new_sync_write fs/read_write.c:586 [inline] vfs_write+0x920/0xcf4 fs/read_write.c:679 ksys_write+0x15c/0x26c fs/read_write.c:731 __do_sys_write fs/read_write.c:742 [inline] __se_sys_write fs/read_write.c:739 [inline] __arm64_sys_write+0x7c/0x90 fs/read_write.c:739 __invoke_syscall arch/arm64/kernel/syscall.c:35 [inline] invoke_syscall+0x98/0x2b8 arch/arm64/kernel/syscall.c:49 el0_svc_common+0x130/0x23c arch/arm64/kernel/syscall.c:132 do_el0_svc+0x48/0x58 arch/arm64/kernel/syscall.c:151 el0_svc+0x54/0x168 arch/arm64/kernel/entry-common.c:744 el0t_64_sync_handler+0x84/0x108 arch/arm64/kernel/entry-common.c:762 [2] ioctl$FS_IOC_SETFLAGS(r0, 0x40086602, &(0x7f00000000c0)=0x20)
CVE-2025-38408	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: genirq/irq_sim: Initialize work context pointers properly Initialize `ops` member's pointers properly by using kzalloc() instead of kmalloc() when allocating the simulation work context. Otherwise the pointers contain random content leading to invalid dereferencing.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38659	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: gfs2: No more self recovery When a node withdraws and it turns out that it is the only node that has the filesystem mounted, gfs2 currently tries to replay the local journal to bring the filesystem back into a consistent state. Not only is that a very bad idea, it has also never worked because gfs2_recover_func() will refuse to do anything during a withdraw. However, before even getting to this point, gfs2_recover_func() dereferences sdp->sd_jdesc->jd_inode. This was a use-after-free before commit 04133b607a78 ("gfs2: Prevent double iput for journal on error") and is a NULL pointer dereference since then. Simply get rid of self recovery to fix that.
CVE-2025-38735	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: gve: prevent ethtool ops after shutdown A crash can occur if an ethtool operation is invoked after shutdown() is called. shutdown() is invoked during system shutdown to stop DMA operations without performing expensive deallocations. It is discouraged to unregister the netdev in this path, so the device may still be visible to userspace and kernel helpers. In gve, shutdown() tears down most internal data structures. If an ethtool operation is dispatched after shutdown(), it will dereference freed or NULL pointers, leading to a kernel panic. While graceful shutdown normally quiesces userspace before invoking the reboot syscall, forced shutdowns (as observed on GCP VMs) can still trigger this path. Fix by calling netif_device_detach() in shutdown(). This marks the device as detached so the ethtool ioctl handler will skip dispatching operations to the driver.
CVE-2025-38716	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: hfs: fix general protection fault in hfs_find_init() The hfs_find_init() method can trigger the crash if tree pointer is NULL: [45.746290][T9787] Oops: general protection fault, probably for non-canonical address 0xdfffffc0000000008: 0000 [#1] SMP KAI [45.747287][T9787] KASAN: null-ptr-deref in range [0x0000000000000040-0x0000000000000047] [45.748716][T9787] CPU: 2 UID: 0 PID: 9787 Comm: repro Not tainted 6.16.0-rc3 #10 PREEMPT(full) [45.750250][T9787] Hardware name: QEMU Ubuntu 24.04 PC (i440FX + PIIX, 1996), BIOS 1.16.3-debian-1.16.3-2 04/01/2014 [45.751983][T9787] RIP: 0010:hfs_find_init+0x86/0x230 [45.752834][T9787] Code: c1 ea 03 80 3c 02 00 0f 85 9a 01 00 00 4c 8d 6b 40 48 c7 45 18 00 00 00 00 48 b8 00 00 00 00 00 fc [45.755574][T9787] RSP: 0018:ffffc90015157668 EFLAGS: 00010202 [45.756432][T9787] RAX: dffffc0000000000 RBX: 0000000000000000 RCX: ffffffff819a4d09 [45.757457][T9787] RDX: 0000000000000008 RSI: ffffffff819acd3a RDI: ffffc900151576e8 [45.758282][T9787] RBP: ffffc900151576d0 R08: 0000000000000005 R09: 0000000000000000 [45.758943][T9787] R10: 0000000080000000 R11: 0000000000000001 R12: 0000000000000004 [45.759619][T9787] R13: 0000000000000040 R14: ffff88802c50814a R15: 0000000000000000 [45.760293][T9787] FS: 00007ffb72734540(0000) GS:ffff8880cec64000(0000) knlGS:0000000000000000 [45.761050][T9787] CS: 0010 DS: 0000 ES: 0000 CR0: 0000000080050033 [45.761606][T9787] CR2: 00007f9bd8225000 CR3: 000000010979a000 CR4: 00000000000006f0 [45.762286][T9787] Call Trace: [45.762570][T9787] [45.762824][T9787] hfs_ext_read_extent+0x190/0x9d0 [45.763269][T9787] ? submit_bio_noacct_nocheck+0x2dd/0xce0 [45.763766][T9787] ? __pfx_hfs_ext_read_extent+0x10/0x10 [45.764250][T9787] hfs_get_block+0x55f/0x830 [45.764646][T9787]

CVE	Severity	CVSS	CVSS Vector	Component	Description
					block_read_full_folio+0x36d/0x850 [45.765105][T9787] ? __pfx_hfs_get_block+0x10/0x10 [45.765541][T9787] ? const_folio_flags+0x5b/0x100 [45.765972][T9787] ? __pfx_hfs_read_folio+0x10/0x10 [45.766415][T9787] filemap_read_folio+0xbe/0x290 [45.766840][T9787] ? __pfx_filemap_read_folio+0x10/0x10 [45.767325][T9787] ? __filemap_get_folio+0x32b/0xbf0 [45.767780][T9787] do_read_cache_folio+0x263/0x5c0 [45.768223][T9787] ? __pfx_hfs_read_folio+0x10/0x10 [45.768666][T9787] read_cache_page+0x5b/0x160 [45.769070][T9787] hfs_btree_open+0x491/0x1740 [45.769481][T9787] hfs_mdb_get+0x15e2/0x1fb0 [45.769877][T9787] ? __pfx_hfs_mdb_get+0x10/0x10 [45.770316][T9787] ? find_held_lock+0x2b/0x80 [45.770731][T9787] ? lockdep_init_map_type+0x5c/0x280 [45.771200][T9787] ? lockdep_init_map_type+0x5c/0x280 [45.771674][T9787] hfs_fill_super+0x38e/0x720 [45.772092][T9787] ? __pfx_hfs_fill_super+0x10/0x10 [45.772549][T9787] ? snprintf+0xbe/0x100 [45.772931][T9787] ? __pfx_snprintf+0x10/0x10 [45.773350][T9787] ? do_raw_spin_lock+0x129/0x2b0 [45.773796][T9787] ? find_held_lock+0x2b/0x80 [45.774215][T9787] ? set_blocksize+0x40a/0x510 [45.774636][T9787] ? sb_set_blocksize+0x176/0x1d0 [45.775087][T9787] ? setup_bdev_super+0x369/0x730 [45.775533][T9787] get_tree_bdev_flags+0x384/0x620 [45.775985][T9787] ? __pfx_hfs_fill_super+0x10/0x10 [45.776453][T9787] ? __pfx_get_tree_bdev_flags+0x10/0x10 [45.776950][T9787] ? bpf_lsm_capable+0x9/0x10 [45.777365][T9787] ? security_capable+0x80/0x260 [45.777803][T9787] vfs_get_tree+0x8e/0x340 [45.778203][T9787] path_mount+0x13de/0x2010 [45.778604] [T9787] ? kmem_cache_free+0x2b0/0x4c0 [45.779052][T9787] ? __pfx_path_mount+0x10/0x10 [45.779480][T9787] ? getname_flags.part.0+0x1c5/0x550 [45.779954][T9787] ? putname+0x154/0x1a0 [45.780335][T9787] __x64_sys_mount+0x27b/0x300 [45.780758][T9787] ? __pfx___x64_sys_mount+0x10/0x10 [45.781232][T9787] ---truncated---

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38683	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: hv_netvsc: Fix panic during namespace deletion with VF The existing code move the VF NIC to new namespace when NETDEV_REGISTER is received on netvsc NIC. During deletion of the namespace, default_device_exit_batch() >> default_device_exit_net() is called. When netvsc NIC is moved back and registered to the default namespace, it automatically brings VF NIC back to the default namespace. This will cause the default_device_exit_net() >> for_each_netdev_safe loop unable to detect the list end, and hit NULL ptr: [231.449420] mana 7870:00:00.0 enP30832s1: Moved VF to namespace with: eth0 [231.449656] BUG: kernel NULL pointer dereference, address: 0000000000000010 [231.450246] #PF: supervisor read access in kernel mode [231.450579] #PF: error_code(0x0000) - not-present page [231.450916] PGD 17b8a8067 P4D 0 [231.451163] Oops: Oops: 0000 [#1] SMP NOPTI [231.451450] CPU: 82 UID: 0 PID: 1394 Comm: kworker/u768:1 Not tainted 6.16.0-rc4+ #3 VOLUNTARY [231.452042] Hardware name: Microsoft Corporation Virtual Machine/Virtual Machine, BIOS Hyper-V UEFI Release v4.1 11/21/2024 [231.452692] Workqueue: netns cleanup_net [231.452947] RIP: 0010:default_device_exit_batch+0x16c/0x3f0 [231.453326] Code: c0 0c f5 b3 e8 d5 db fe ff 48 85 c0 74 15 48 c7 c2 f8 fd ca b2 be 10 00 00 00 48 8d 7d c0 e8 7b 77 25 00 49 8b 86 28 01 00 00 <48> 8b 50 10 4c 8b 2a 4c 8d 62 f0 49 83 ed 10 4c 39 e0 0f 84 d6 00 [231.454294] RSP: 0018:ff75fc7c9bf9fd00 EFLAGS: 00010246 [231.454610] RAX: 0000000000000000 RBX: 0000000000000002 RCX: 61c8864680b583eb [231.455094] RDX: ff1fa9f71462d800 RSI: ff75fc7c9bf9fd38 RDI: 0000000030766564 [231.455686] RBP: ff75fc7c9bf9fd78 R08: 0000000000000000 R09: 0000000000000000 [231.456126] R10: 0000000000000001 R11: 0000000000000004 R12: ff1fa9f70088e340 [231.456621] R13: ff1fa9f70088e340 R14: ffffffffb3f50c20 R15: ff1fa9f7103e6340 [231.457161] FS: 0000000000000000(0000) GS:ff1faa6783a08000(0000) knlGS:0000000000000000 [231.457707] CS: 0010 DS: 0000 ES: 0000 CR0: 0000000080050033 [231.458031] CR2: 0000000000000010 CR3: 0000000179ab2006 CR4: 0000000000b73ef0 [231.458434] Call Trace: [231.458600] [231.458777] ops_undo_list+0x100/0x220 [231.459015] cleanup_net+0x1b8/0x300 [231.459285] process_one_work+0x184/0x340 To fix it, move the ns change to a workqueue, and take rtnl_lock to avoid changing the netdev list when default_device_exit_net() is using it.
CVE-2025-38664	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ice: Fix a null pointer dereference in ice_copy_and_init_pkg() Add check for the return value of devm_kmemdup() to prevent potential null pointer dereference.
CVE-2025-38526	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ice: add NULL check in eswitch lag check The function ice_lag_is_switchdev_running() is being called from outside of the LAG event handler code. This results in the lag->upper_netdev being NULL sometimes. To avoid a NULL-pointer dereference, there needs to be a check before it is dereferenced.

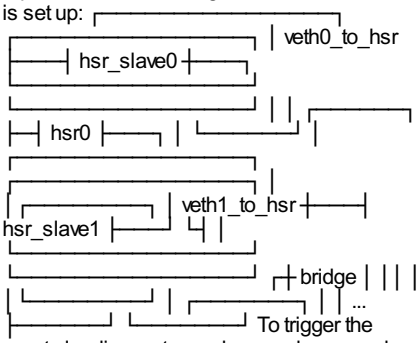
CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2024-42083	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ionic: fix kernel panic due to multi-buffer handling Currently, the ionic_run_xdp() doesn't handle multi-buffer packets properly for XDP_TX and XDP_REDIRECT. When a jumbo frame is received, the ionic_run_xdp() first makes xdp frame with all necessary pages in the rx descriptor. And if the action is either XDP_TX or XDP_REDIRECT, it should unmap dma-mapping and reset page pointer to NULL for all pages, not only the first page. But it doesn't for SG pages. So, SG pages unexpectedly will be reused. It eventually causes kernel panic. Oops: general protection fault, probably for non-canonical address 0x504f4e4dbebc64ff: 0000 [#1] PREEMPT SMP NOPTI CPU: 3 PID: 0 Comm: swapper/3 Not tainted 6.10.0-rc3+ #25 RIP: 0010:xdp_return_frame+0x42/0x90 Code: 01 75 12 5b 4c 89 e6 5d 31 c9 41 5c 31 d2 41 5d e9 73 fd ff ff 44 8b 6b 20 0f b7 43 0a 49 81 ed 68 01 00 00 49 29 c5 49 01 fd <41> 80 7d0 RSP: 0018:ffff99d00122ce08 EFLAGS: 00010202 RAX: 0000000000005453 RBX: ffff8d325f904000 RCX: 0000000000000001 RDX: 00000000670e1000 RSI: 000000011f90d000 RDI: 504f4e4d4c4b4a49 RBP: ffff99d003907740 R08: 0000000000000000 R09: 0000000000000000 R10: 000000011f90d000 R11: 0000000000000000 R12: ffff8d325f904010 R13: 504f4e4dbebc64fd R14: ffff8d3242b070c8 R15: ffff99d0039077c0 FS: 0000000000000000(0000) GS:ffff8d399f780000(0000) knlGS:0000000000000000 CS: 0010 DS: 0000 ES: 0000 CR0: 0000000080050033 CR2: 00007f41f6c85e38 CR3: 000000037ac30000 CR4: 00000000007506f0 PKRU: 55555554 Call Trace: ? die_addr+0x33/0x90 ? exc_general_protection+0x251/0x2f0 ? asm_exc_general_protection+0x22/0x30 ? xdp_return_frame+0x42/0x90 ionic_tx_clean+0x211/0x280 [ionic 15881354510e6a9c655c59c54812b319ed2cd015] ionic_tx_cq_service+0xd3/0x210 [ionic 15881354510e6a9c655c59c54812b319ed2cd015] ionic_txrx_napi+0x41/0x1b0 [ionic 15881354510e6a9c655c59c54812b319ed2cd015] __napi_poll.constprop.0+0x29/0x1b0 net_rx_action+0x2c4/0x350 handle_softirqs+0xf4/0x320 irq_exit_rcu+0x78/0xa0 common_interrupt+0x77/0x90

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-71098	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ip6_gre: make ip6gre_header() robust Over the years, syzbot found many ways to crash the kernel in ip6gre_header() [1]. This involves team or bonding drivers ability to dynamically change their dev->needed_headroom and/or dev->hard_header_len In this particular crash mld_newpack() allocated an skb with a too small reserve/headroom, and by the time mld_sendpack() was called, syzbot managed to attach an ip6gre device. [1] skbuff: skb_under_panic: text:ffffffff8a1d69a8 len:136 put:40 head:ffff888059bc7000 data:ffff888059bc6fe8 tail:0x70 end:0x6c0 dev:team0 ----------[cut here]---------- kernel BUG at net/core/skbuff.c:213 ! skb_under_panic net/core/skbuff.c:223 [inline] skb_push+0xc3/0xe0 net/core/skbuff.c:2641 ip6gre_header+0xc8/0x790 net/ipv6/ip6_gre.c:1371 dev_hard_header include/linux/netdevice.h:3436 [inline] neigh_connected_output+0x286/0x460 net/core/neighbour.c:1618 neigh_output include/net/neighbour.h:556 [inline] ip6_finish_output2+0xfb3/0x1480 net/ipv6/ip6_output.c:136 __ip6_finish_output net/ipv6/ip6_output.c:-1 [inline] ip6_finish_output+0x234/0x7d0 net/ipv6/ip6_output.c:220 NF_HOOK_COND include/linux/netfilter.h:307 [inline] ip6_output+0x340/0x550 net/ipv6/ip6_output.c:247 NF_HOOK+0x9e/0x380 include/linux/netfilter.h:318 mld_sendpack+0x8d4/0xe60 net/ipv6/mcast.c:1855 mld_send_cr net/ipv6/mcast.c:2154 [inline] mld_ifc_work+0x83e/0xd60 net/ipv6/mcast.c:2693
CVE-2025-38203	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: jfs: Fix null-ptr-deref in jfs_ioc_trim [Syzkaller Report] Oops: general protection fault, probably for non-canonical address 0xdffffc0000000087: 0000 [#1 KASAN: null-ptr-deref in range [0x0000000000000438-0x000000000000043f] CPU: 2 UID: 0 PID: 10614 Comm: syz-executor.0 Not tainted 6.13.0-rc6-gfbfd64d25c7a-dirty #1 Hardware name: QEMU Standard PC (i440FX + PIIX, 1996), BIOS 1.15.0-1 04/01/2014 Sched_ext: serialise (enabled+all), task: runnable_at=-30ms RIP: 0010:jfs_ioc_trim+0x34b/0x8f0 Code: e7 e8 59 a4 87 fe 4d 8b 24 24 4d 8d bc 24 38 04 00 00 48 8d 93 90 82 fe ff 4c 89 ff 31 f6 RSP: 0018:ffffc900055f7cd0 EFLAGS: 00010206 RAX: 0000000000000087 RBX: 00005866a9e67ff8 RCX: 000000000000000a RDX: 0000000000000001 RSI: 0000000000000004 RDI: 0000000000000001 RBP: dfffffc0000000000 R08: ffff88807c180003 R09: 1ffff1100f830000 R10: dfffffc0000000000 R11: ffffed100f830001 R12: 0000000000000000 R13: 0000000000000000 R14: 0000000000000001 R15: 0000000000000438 FS: 00007fe520225640(0000) GS:ffff8880b7e80000(0000) knlGS:0000000000000000 CS: 0010 DS: 0000 ES: 0000 CR0: 0000000080050033 CR2: 00005593c91b2c88 CR3: 000000014927c000 CR4: 00000000000006f0 DR0: 0000000000000000 DR1: 0000000000000000 DR2: 0000000000000000 DR3: 0000000000000000 DR6: 00000000fffe0ff0 DR7: 0000000000000400 Call Trace: ? __die_body+0x61/0xb0 ? die_addr+0xb1/0xe0 ? exc_general_protection+0x333/0x510 ? asm_exc_general_protection+0x26/0x30 ? jfs_ioc_trim+0x34b/0x8f0 jfs_ioctl+0x3c8/0x4f0 ? __pfx_jfs_ioctl+0x10/0x10 ? __pfx_jfs_ioctl+0x10/0x10 __se_sys_ioctl+0x269/0x350 ? __pfx___se_sys_ioctl+0x10/0x10 ? do_syscall_64+0xfb/0x210 do_syscall_64+0xee/0x210 ?

CVE	Severity	CVSS	CVSS Vector	Component	Description
					syscall_exit_to_user_mode+0x1e0/0x330 entry_SYSCALL_64_after_hwframe+0x77/0x7f RIP: 0033:0x7fe51f4903ad Code: c3 e8 a7 2b 00 00 0f 1f 80 00 00 00 00 f3 0f 1e fa 48 89 f8 48 89 f7 48 89 d6 48 89 ca 4d RSP: 002b:00007fe5202250c8 EFLAGS: 00000246 ORIG_RAX: 0000000000000010 RAX: ffffffffffffffda RBX: 00007fe51f5cbf80 RCX: 00007fe51f4903ad RDX: 0000000020000680 RSI: 00000000c0185879 RDI: 0000000000000005 RBP: 0000000000000000 R08: 0000000000000000 R09: 0000000000000000 R10: 0000000000000000 R11: 0000000000000246 R12: 00007fe520225640 R13: 000000000000000e R14: 00007fe51f44fca0 R15: 00007fe52021d000 Modules linked in: ---[end trace 0000000000000000]--- RIP: 0010:jfs_ioc_trim+0x34b/0x8f0 Code: e7 e8 59 a4 87 fe 4d 8b 24 24 4d 8d bc 24 38 04 00 00 48 8d 93 90 82 fe ff 4c 89 ff 31 f6 RSP: 0018:ffffc900055f7cd0 EFLAGS: 00010206 RAX: 0000000000000087 RBX: 00005866a9e67ff8 RCX: 000000000000000a RDX: 0000000000000001 RSI: 0000000000000004 RDI: 0000000000000001 RBP: dfffffc0000000000 R08: ffff88807c180003 R09: 1ffff1100f830000 R10: dfffffc0000000000 R11: ffffed100f830001 R12: 0000000000000000 R13: 0000000000000000 R14: 0000000000000001 R15: 0000000000000438 FS: 00007fe520225640(0000) GS:ffff8880b7e80000(0000) knlGS:0000000000000000 CS: 0010 DS: 0000 ES: 0000 CR0: 0000000080050033 CR2: 00005593c91b2c88 CR3: 000000014927c000 CR4: 00000000000006f0 DR0: 0000000000000000 DR1: 0000000000000000 DR2: 0000000000000000 DR3: 0000000000000000 DR6: 00000000fffe0ff0 DR7: 0000000000000400 Kernel panic - not syncing: Fatal exception [Analysis] We believe that we have found a concurrency bug in the `fs/jfs` module that results in a null pointer dereference. There is a closely related issue which has been fixed: https://git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux.git/commit/?id=d6c1b3599b2feb5c7291f5ac3a36e5fa7cedb234 ... but, unfortunately, the accepted patch appears to still be susceptible to a null pointer dereference under some interleavings. To trigger the bug, we think that `JFS_SBI(ipbmap->i_sb)->bmap` is set to NULL in `dbFreeBits` and then dereferenced in `jfs_ioc_trim`. This bug manifests quite rarely under normal circumstances, but is triggereable from a syz-program.
CVE-2025-38510	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: kasan: remove kasan_find_vm_area() to prevent possible deadlock find_vm_area() couldn't be called in atomic_context. If find_vm_area() is called to reports vm area information, kasan can trigger deadlock like: CPU0 CPU1 vmalloc(); alloc_vmap_area(); spin_lock(&vn->busy.lock) spin_lock_bh(&some_lock); spin_lock(&some_lock); kasan_report(); print_report(); print_address_description(); kasan_find_vm_area(); find_vm_area(); spin_lock(&vn->busy.lock) // deadlock! To prevent possible deadlock while kasan reports, remove kasan_find_vm_area().
CVE-2025-38562	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ksmbd: fix null pointer dereference error in generate_encryptionkey If client send two session setups with krb5 authenticate to ksmbd, null pointer dereference error in generate_encryptionkey could happen. sess->Preauth_HashValue is set to NULL if session is valid. So this patch skip generate encryption key if session is valid.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-22037	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ksmbd: fix null pointer dereference in alloc_preauth_hash() The Client send malformed smb2 negotiate request. ksmbd return error response. Subsequently, the client can send smb2 session setup even thought conn->preauth_info is not allocated. This patch add KSMBD_SESS_NEED_SETUP status of connection to ignore session setup request if smb2 negotiate phase is not complete.
CVE-2026-22991	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: libceph: make free_choose_arg_map() resilient to partial allocation free_choose_arg_map() may dereference a NULL pointer if its caller fails after a partial allocation. For example, in decode_choose_args(), if allocation of arg_map->args fails, execution jumps to the fail label and free_choose_arg_map() is called. Since arg_map->size is updated to a non-zero value before memory allocation, free_choose_arg_map() will iterate over arg_map->args and dereference a NULL pointer. To prevent this potential NULL pointer dereference and make free_choose_arg_map() more resilient, add checks for pointers before iterating.
CVE-2026-22992	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: libceph: return the handler error from mon_handle_auth_done() Currently any error from ceph_auth_handle_reply_done() is propagated via finish_auth() but isn't returned from mon_handle_auth_done(). This results in higher layers learning that (despite the monitor considering us to be successfully authenticated) something went wrong in the authentication phase and reacting accordingly, but msgr2 still trying to proceed with establishing the session in the background. In the case of secure mode this can trigger a WARN in setup_crypto() and later lead to a NULL pointer dereference inside of prepare_auth_signature().
CVE-2025-38364	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: maple_tree: fix MA_STATE_PREALLOC flag in mas_preallocate() Temporarily clear the preallocation flag when explicitly requesting allocations. Pre-existing allocations are already counted against the request through mas_node_count_gfp(), but the allocations will not happen if the MA_STATE_PREALLOC flag is set. This flag is meant to avoid re-allocating in bulk allocation mode, and to detect issues with preallocation calculations. The MA_STATE_PREALLOC flag should also always be set on zero allocations so that detection of underflow allocations will print a WARN_ON() during consumption. User visible effect of this flaw is a WARN_ON() followed by a null pointer dereference when subsequent requests for larger number of nodes is ignored, such as the vma merge retry in mmap_region() caused by drivers altering the vma flags (which happens in v6.6, at least)
CVE-2025-38694	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: media: dvb-frontends: dib7090p: fix null-ptr-deref in dib7090p_rw_on_apb() In dib7090p_rw_on_apb, msg is controlled by user. When msg[0].buf is null and msg[0].len is zero, former checks on msg[0].buf would be passed. If accessing msg[0].buf[2] without sanity check, null pointer deref would happen. We add check on msg[0].len to prevent crash. Similar issue occurs when access msg[1].buf[0] and msg[1].buf[1]. Similar commit: commit 0ed554fd769a ("media: dvb-usb: az6027: fix null-ptr-deref in az6027_i2c_xfer()")

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38693	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: media: dvb-frontends: w7090p: fix null-ptr-deref in w7090p_tuner_write_serpar and w7090p_tuner_read_serpar In w7090p_tuner_write_serpar, msg is controlled by user. When msg[0].buf is null and msg[0].len is zero, former checks on msg[0].buf would be passed. If accessing msg[0].buf[2] without sanity check, null pointer deref would happen. We add check on msg[0].len to prevent crash. Similar commit: commit 0ed554fd769a ("media: dvb-usb: az6027: fix null-ptr-deref in az6027_i2c_xfer()")
CVE-2025-39709	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: media: venus: protect against spurious interrupts during probe Make sure the interrupt handler is initialized before the interrupt is registered. If the IRQ is registered before hfi_create(), it's possible that an interrupt fires before the handler setup is complete, leading to a NULL dereference. This error condition has been observed during system boot on Rb3Gen2.
CVE-2025-39902	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: mm/slub: avoid accessing metadata when pointer is invalid in object_err() object_err() reports details of an object for further debugging, such as the freelist pointer, redzone, etc. However, if the pointer is invalid, attempting to access object metadata can lead to a crash since it does not point to a valid object. One known path to the crash is when alloc_consistency_checks() determines the pointer to the allocated object is invalid because of a freelist corruption, and calls object_err() to report it. The debug code should report and handle the corruption gracefully and not crash in the process. In case the pointer is NULL or check_valid_pointer() returns false for the pointer, only print the pointer value and skip accessing metadata.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-39703	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net, hsr: reject HSR frame if skb can't hold tag Receiving HSR frame with insufficient space to hold HSR tag in the skb can result in a crash (kernel BUG): [45.390915] skbuff: skb_under_panic: text:ffffffff86f32cac len:26 put:14 head:ffff888042418000 data:ffff888042417ff4 tail:0xe end:0x180 dev:bridge_slave_1 [45.392559] ------------[cut here]------------ [45.392912] kernel BUG at net/core/skbuff.c:211! [45.393276] Oops: invalid opcode: 0000 [#1] SMP DEBUG_PAGEALLOC KASAN NOPTI [45.393809] CPU: 1 UID: 0 PID: 2496 Comm: reproducer Not tainted 6.15.0 #12 PREEMPT(undef) [45.394433] Hardware name: QEMU Standard PC (i440FX + PIIX, 1996), BIOS rel-1.16.3-0-ga6ed6b701f0a-prebuilt.qemu.org 04/01/2014 [45.395273] RIP: 0010:skb_panic+0x15b/0x1d0 [45.402911] Call Trace: [45.403105] [45.404470] skb_push+0xcd/0xf0 [45.404726] br_dev_queue_push_xmit+0x7c/0x6c0 [45.406513] br_forward_finish+0x128/0x260 [45.408483] __br_forward+0x42d/0x590 [45.409464] maybe_deliver+0x2eb/0x420 [45.409763] br_flood+0x174/0x4a0 [45.410030] br_handle_frame_finish+0xc7c/0x1bc0 [45.411618] br_handle_frame+0xac3/0x1230 [45.413674] __netif_receive_skb_core.constprop.0+0x808/0x3df0 [45.422966] __netif_receive_skb_one_core+0xb4/0x1f0 [45.424478] __netif_receive_skb+0x22/0x170 [45.424806] process_backlog+0x242/0x6d0 [45.425116] __napi_poll+0xbb/0x630 [45.425394] net_rx_action+0x4d1/0xcc0 [45.427613] handle_softirqs+0x1a4/0x580 [45.427926] do_softirq+0x74/0x90 [45.428196] This issue was found by syzkaller. The panic happens in br_dev_queue_push_xmit() once it receives a corrupted skb with ETH header already pushed in linear data. When it attempts the skb_push() call, there's not enough headroom and skb_push() panics. The corrupted skb is put on the queue by HSR layer, which makes a sequence of unintended transformations when it receives a specific corrupted HSR frame (with incomplete TAG). Fix it by dropping and consuming frames that are not long enough to contain both ethernet and hsr headers. Alternative fix would be to check for enough headroom before skb_push() in br_dev_queue_push_xmit(). In the reproducer, this is injected via AF_PACKET, but I don't easily see why it couldn't be sent over the wire from adjacent network. Further Details: In the reproducer, the following network interface chain is set up:  To trigger the events leading up to crash, reproducer sends a corrupted HSR fr ---truncated---
CVE-2025-38645	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net/mlx5: Check device memory pointer before usage Add a NULL check before accessing device memory to prevent a crash if dev->dm allocation in mlx5_init_once() fails.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2026-22996	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net/mlx5e: Don't store mlx5e_priv in mlx5e_dev devlink priv mlx5e_priv is an unstable structure that can be memset(0) if profile attaching fails, mlx5e_priv in mlx5e_dev devlink private is used to reference the netdev and mdev associated with that struct. Instead, store netdev directly into mlx5e_dev and get mdev from the containing mlx5_adev aux device structure. This fixes a kernel oops in mlx5e_remove when switchdev mode fails due to change profile failure. $ devlink dev eswitch set pci/0000:00:03.0 mode switchdev Error: mlx5_core: Failed setting eswitch to offloads. dmesg: workqueue: Failed to create a rescuer kthread for wq "mlx5e": -EINTR mlx5_core 0012:03:00.1: mlx5e_netdev_init_profile:6214: (pid 37199): mlx5e_priv_init failed, err=-12 mlx5_core 0012:03:00.1 gpu3rdma1: mlx5e_netdev_change_profile: new profile init failed, -12 workqueue: Failed to create a rescuer kthread for wq "mlx5e": -EINTR mlx5_core 0012:03:00.1: mlx5e_netdev_init_profile:6214: (pid 37199): mlx5e_priv_init failed, err=-12 mlx5_core 0012:03:00.1 gpu3rdma1: mlx5e_netdev_change_profile: failed to rollback to orig profile, -12 $ devlink dev reload pci/0000:00:03.0 ==> oops BUG: kernel NULL pointer dereference, address: 0000000000000520 #PF: supervisor read access in kernel mode #PF: error_code(0x0000) - not-present page PGD 0 P4D 0 Oops: Oops: 0000 [#1] SMP NOPTI CPU: 3 UID: 0 PID: 521 Comm: devlink Not tainted 6.18.0-rc5+ #117 PREEMPT(voluntary) Hardware name: QEMU Standard PC (Q35 + ICH9, 2009), BIOS 1.16.3-2.fc40 04/01/2014 RIP: 0010:mlx5e_remove+0x68/0x130 RSP: 0018:ffffc900034838f0 EFLAGS: 00010246 RAX: ffff88810283c380 RBX: ffff888101874400 RCX: ffffffff826ffc45 RDX: 0000000000000000 RSI: 0000000000000001 RDI: 0000000000000000 RBP: ffff888102d789c0 R08: ffff8881007137f0 R09: ffff888100264e10 R10: ffffc90003483898 R11: ffffc900034838a0 R12: ffff888100d261a0 R13: ffff888100d261a0 R14: ffff8881018749a0 R15: ffff888101874400 FS: 00007f8565fea740(0000) GS:ffff88856a759000(0000) knlGS:0000000000000000 CS: 0010 DS: 0000 ES: 0000 CR0: 0000000080050033 CR2: 0000000000000520 CR3: 000000010b11a004 CR4: 0000000000370ef0 Call Trace: device_release_driver_internal+0x19c/0x200 bus_remove_device+0xc6/0x130 device_del+0x160/0x3d0 ? devl_param_driverinit_value_get+0x2d/0x90 mlx5_detach_device+0x89/0xe0 mlx5_unload_one_devl_locked+0x3a/0x70 mlx5_devlink_reload_down+0xc8/0x220 devlink_reload+0x7d/0x260 devlink_nl_reload_doit+0x45b/0x5a0 genl_family_rcv_msg_doit+0xe8/0x140
CVE-2026-23000	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net/mlx5e: Fix crash on profile change rollback failure mlx5e_netdev_change_profile can fail to attach a new profile and can fail to rollback to old profile, in such case, we could end up with a dangling netdev with a fully reset netdev_priv. A retry to change profile, e.g. another attempt to call mlx5e_netdev_change_profile via switchdev mode change, will crash trying to access the now NULL priv->mdev. This fix allows mlx5e_netdev_change_profile() to handle previous failures and an empty priv, by not assuming priv is valid. Pass netdev and mdev to all flows requiring mlx5e_netdev_change_profile() and avoid passing priv. In mlx5e_netdev_change_profile() check if current priv is valid, and if not, just attach the new profile without trying to access the old one. This fixes the following oops, when enabling switchdev mode for the 2nd time after first time

CVE	Severity	CVSS	CVSS Vector	Component	Description
					failure: ## Enabling switchdev mode first time: mlx5_core 0012:03:00.1: E-Switch: Supported tc chains and prios offload workqueue: Failed to create a rescuer kthread for wq "mlx5e": -EINTR mlx5_core 0012:03:00.1: mlx5e_netdev_init_profile:6214:(pid 37199): mlx5e_priv_init failed, err=-12 mlx5_core 0012:03:00.1 gpu3rdma1: mlx5e_netdev_change_profile: new profile init failed, -12 workqueue: Failed to create a rescuer kthread for wq "mlx5e": -EINTR mlx5_core 0012:03:00.1: mlx5e_netdev_init_profile:6214: (pid 37199): mlx5e_priv_init failed, err=-12 mlx5_core 0012:03:00.1 gpu3rdma1: mlx5e_netdev_change_profile: failed to rollback to orig profile, -12 ^^^^^^^ mlx5_core 0000:00:03.0: E-Switch: Disable: mode(LEGACY), nvfs(0), necvfs(0), active vports(0) ## retry. Enabling switchdev mode 2nd time: mlx5_core 0000:00:03.0: E-Switch: Supported tc chains and prios offload BUG: kernel NULL pointer dereference, address: 0000000000000038 #PF: supervisor read access in kernel mode #PF: error_code(0x0000) - not-present page PGD 0 P4D 0 Oops: Oops: 0000 [#1] SMP NOPTI CPU: 13 UID: 0 PID: 520 Comm: devlink Not tainted 6.18.0-rc4+ #91 PREEMPT(voluntary) Hardware name: QEMU Standard PC (Q35 + ICH9, 2009), BIOS 1.16.3-2.fc40 04/01/2014 RIP: 0010:mlx5e_detach_netdev+0x3c/0x90 Code: 50 00 00 f0 80 4f 78 02 48 8b bf e8 07 00 00 48 85 ff 74 16 48 8b 73 78 48 d1 ee 83 e6 01 83 f6 01 40 0f b6 f6 e8 c4 42 00 00 <48> 8b 45 38 48 85 c0 74 08 48 89 df e8 cc 47 40 1e 48 8b bb f0 07 RSP: 0018:ffffc90000673890 EFLAGS: 00010246 RAX: 0000000000000000 RBX: ffff8881036a89c0 RCX: 0000000000000000 RDX: ffff888113f63800 RSI: ffffffff822fe720 RDI: 0000000000000000 RBP: 0000000000000000 R08: 0000000000002dcd R09: 0000000000000000 R10: ffffc900006738e8 R11: 00000000ffffffff R12: 0000000000000000 R13: 0000000000000000 R14: ffff8881036a89c0 R15: 0000000000000000 FS: 00007fdfb8384740(0000) GS:ffff88856a9d6000(0000) knlGS:0000000000000000 CS: 0010 DS: 0000 ES: 0000 CR0: 0000000080050033 CR2: 0000000000000038 CR3: 0000000112ae0005 CR4: 0000000000370ef0 Call Trace: mlx5e_netdev_change_profile+0x45/0xb0 mlx5e_vport_rep_load+0x27b/0x2d0 mlx5_esw_offloads_rep_load+0x72/0xf0 esw_offloads_enable+0x5d0/0x970 mlx5_eswitch_enable_locked+0x349/0x430 ? is_mp_supported+0x57/0xb0 mlx5_devlink_eswitch_mode_set+0x26b/0x430 devlink_nl_eswitch_set_doit+0x6f/0xf0 genl_family_rcv_msg_doit+0xe8/0x140 genl_rcv_msg+0x18b/0x290 ? __pfx_devlink_nl_pre_doit+0x10/0x10 ? __pfx_devlink_nl_eswitch_set_doit+0x10/0x10 ? __pfx_devlink_nl_post_doit+0x10/0x10 ? __pfx_genl_rcv_msg+0x10/0x10 netlink_rcv_skb+0x52/0x100 genl_rcv+0x28/0x40 netlink_unicast+0x282/0x3e0 ? __alloc_skb+0xd6/0x190 netlink_sendmsg+0x1f7/0x430 __sys_sendto+0x213/0x220 ? __sys_recvmsg+0x6a/0xd0 __x64_sys_sendto+0x24/0x30 do_syscall_64+0x50/0x1f0 entry_SYSCALL_64_after_hwframe+0x76/0x7e RIP: 0033:0x7fdfb8495047

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-39947	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net/mlx5e: Harden uplink netdev access against device unbind The function mlx5_uplink_netdev_get() gets the uplink netdevice pointer from mdev->mlx5e_res.uplink_netdev. However, the netdevice can be removed and its pointer cleared when unbound from the mlx5_core.eth driver. This results in a NULL pointer, causing a kernel panic. BUG: unable to handle page fault for address: 0000000000001300 at RIP: 0010:mlx5e_vport_rep_load+0x22a/0x270 [mlx5_core] Call Trace: mlx5_esw_offloads_rep_load+0x68/0xe0 [mlx5_core] esw_offloads_enable+0x593/0x910 [mlx5_core] mlx5_eswitch_enable_locked+0x341/0x420 [mlx5_core] mlx5_devlink_eswitch_mode_set+0x17e/0x3a0 [mlx5_core] devlink_nl_eswitch_set_doit+0x60/0xd0 genl_family_rcv_msg_doit+0xe0/0x130 genl_rcv_msg+0x183/0x290 netlink_rcv_skb+0x4b/0xf0 genl_rcv+0x24/0x40 netlink_unicast+0x255/0x380 netlink_sendmsg+0x1f3/0x420 __sock_sendmsg+0x38/0x60 __sys_sendto+0x119/0x180 do_syscall_64+0x53/0x1d0 entry_SYSCALL_64_after_hwframe+0x4b/0x53 Ensure the pointer is valid before use by checking it for NULL. If it is valid, immediately call netdev_hold() to take a reference, and preventing the netdevice from being freed while it is in use.
CVE-2026-23173	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net/mlx5e: TC, delete flows only for existing peers When deleting TC steering flows, iterate only over actual devcom peers instead of assuming all possible ports exist. This avoids touching non-existent peers and ensures cleanup is limited to devices the driver is currently connected to. BUG: kernel NULL pointer dereference, address: 0000000000000008 #PF: supervisor write access in kernel mode #PF: error_code(0x0002) - not-present page PGD 133c8a067 P4D 0 Oops: Oops: 0002 [#1] SMP CPU: 19 UID: 0 PID: 2169 Comm: tc Not tainted 6.18.0+ #156 NONE Hardware name: QEMU Standard PC (Q35 + ICH9, 2009), BIOS rel-1.16.0-0-gd239552ce722-prebuilt.qemu.org 04/01/2014 RIP: 0010:mlx5e_tc_del_fdb_peers_flow+0xbe/0x200 [mlx5_core] Code: 00 00 a8 08 74 a8 49 8b 46 18 f6 c4 02 74 9f 4c 8d bf a0 12 00 00 4c 89 ff e8 0e e7 96 e1 49 8b 44 24 08 49 8b 0c 24 4c 89 ff <48> 89 41 08 48 89 08 49 89 2c 24 49 89 5c 24 08 e8 7d ce 96 e1 49 RSP: 0018:ff11000143867528 EFLAGS: 00010246 RAX: 0000000000000000 RBX: dead000000000122 RCX: 0000000000000000 RDX: ff11000143691580 RSI: ff110001026e5000 RDI: ff11000106f3d2a0 RBP: dead000000000100 R08: 00000000000003fd R09: 0000000000000002 R10: ff11000101c75690 R11: ff1100085faea178 R12: ff11000115f0ae78 R13: 0000000000000000 R14: ff11000115f0a800 R15: ff11000106f3d2a0 FS: 00007f35236bf740(0000) GS:ff110008dc809000(0000) knlGS:0000000000000000 CS: 0010 DS: 0000 ES: 0000 CR0: 0000000080050033 CR2: 0000000000000008 CR3: 0000000157a01001 CR4: 0000000000373eb0 Call Trace: mlx5e_tc_del_flow+0x46/0x270 [mlx5_core] mlx5e_flow_put+0x25/0x50 [mlx5_core] mlx5e_delete_flower+0x2a6/0x3e0 [mlx5_core] tc_setup_cb_reoffload+0x20/0x80 fl_reoffload+0x26f/0x2f0 [cls_flower] ? mlx5e_tc_reoffload_flows_work+0xc0/0xc0 [mlx5_core] ? mlx5e_tc_reoffload_flows_work+0xc0/0xc0 [mlx5_core] tcf_block_playback_offloads+0x9e/0x1c0 tcf_block_unbind+0x7b/0xd0

CVE	Severity	CVSS	CVSS Vector	Component	Description
					tcf_block_setup+0x186/0x1d0 tcf_block_offload_cmd.isra.0+0xef/0x130 tcf_block_offload_unbind+0x43/0x70 __tcf_block_put+0x85/0x160 ingress_destroy+0x32/0x110 [sch_ingress] __qdisc_destroy+0x44/0x100 qdisc_graft+0x22b/0x610 tc_get_qdisc+0x183/0x4d0 rtnetlink_rcv_msg+0x2d7/0x3d0 ? rtnl_calcit.isra.0+0x100/0x100 netlink_rcv_skb+0x53/0x100 netlink_unicast+0x249/0x320 ? __alloc_skb+0x102/0x1f0 netlink_sendmsg+0x1e3/0x420 __sock_sendmsg+0x38/0x60 ____sys_sendmsg+0x1ef/0x230 ? copy_msghdr_from_user+0x6c/0xa0 ___sys_sendmsg+0x7f/0xc0 ? ___sys_recvmsg+0x8a/0xc0 ? __sys_sendto+0x119/0x180 __sys_sendmsg+0x61/0xb0 do_syscall_64+0x55/0x640 entry_SYSCALL_64_after_hwframe+0x4b/0x53 RIP: 0033:0x7f35238bb764 Code: 15 b9 86 0c 00 f7 d8 64 89 02 b8 ff ff ff ff eb bf 0f 1f 44 00 00 f3 0f 1e fa 80 3d e5 08 0d 00 00 74 13 b8 2e 00 00 00 0f 05 <48> 3d 00 f0 ff ff 77 4c c3 0f 1f 00 55 48 89 e5 48 83 ec 20 89 55 RSP: 002b:00007ffed4c35638 EFLAGS: 00000202 ORIG_RAX: 000000000000002e RAX: ffffffffffffffda RBX: 000055a2efcc75e0 RCX: 00007f35238bb764 RDX: 0000000000000000 RSI: 00007ffed4c356a0 RDI: 0000000000000003 RBP: 00007ffed4c35710 R08: 0000000000000010 R09: 00007f3523984b20 R10: 0000000000000004 R11: 0000000000000202 R12: 00007ffed4c35790 R13: 000000006947df8f R14: 000055a2efcc75e0 R15: 00007ffed4c35780

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38468	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net/sched: Return NULL when htb_lookup_leaf encounters an empty rbtree htb_lookup_leaf has a BUG_ON that can trigger with the following: tc qdisc del dev lo root tc qdisc add dev lo root handle 1: htb default 1 tc class add dev lo parent 1: classid 1:1 htb rate 64bit tc qdisc add dev lo parent 1:1 handle 2: netem tc qdisc add dev lo parent 2:1 handle 3: blackhole ping -I lo -c1 -W0.001 127.0.0.1 The root cause is the following: 1. htb_dequeue calls htb_dequeue_tree which calls the dequeue handler on the selected leaf qdisc 2. netem_dequeue calls enqueue on the child qdisc 3. blackhole_enqueue drops the packet and returns a value that is not just NET_XMIT_SUCCESS 4. Because of this, netem_dequeue calls qdisc_tree_reduce_backlog, and since qlen is now 0, it calls htb_qlen_notify -> htb_deactivate -> htb_deactiviate_prios -> htb_remove_class_from_row -> htb_safe_rb_erase 5. As this is the only class in the selected hprio rbtree, __rb_change_child in __rb_erase_augmented sets the rb_root pointer to NULL 6. Because blackhole_dequeue returns NULL, netem_dequeue returns NULL, which causes htb_dequeue_tree to call htb_lookup_leaf with the same hprio rbtree, and fail the BUG_ON The function graph for this scenario is shown here: 0) | htb_enqueue() { 0) + 13.635 us | netem_enqueue(); 0) 4.719 us | htb_activate_prios(); 0) # 2249.199 us | } 0) | htb_dequeue() { 0) 2.355 us | htb_lookup_leaf(); 0) | netem_dequeue() { 0) + 11.061 us | blackhole_enqueue(); 0) | qdisc_tree_reduce_backlog() { 0) | qdisc_lookup_rcu() { 0) 1.873 us | qdisc_match_from_root(); 0) 6.292 us | } 0) 1.894 us | htb_search(); 0) | htb_qlen_notify() { 0) 2.655 us | htb_deactivate_prios(); 0) 6.933 us | } 0) + 25.227 us | } 0) 1.983 us | blackhole_dequeue(); 0) + 86.553 us | } 0) # 2932.761 us | qdisc_warn_nonwc(); 0) | htb_lookup_leaf() { 0) | BUG_ON(); ---------- The full original bug report can be seen here [1]. We can fix this just by returning NULL instead of the BUG_ON, as htb_dequeue_tree returns NULL when htb_lookup_leaf returns NULL. [1] https://lore.kernel.org/netdev/pF5XOOlim0IuEfhI-SOxTgRvNoDwuux7UHKnE_Y5-zVd4wmGvNk2ceHjKb8ORnzw0cGwfmVu42g9dL7XyJLf1NEzaztboTWcm0Ogxuojoeo=@willsroot.io/

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2026-23064	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net/sched: act_ife: avoid possible NULL deref tcf_ife_encode() must make sure ife_encode() does not return NULL. syzbot reported: Oops: general protection fault, probably for non-canonical address 0xdfffffc0000000000: 0000 [#1] SMP KASAN NOPTI KASAN: null-ptr-deref in range [0x0000000000000000-0x0000000000000007] RIP: 0010:ife_tlv_meta_encode+0x41/0xa0 net/ife/ife.c:166 CPU: 3 UID: 0 PID: 8990 Comm: syz.0.696 Not tainted syzkaller #0 PREEMPT(full) Call Trace: ife_encode_meta_u32+0x153/0x180 net/sched/act_ife.c:101 tcf_ife_encode net/sched/act_ife.c:841 [inline] tcf_ife_act+0x1022/0x1de0 net/sched/act_ife.c:877 tc_act include/net/tc_wrapper.h:130 [inline] tcf_action_exec+0x1c0/0xa20 net/sched/act_api.c:1152 tcf_exts_exec include/net/pkt_cls.h:349 [inline] mall_classify+0x1a0/0x2a0 net/sched/cls_matchall.c:42 tc_classify include/net/tc_wrapper.h:197 [inline] __tcf_classify net/sched/cls_api.c:1764 [inline] tcf_classify+0x7f2/0x1380 net/sched/cls_api.c:1860 multiq_classify net/sched/sch_multiq.c:39 [inline] multiq_enqueue+0xe0/0x510 net/sched/sch_multiq.c:66 dev_qdisc_enqueue+0x45/0x250 net/core/dev.c:4147 __dev_xmit_skb net/core/dev.c:4262 [inline] __dev_queue_xmit+0x2998/0x46c0 net/core/dev.c:4798
CVE-2025-38684	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net/sched: ets: use old 'nbands' while purging unused classes Shuang reported sch_ets test-case [1] crashing in ets_class_qlen_notify() after recent changes from Lion [2]. The problem is: in ets_qdisc_change() we purge unused DWRR queues; the value of 'q->nbands' is the new one, and the cleanup should be done with the old one. The problem is here since my first attempts to fix ets_qdisc_change(), but it surfaced again after the recent qdisc len accounting fixes. Fix it purging idle DWRR queues before assigning a new value of 'q->nbands', so that all purge operations find a consistent configuration: - old 'q->nbands' because it's needed by ets_class_find() - old 'q->nstrict' because it's needed by ets_class_is_strict() BUG: kernel NULL pointer dereference, address: 0000000000000000 #PF: supervisor read access in kernel mode #PF: error_code(0x0000) - not-present page PGD 0 P4D 0 Oops: Oops: 0000 [#1] SMP NOPTI CPU: 62 UID: 0 PID: 39457 Comm: tc Kdump: loaded Not tainted 6.12.0-116.el10.x86_64 #1 PREEMPT(voluntary) Hardware name: Dell Inc. PowerEdge R640/06DKY5, BIOS 2.12.2 07/09/2021 RIP: 0010:__list_del_entry_valid_or_report+0x4/0x80 Code: ff 4c 39 c7 0f 84 39 19 8e ff b8 01 00 00 00 c3 cc cc cc cc 66 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 f3 0f 1e fa <48> 8b 17 48 8b 4f 08 48 85 d2 0f 84 56 19 8e ff 48 85 c9 0f 84 ab RSP: 0018:ffffba186009f400 EFLAGS: 00010202 RAX: 00000000000000d6 RBX: 0000000000000000 RCX: 0000000000000004 RDX: ffff9f0fa29b69c0 RSI: 0000000000000000 RDI: 0000000000000000 RBP: ffffffffc12c2400 R08: 0000000000000008 R09: 0000000000000004 R10: ffffffffffffffff R11: 0000000000000004 R12: 0000000000000000 R13: ffff9f0f8cfe0000 R14: 0000000000100005 R15: 0000000000000000 FS: 00007f2154f37480(0000) GS:ffff9f269c1c0000(0000) knlGS:0000000000000000 CS: 0010 DS: 0000 ES: 0000 CR0: 0000000080050033 CR2: 0000000000000000 CR3: 00000001530be001 CR4: 00000000007726f0 DR0: 0000000000000000 DR1: 0000000000000000

CVE	Severity	CVSS	CVSS Vector	Component	Description
					DR2: 0000000000000000 DR3: 0000000000000000 DR6: 00000000fffe0ff0 DR7: 0000000000000400 PKRU: 55555554 Call Trace: ets_class_qlen_notify+0x65/0x90 [sch_ets] qdisc_tree_reduce_backlog+0x74/0x110 ets_qdisc_change+0x630/0xa40 [sch_ets] __tc_modify_qdisc.constprop.0+0x216/0x7f0 tc_modify_qdisc+0x7c/0x120 rtnetlink_rcv_msg+0x145/0x3f0 netlink_rcv_skb+0x53/0x100 netlink_unicast+0x245/0x390 netlink_sendmsg+0x21b/0x470 ____sys_sendmsg+0x39d/0x3d0 ___sys_sendmsg+0x9a/0xe0 __sys_sendmsg+0x7a/0xd0 do_syscall_64+0x7d/0x160 entry_SYSCALL_64_after_hwframe+0x76/0x7e RIP: 0033:0x7f2155114084 Code: 89 02 b8 ff ff ff ff eb bb 66 2e 0f 1f 84 00 00 00 00 00 0f 1f 00 f3 0f 1e fa 80 3d 25 f0 0c 00 00 74 13 b8 2e 00 00 00 0f 05 <48> 3d 00 f0 ff ff 77 54 c3 0f 1f 00 48 83 ec 28 89 54 24 1c 48 89 RSP: 002b:00007fff1fd7a988 EFLAGS: 00000202 ORIG_RAX: 000000000000002e RAX: ffffffffffffffda RBX: 0000560ec063e5e0 RCX: 00007f2155114084 RDX: 0000000000000000 RSI: 00007fff1fd7a9f0 RDI: 0000000000000003 RBP: 00007fff1fd7aa60 R08: 0000000000000010 R09: 000000000000003f R10: 0000560ee9b3a010 R11: 0000000000000202 R12: 00007fff1fd7aae0 R13: 000000006891ccde R14: 0000560ec063e5e0 R15: 00007fff1fd7aad0 [1] https://lore.kernel.org/netdev/e08c7f4a6882f260011909a868311c6e9b54f3e4.1639153474.git.dcaratti@redhat.com/ [2] https://lore.kernel.org/netdev/d912cbd7-193b-4269-9857-525bee8bbb6a@gmail.com/
CVE-2026-22976	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net/sched: sch_qfq: Fix NULL deref when deactivating inactive aggregate in qfq_reset `qfq_class->leaf_qdisc->q.qlen > 0` does not imply that the class itself is active. Two qfq_class objects may point to the same leaf_qdisc. This happens when: 1. one QFQ qdisc is attached to the dev as the root qdisc, and 2. another QFQ qdisc is temporarily referenced (e.g., via qdisc_get() / qdisc_put()) and is pending to be destroyed, as in function tc_new_tfilter. When packets are enqueued through the root QFQ qdisc, the shared leaf_qdisc->q.qlen increases. At the same time, the second QFQ qdisc triggers qdisc_put and qdisc_destroy: the qdisc enters qfq_reset() with its own q->q.qlen == 0, but its class's leaf qdisc->q.qlen > 0. Therefore, the qfq_reset would wrongly deactivate an inactive aggregate and trigger a null-deref in qfq_deactivate_agg: [0.903172] BUG: kernel NULL pointer dereference, address: 0000000000000000 [0.903571] #PF: supervisor write access in kernel mode [0.903860] #PF: error_code(0x0002) - not-present page [0.904177] PGD 10299b067 P4D 10299b067 PUD 10299c067 PMD 0 [0.904502] Oops: Oops: 0002 [#1] SMP NOPTI [0.904737] CPU: 0 UID: 0 PID: 135 Comm: exploit Not tainted 6.19.0-rc3+ #2 NONE [0.905157] Hardware name: QEMU Standard PC (i440FX + PIIX, 1996), BIOS rel-1.17.0-0-gb52ca86e094d-prebuilt.qemu.org 04/01/2014 [0.905754] RIP: 0010:qfq_deactivate_agg (include/linux/list.h:992 (discriminator 2) include/linux/list.h:1006 (discriminator 2) net/sched/sch_qfq.c:1367 (discriminator 2) net/sched/sch_qfq.c:1393 (discriminator 2)) [0.906046] Code: 0f 84 4d 01 00 00 48 89 70 18 8b 4b 10 48 c7 c2 ff ff ff ff 48 8b 78 08 48 d3 e2 48 21 f2 48 2b 13 48 8b 30 48 d3 ea 8b 4b 18 0 Code starting with the faulting instruction ========== 0: 0f 84 4d 01 00 00 je 0x153 6: 48 89 70 18 mov %rsi,0x18(%rax) a: 8b 4b 10 mov 0x10(%rbx),%ecx d: 48 c7 c2 ff ff ff ff mov $0xffffffffffffffff,%rdx 14: 48 8b 78 08 mov 0x8(%rax),%rdi 18: 48 d3 e2 shl %cl,%rdx 1b: 48

CVE	Severity	CVSS	CVSS Vector	Component	Description
					21 f2 and %rsi,%rdx 1e: 48 2b 13 sub (%rbx),%rdx 21: 48 8b 30 mov (%rax),%rsi 24: 48 d3 ea shr %cl,%rdx 27: 8b 4b 18 mov 0x18(%rbx),%ecx ... [0.907095] RSP: 0018:ffffc900004a39a0 EFLAGS: 00010246 [0.907368] RAX: ffff8881043a0880 RBX: ffff888102953340 RCX: 0000000000000000 [0.907723] RDX: 0000000000000000 RSI: 0000000000000000 RDI: 0000000000000000 [0.908100] RBP: ffff888102952180 R08: 0000000000000000 R09: 0000000000000000 [0.908451] R10: ffff8881043a0000 R11: 0000000000000000 R12: ffff888102952000 [0.908804] R13: ffff888102952180 R14: ffff8881043a0ad8 R15: ffff8881043a0880 [0.909179] FS: 000000002a1a0380(0000) GS:ffff888196d8d000(0000) knlGS:0000000000000000 [0.909572] CS: 0010 DS: 0000 ES: 0000 CR0: 0000000080050033 [0.909857] CR2: 0000000000000000 CR3: 0000000102993002 CR4: 0000000000772ef0 [0.910247] PKRU: 55555554 [0.910391] Call Trace: [0.910527] [0.910638] qfq_reset_qdisc (net/sched/sch_qfq.c:357 net/sched/sch_qfq.c:1485) [0.910826] qdisc_reset (include/linux/skbuff.h:2195 include/linux/skbuff.h:2501 include/linux/skbuff.h:3424 include/linux/skbuff.h:3430 net/sched/sch_generic.c:1036) [0.911040] __qdisc_destroy (net/sched/sch_generic.c:1076) [0.911236] tc_new_tfilter (net/sched/cls_api.c:2447) [0.911447] rtnetlink_rcv_msg (net/core/rtnetlink.c:6958) [0.911663] ? __pfx_rtnetlink_rcv_msg (net/core/rtnetlink.c:6861) [0.911894] netlink_rcv_skb (net/netlink/af_netlink.c:2550) [0.912100] netlink_unicast (net/netlink/af_netlink.c:1319 net/netlink/af_netlink.c:1344) [0.912296] ? __alloc_skb (net/core/skbuff.c:706) [0.912484] netlink_sendmsg (net/netlink/af ---truncated---
CVE-2025-39857	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net/smc: fix one NULL pointer dereference in smc_ib_is_sg_need_sync() BUG: kernel NULL pointer dereference, address: 00000000000002ec PGD 0 P4D 0 Oops: Oops: 0000 [#1] SMP PTI CPU: 28 UID: 0 PID: 343 Comm: kworker/28:1 Kdump: loaded Tainted: G OE 6.17.0-rc2+ #9 NONE Tainted: [O]=OOT_MODULE, [E]=UNSIGNED_MODULE Hardware name: QEMU Standard PC (Q35 + ICH9, 2009), BIOS 1.15.0-1 04/01/2014 Workqueue: smc_hs_wq smc_listen_work [smc] RIP: 0010:smc_ib_is_sg_need_sync+0x9e/0xd0 [smc] ... Call Trace: smcr_buf_map_link+0x211/0x2a0 [smc] __smc_buf_create+0x522/0x970 [smc] smc_buf_create+0x3a/0x110 [smc] smc_find_rdma_v2_device_serv+0x18f/0x240 [smc] ? smc_vlan_by_tcpsk+0x7e/0xe0 [smc] smc_listen_find_device+0x1dd/0x2b0 [smc] smc_listen_work+0x30f/0x580 [smc] process_one_work+0x18c/0x340 worker_thread+0x242/0x360 kthread+0xe7/0x220 ret_from_fork+0x13a/0x160 ret_from_fork_asm+0x1a/0x30 If the software RoCE device is used, ibdev->dma_device is a null pointer. As a result, the problem occurs. Null pointer detection is added to prevent problems.
CVE-2026-23020	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net: 3com: 3c59x fix possible null dereference in vortex_probe1() pdev can be null and free_ring: can be called in 1297 with a null pdev.
CVE-2025-23143	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net: Fix null-ptr-deref by sock_lock_init_class_and_name() and rmmod. When I ran the repro [0] and waited a few seconds, I observed two LOCKDEP splats: a warning immediately followed by a null-ptr-deref. [1] Reproduction Steps: 1) Mount CIFS 2) Add an iptables rule to drop incoming FIN packets for CIFS 3) Unmount CIFS 4) Unload the CIFS

CVE	Severity	CVSS	CVSS Vector	Component	Description
					module 5) Remove the iptables rule At step 3), the CIFS module calls sock_release() for the underlying TCP socket, and it returns quickly. However, the socket remains in FIN_WAIT_1 because incoming FIN packets are dropped. At this point, the module's refcnt is 0 while the socket is still alive, so the following rmmod command succeeds. # ss -tan State Recv-Q Send-Q Local Address:Port Peer Address:Port FIN-WAIT-1 0 477 10.0.2.15:51062 10.0.0.137:445 # lsmod | grep cifs cifs 1159168 0 This highlights a discrepancy between the lifetime of the CIFS module and the underlying TCP socket. Even after CIFS calls sock_release() and it returns, the TCP socket does not die immediately in order to close the connection gracefully. While this is generally fine, it causes an issue with LOCKDEP because CIFS assigns a different lock class to the TCP socket's sk->sk_lock using sock_lock_init_class_and_name(). Once an incoming packet is processed for the socket or a timer fires, sk->sk_lock is acquired. Then, LOCKDEP checks the lock context in check_wait_context(), where hlock_class() is called to retrieve the lock class. However, since the module has already been unloaded, hlock_class() logs a warning and returns NULL, triggering the null-ptr-deref. If LOCKDEP is enabled, we must ensure that a module calling sock_lock_init_class_and_name() (CIFS, NFS, etc) cannot be unloaded while such a socket is still alive to prevent this issue. Let's hold the module reference in sock_lock_init_class_and_name() and release it when the socket is freed in sk_prot_free(). Note that sock_lock_init() clears sk->sk_owner for svc_create_socket() that calls sock_lock_init_class_and_name() for a listening socket, which clones a socket by sk_clone_lock() without GFP_ZERO. [0]: CIFS_SERVER="10.0.0.137" CIFS_PATH="//${CIFS_SERVER}/Users/Administrator/Desktop/CIFS_TEST" DEV="enp0s3" CRED="/root/WindowsCredential.txt" MNT=$(mktemp -d /tmp/XXXXXX) mount -t cifs ${CIFS_PATH} ${MNT} -o vers=3.0,credentials=${CRED},cache=none,echo_interval=1 iptables -A INPUT -s ${CIFS_SERVER} -j DROP for i in $(seq 10); do umount ${MNT} rmmod cifs sleep 1 done rm -r ${MNT} iptables -D INPUT -s ${CIFS_SERVER} -j DROP [1]: DEBUG_LOCKS_WARN_ON(1) WARNING: CPU: 10 PID: 0 at kernel/locking/lockdep.c:234 hlock_class (kernel/locking/lockdep.c:234 kernel/locking/lockdep.c:223) Modules linked in: cifs_arc4 nls_ucs2_utils cifs_md4 [last unloaded: cifs] CPU: 10 UID: 0 PID: 0 Comm: swapper/10 Not tainted 6.14.0 #36 Hardware name: QEMU Standard PC (i440FX + PIIX, 1996), BIOS rel-1.16.0-0-gd239552ce722-prebuilt.qemu.org 04/01/2014 RIP: 0010:hlock_class (kernel/locking/lockdep.c:234 kernel/locking/lockdep.c:223) ... Call Trace: __lock_acquire (kernel/locking/lockdep.c:4853 kernel/locking/lockdep.c:5178) lock_acquire (kernel/locking/lockdep.c:469 kernel/locking/lockdep.c:5853 kernel/locking/lockdep.c:5816) _raw_spin_lock_nested (kernel/locking/spinlock.c:379) tcp_v4_rcv (./include/linux/skbuff.h:1678 ./include/net/tcp.h:2547 net/ipv4/tcp_ipv4.c:2350) ... BUG: kernel NULL pointer dereference, address: 00000000000000c4 PF: supervisor read access in kernel mode PF: error_code(0x0000) - not-present page PGD 0 Oops: Oops: 0000 [#1] PREEMPT SMP NOPTI CPU: 10 UID: 0 PID: 0 Comm: swapper/10 Tainted: G W 6.14.0 #36 Tainted: [W]=WARN Hardware name: QEMU Standard PC (i440FX + PIIX, 1996), BIOS rel-1.16.0-0-gd239552ce722-prebuilt.qemu.org 04/01/2014 RIP: 0010:__lock_acquire (kernel/ ---truncated---

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-39876	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net: fec: Fix possible NPD in fec_enet_phy_reset_after_clk_enable() The function of_phy_find_device may return NULL, so we need to take care before dereferencing phy_dev.
CVE-2025-22103	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net: fix NULL pointer dereference in l3mdev_l3_rcv When delete l3s ipvlan: ip link del link eth0 ipvlan1 type ipvlan mode l3s This may cause a null pointer dereference: Call trace: ip_rcv_finish+0x48/0xd0 ip_rcv+0x5c/0x100 __netif_receive_skb_one_core+0x64/0xb0 __netif_receive_skb+0x20/0x80 process_backlog+0xb4/0x204 napi_poll+0xe8/0x294 net_rx_action+0xd8/0x22c __do_softirq+0x12c/0x354 This is because l3mdev_l3_rcv() visit dev->l3mdev_ops after ipvlan_l3s_unregister() assign the dev->l3mdev_ops to NULL. The process like this: (CPU1) | (CPU2) l3mdev_l3_rcv() | check dev->priv_flags: | master = skb->dev; | | ipvlan_l3s_unregister() | set dev->priv_flags | dev->l3mdev_ops = NULL; | visit master->l3mdev_ops | To avoid this by do not set dev->l3mdev_ops when unregister l3s ipvlan.
CVE-2026-23019	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net: marvell: prestera: fix NULL dereference on devlink_alloc() failure devlink_alloc() may return NULL on allocation failure, but prestera_devlink_alloc() unconditionally calls devlink_priv() on the returned pointer. This leads to a NULL pointer dereference if devlink allocation fails. Add a check for a NULL devlink pointer and return NULL early to avoid the crash.
CVE-2026-22982	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net: mscc: ocelot: Fix crash when adding interface under a lag Commit 15faa1f67ab4 ("lan966x: Fix crash when adding interface under a lag") fixed a similar issue in the lan966x driver caused by a NULL pointer dereference. The ocelot_set_aggr_pgids() function in the ocelot driver has similar logic and is susceptible to the same crash. This issue specifically affects the ocelot_vsc7514.c frontend, which leaves unused ports as NULL pointers. The felix_vsc9959.c frontend is unaffected as it uses the DSA framework which registers all ports. Fix this by checking if the port pointer is valid before accessing it.
CVE-2025-37945	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net: phy: allow MDIO bus PM ops to start/stop state machine for phylink-controlled PHY DSA has 2 kinds of drivers: 1. Those who call dsa_switch_suspend() and dsa_switch_resume() from their device PM ops: qca8k-8xxx, bcm_sf2, microchip ksz 2. Those who don't: all others. The above methods should be optional. For type 1, dsa_switch_suspend() calls dsa_user_suspend() -> phylink_stop(), and dsa_switch_resume() calls dsa_user_resume() -> phylink_start(). These seem good candidates for setting mac_managed_pm = true because that is essentially its definition [1], but that does not seem to be the biggest problem for now, and is not what this change focuses on. Talking strictly about the 2nd category of DSA drivers here (which do not have MAC managed PM, meaning that for their attached PHYs, mdio_bus_phy_suspend() and mdio_bus_phy_resume() should run in full), I have noticed that the following warning from mdio_bus_phy_resume() is triggered: WARN_ON(phydev->state != PHY_HALTED && phydev->state != PHY_READY && phydev->state != PHY_UP); because the PHY state machine is running. It's running as a result of a previous dsa_user_open() -> ... -> phylink_start() -> phy_start() having been initiated by the user. The previous mdio_bus_phy_suspend() was

CVE	Severity	CVSS	CVSS Vector	Component	Description
					supposed to have called phy_stop_machine(), but it didn't. So this is why the PHY is in state PHY_NOLINK by the time mdio_bus_phy_resume() runs. mdio_bus_phy_suspend() did not call phy_stop_machine() because for phylink, the phydev->adjust_link function pointer is NULL. This seems a technicality introduced by commit fddd91016d16 ("phylib: fix PAL state machine restart on resume"). That commit was written before phylink existed, and was intended to avoid crashing with consumer drivers which don't use the PHY state machine - phylink always does, when using a PHY. But phylink itself has historically not been developed with suspend/resume in mind, and apparently not tested too much in that scenario, allowing this bug to exist unnoticed for so long. Plus, prior to the WARN_ON(), it would have likely been invisible. This issue is not in fact restricted to type 2 DSA drivers (according to the above ad-hoc classification), but can be extrapolated to any MAC driver with phylink and MDIO-bus-managed PHY PM ops. DSA is just where the issue was reported. Assuming mac_managed_pm is set correctly, a quick search indicates the following other drivers might be affected: $ grep -Zlr PHYLINK_NETDEV drivers/ | xargs -0 grep -L mac_managed_pm drivers/net/ethernet/atheros/ag71xx.c drivers/net/ethernet/microchip/sparx5/sparx5_main.c drivers/net/ethernet/microchip/lan966x/lan966x_main.c drivers/net/ethernet/freescale/dpaa2/dpaa2-mac.c drivers/net/ethernet/freescale/fs_enet/fs_enet-main.c drivers/net/ethernet/freescale/dpaa/dpaa_eth.c drivers/net/ethernet/freescale/ucc_geth.c drivers/net/ethernet/freescale/enetc/enetc_pf_common.c drivers/net/ethernet/marvell/mvpp2/mvpp2_main.c drivers/net/ethernet/marvell/mvneta.c drivers/net/ethernet/marvell/prestera/prestera_main.c drivers/net/ethernet/mediatek/mtk_eth_soc.c drivers/net/ethernet/altera/altera_tse_main.c drivers/net/ethernet/wangxun/txgbe/txgbe_phy.c drivers/net/ethernet/meta/fbnic/fbnic_phylink.c drivers/net/ethernet/tehuti/tn40_phy.c drivers/net/ethernet/mscc/ocelot_net.c Make the existing conditions dependent on the PHY device having a phydev->phy_link_change() implementation equal to the default phy_link_change() provided by phylib. Otherwise, we implicitly know that the phydev has the phylink-provided phylink_phy_change() callback, and when phylink is used, the PHY state machine always needs to be stopped/ started on the suspend/resume path. The code is structured as such that if phydev->phy_link_change() is absent, it is a matter of time until the kernel will crash - no need to further complicate the test. Thus, for the situation where the PM is not managed b ---truncated---

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-39937	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net: rfkill: gpio: Fix crash due to dereferencing uninitialized pointer Since commit 7d5e9737efda ("net: rfkill: gpio: get the name and type from device property") rfkill_find_type() gets called with the possibly uninitialized "const char *type_name;" local variable. On x86 systems when rfkill-gpio binds to a "BCM4752" or "LNV4752" acpi_device, the rfkill->type is set based on the ACPI acpi_device_id: rfkill->type = (unsigned)id->driver_data; and there is no "type" property so device_property_read_string() will fail and leave type_name uninitialized, leading to a potential crash. rfkill_find_type() does accept a NULL pointer, fix the potential crash by initializing type_name to NULL. Note likely so far this has not been caught because: 1. Not many x86 machines actually have a "BCM4752"/"LNV4752" acpi_device 2. The stack happened to contain NULL where type_name is stored
CVE-2026-22977	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net: sock: fix hardened usercopy panic in sock_recv_errqueue skbuff_fclone_cache was created without defining a usercopy region, [1] unlike skbuff_head_cache which properly whitelists the cb[] field. [2] This causes a usercopy BUG() when CONFIG_HARDENED_USERCOPY is enabled and the kernel attempts to copy sk_buff.cb data to userspace via sock_recv_errqueue() -> put_cmsg(). The crash occurs when: 1. TCP allocates an skb using alloc_skb_fclone() (from skbuff_fclone_cache) [1] 2. The skb is cloned via skb_clone() using the pre-allocated fclone [3] 3. The cloned skb is queued to sk_error_queue for timestamp reporting 4. Userspace reads the error queue via recvmsg(MSG_ERRQUEUE) 5. sock_recv_errqueue() calls put_cmsg() to copy serr->ee from skb->cb [4] 6. __check_heap_object() fails because skbuff_fclone_cache has no usercopy whitelist [5] When cloned skbs allocated from skbuff_fclone_cache are used in the socket error queue, accessing the sock_exterr_skb structure in skb->cb via put_cmsg() triggers a usercopy hardening violation: [5.379589] usercopy: Kernel memory exposure attempt detected from SLUB object 'skbuff_fclone_cache' (offset 296, size 16)! [5.382796] kernel BUG at mm/usercopy.c:102! [5.383923] Oops: invalid opcode: 0000 [#1] SMP KASAN NOPTI [5.384903] CPU: 1 UID: 0 PID: 138 Comm: poc_put_cmsg Not tainted 6.12.57 #7 [5.384903] Hardware name: QEMU Standard PC (i440FX + PIIX, 1996), BIOS rel-1.16.3-0-ga6ed6b701f0a-prebuilt.qemu.org 04/01/2014 [5.384903] RIP: 0010:usercopy_abort+0x6c/0x80 [5.384903] Code: 1a 86 51 48 c7 c2 40 15 1a 86 41 52 48 c7 c7 c0 15 1a 86 48 0f 45 d6 48 c7 c6 80 15 1a 86 48 89 c1 49 0f 45 f3 e8 84 27 88 ff <0f> 0b 490 [5.384903] RSP: 0018:ffffc900006f77a8 EFLAGS: 00010246 [5.384903] RAX: 000000000000006f RBX: ffff88800f0ad2a8 RCX: 1fffffff0f72e74 [5.384903] RDX: 0000000000000000 RSI: 0000000000000004 RDI: ffffffff87b973a0 [5.384903] RBP: 0000000000000010 R08: 0000000000000000 R09: fffffbfff0f72e74 [5.384903] R10: 0000000000000003 R11: 79706f6372657375 R12: 0000000000000001 [5.384903] R13: ffff88800f0ad2b8 R14: ffffea00003c2b40 R15: ffffea00003c2b00 [5.384903] FS: 0000000011bc4380(0000) GS:ffff8880bf100000(0000) knlGS:0000000000000000 [5.384903] CS: 0010 DS: 0000 ES: 0000 CR0: 0000000080050033 [5.384903] CR2: 000056aa3b8e5fe4 CR3: 000000000ea26004 CR4: 0000000000770ef0 [5.384903] PKRU: 55555554 [5.384903] Call Trace: [5.384903] [5.384903] __check_heap_object+0x9a/0xd0 [5.384903] __check_object_size+0x46c/0x690 [5.384903] put_cmsg+0x129/0x5e0 [5.384903]

CVE	Severity	CVSS	CVSS Vector	Component	Description
					sock_recv_errqueue+0x22f/0x380 [5.384903] tls_sw_recvmsg+0x7ed/0x1960 [5.384903] ? srso_alias_return_thunk+0x5/0xfbef5 [5.384903] ? schedule+0x6d/0x270 [5.384903] ? srso_alias_return_thunk+0x5/0xfbef5 [5.384903] ? mutex_unlock+0x81/0xd0 [5.384903] ? __pfx_mutex_unlock+0x10/0x10 [5.384903] ? __pfx_tls_sw_recvmsg+0x10/0x10 [5.384903] ? _raw_spin_lock_irqsave+0x8f/0xf0 [5.384903] ? _raw_read_unlock_irqrestore+0x20/0x40 [5.384903] ? srso_alias_return_thunk+0x5/0xfbef5 The crash offset 296 corresponds to skb2->cb within skbuff_fclones: - sizeof(struct sk_buff) = 232 - offsetof(struct sk_buff, cb) = 40 - offset of skb2.cb in fclones = 232 + 40 = 272 - crash offset 296 = 272 + 24 (inside sock_exterr_skb.ee) This patch uses a local stack variable as a bounce buffer to avoid the hardened usercopy check failure. [1] https://elixir.bootlin.com/linux/v6.12.62/source/net/ipv4/tcp.c#L885 [2] https://elixir.bootlin.com/linux/v6.12.62/source/net/core/skbuff.c#L5104 [3] https://elixir.bootlin.com/linux/v6.12.62/source/net/core/skbuff.c#L5566 [4] https://elixir.bootlin.com/linux/v6.12.62/source/net/core/skbuff.c#L5491 [5] https://elixir.bootlin.com/linux/v6.12.62/source/mm/slub.c#L5719
CVE-2025-38725	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net: usb: asix_devices: add phy_mask for ax88772 mdio bus Without setting phy_mask for ax88772 mdio bus, current driver may create at most 32 mdio phy devices with phy address range from 0x00 ~ 0x1f. DLink DUB-E100 H/W Ver B1 is such a device. However, only one main phy device will bind to net phy driver. This is creating issue during system suspend/resume since phy_polling_mode() in phy_state_machine() will directly deference member of phydev->drv for non-main phy devices. Then NULL pointer dereference issue will occur. Due to only external phy or internal phy is necessary, add phy_mask for ax88772 mdio bus to workarnoud the issue.
CVE-2024-43826	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: nfs: pass explicit offset/count to trace events nfs_folio_length is unsafe to use without having the folio locked and a check for a NULL ->f_mapping that protects against truncations and can lead to kernel crashes. E.g. when running xfstests generic/065 with all nfs trace points enabled. Follow the model of the XFS trace points and pass in an explicit offset and length. This has the additional benefit that these values can be more accurate as some of the users touch partial folio ranges.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2024-36478	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: null_blk: fix null-ptr-dereference while configuring 'power' and 'submit_queues' Writing 'power' and 'submit_queues' concurrently will trigger kernel panic: Test script: modprobe null_blk nr_devices=0 mkdir -p /sys/kernel/config/nullb/nullb0 while true; do echo 1 > submit_queues; echo 4 > submit_queues; done & while true; do echo 1 > power; echo 0 > power; done Test result: BUG: kernel NULL pointer dereference, address: 0000000000000148 Oops: 0000 [#1] PREEMPT SMP RIP: 0010:__lock_acquire+0x41d/0x28f0 Call Trace: lock_acquire+0x121/0x450 down_write+0x5f/0x1d0 simple_recursive_removal+0x12f/0x5c0 blk_mq_debugfs_unregister_hctxs+0x7c/0x100 blk_mq_update_nr_hw_queues+0x4a3/0x720 nullb_update_nr_hw_queues+0x71/0xf0 [null_blk] nullb_device_submit_queues_store+0x79/0xf0 [null_blk] configfs_write_iter+0x119/0x1e0 vfs_write+0x326/0x730 ksys_write+0x74/0x150 This is because del_gendisk() can concurrent with blk_mq_update_nr_hw_queues(): nullb_device_power_store nullb_apply_submit_queues null_del_dev del_gendisk nullb_update_nr_hw_queues if (!dev->nullb) // still set while gendisk is deleted return 0 blk_mq_update_nr_hw_queues dev->nullb = NULL Fix this problem by resuing the global mutex to protect nullb_device_power_store() and nullb_update_nr_hw_queues() from configfs.
CVE-2025-39920	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: pcmcia: Add error handling for add_interval() in do_validate_mem() In the do_validate_mem(), the call to add_interval() does not handle errors. If kmalloc() fails in add_interval(), it could result in a null pointer being inserted into the linked list, leading to illegal memory access when sub_interval() is called next. This patch adds an error handling for the add_interval(). If add_interval() returns an error, the function will return early with the error code.
CVE-2025-39846	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: pcmcia: Fix a NULL pointer dereference in __iodyn_find_io_region() In __iodyn_find_io_region(), pcmcia_make_resource() is assigned to res and used in pci_bus_alloc_resource(). There is a dereference of res in pci_bus_alloc_resource(), which could lead to a NULL pointer dereference on failure of pcmcia_make_resource(). Fix this bug by adding a check of res.
CVE-2024-53205	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: phy: realtek: usb: fix NULL deref in rtk_usb2phy_probe In rtk_usb2phy_probe() devm_kzalloc() may return NULL but this returned value is not checked.
CVE-2024-53204	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: phy: realtek: usb: fix NULL deref in rtk_usb3phy_probe In rtk_usb3phy_probe() devm_kzalloc() may return NULL but this returned value is not checked.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38516	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: pinctrl: qcom: msm: mark certain pins as invalid for interrupts On some platforms, the UFS-reset pin has no interrupt logic in TLMM but is nevertheless registered as a GPIO in the kernel. This enables the user-space to trigger a BUG() in the pinctrl-msm driver by running, for example: `gpiomon -c 0 113` on RB2. The exact culprit is requesting pins whose intr_detection_width setting is not 1 or 2 for interrupts. This hits a BUG() in msm_gpio_irq_set_type(). Potentially crashing the kernel due to an invalid request from user-space is not optimal, so let's go through the pins and mark those that would fail the check as invalid for the irq chip as we should not even register them as available irqs. This function can be extended if we determine that there are more corner-cases like this.
CVE-2026-23237	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: platform/x86: classmate-laptop: Add missing NULL pointer checks In a few places in the Classmate laptop driver, code using the accel object may run before that object's address is stored in the driver data of the input device using it. For example, cmpc_accel_sensitivity_store_v4() is the "show" method of cmpc_accel_sensitivity_attr_v4 which is added in cmpc_accel_add_v4(), before calling dev_set_drvdata() for inputdev->dev. If the sysfs attribute is accessed prematurely, the dev_get_drvdata(&inputdev->dev) call in in cmpc_accel_sensitivity_store_v4() returns NULL which leads to a NULL pointer dereference going forward. Moreover, sysfs attributes using the input device are added before initializing that device by cmpc_add_acpi_notify_device() and if one of them is accessed before running that function, a NULL pointer dereference will occur. For example, cmpc_accel_sensitivity_attr_v4 is added before calling cmpc_add_acpi_notify_device() and if it is read prematurely, the dev_get_drvdata(&acpi->dev) call in cmpc_accel_sensitivity_show_v4() returns NULL which leads to a NULL pointer dereference going forward. Fix this by adding NULL pointer checks in all of the relevant places.
CVE-2026-23062	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: platform/x86: hp-bioscfg: Fix kernel panic in GET_INSTANCE_ID macro The GET_INSTANCE_ID macro that caused a kernel panic when accessing sysfs attributes: 1. Off-by-one error: The loop condition used '<=' instead of '<', causing access beyond array bounds. Since array indices are 0-based and go from 0 to instances_count-1, the loop should use '<'. 2. Missing NULL check: The code dereferenced attr_name_kobj->name without checking if attr_name_kobj was NULL, causing a null pointer dereference in min_length_show() and other attribute show functions. The panic occurred when fwupd tried to read BIOS configuration attributes: Oops: general protection fault [#1] SMP KASAN NOPTI KASAN: null-ptr-deref in range [0x0000000000000000-0x0000000000000007] RIP: 0010:min_length_show+0xcf/0x1d0 [hp_bioscfg] Add a NULL check for attr_name_kobj before dereferencing and corrects the loop boundary to match the pattern used elsewhere in the driver.
CVE-2025-38634	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: power: supply: cpcap-charger: Fix null check for power_supply_get_by_name In the cpcap_usb_detect() function, the power_supply_get_by_name() function may return `NULL` instead of an error pointer. To prevent potential null pointer dereferences, Added a null check.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38610	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: powercap: dtpm_cpu: Fix NULL pointer dereference in get_pd_power_uw() The get_pd_power_uw() function can crash with a NULL pointer dereference when em_cpu_get() returns NULL. This occurs when a CPU becomes impossible during runtime, causing get_cpu_device() to return NULL, which propagates through em_cpu_get() and leads to a crash when em_span_cpus() dereferences the NULL pointer. Add a NULL check after em_cpu_get() and return 0 if unavailable, matching the existing fallback behavior in __dtpm_cpu_setup(). [rjw: Drop an excess empty code line]
CVE-2025-38668	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: regulator: core: fix NULL dereference on unbind due to stale coupling data Failing to reset coupling_desc.n_coupled after freeing coupled_rdevs can lead to NULL pointer dereference when regulators are accessed post-unbind. This can happen during runtime PM or other regulator operations that rely on coupling metadata. For example, on ridesx4, unbinding the 'reg-dummy' platform device triggers a panic in regulator_lock_recursive() due to stale coupling state. Ensure n_coupled is set to 0 to prevent access to invalid pointers.
CVE-2025-38489	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: s390/bpf: Fix bpf_arch_text_poke() with new_addr == NULL again Commit 7ded842b356d ("s390/bpf: Fix bpf_plt pointer arithmetic") has accidentally removed the critical piece of commit c730fce7c70c ("s390/bpf: Fix bpf_arch_text_poke() with new_addr == NULL"), causing intermittent kernel panics in e.g. perf's on_switch() prog to reappear. Restore the fix and add a comment.
CVE-2025-39694	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: s390/sclp: Fix SCCB present check Tracing code called by the SCLP interrupt handler contains early exits if the SCCB address associated with an interrupt is NULL. This check is performed after physical to virtual address translation. If the kernel identity mapping does not start at address zero, the resulting virtual address is never zero, so that the NULL checks won't work. Subsequently this may result in incorrect accesses to the first page of the identity mapping. Fix this by introducing a function that handles the NULL case before address translation.
CVE-2025-39895	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: sched: Fix sched_numa_find_nth_cpu() if mask offline sched_numa_find_nth_cpu() uses a bsearch to look for the 'closest' CPU in sched_domains_numa_masks and given cpus mask. However they might not intersect if all CPUs in the cpus mask are offline. bsearch will return NULL in that case, bail out instead of dereferencing a bogus pointer. The previous behaviour lead to this bug when using maxcpus=4 on an rk3399 (LLLLbb) (i.e. booting with all big CPUs offline): [1.422922] Unable to handle kernel paging request at virtual address ffffff8000000000 [1.423635] Mem abort info: [1.423889] ESR = 0x0000000096000006 [1.424227] EC = 0x25: DABT (current EL), IL = 32 bits [1.424715] SET = 0, FnV = 0 [1.424995] EA = 0, S1PTW = 0 [1.425279] FSC = 0x06: level 2 translation fault [1.425735] Data abort info: [1.425998] ISV = 0, ISS = 0x00000006, ISS2 = 0x00000000 [1.426499] CM = 0, WnR = 0, TnD = 0, TagAccess = 0 [1.426952] GCS = 0, Overlay = 0, DirtyBit = 0, Xs = 0 [1.427428] swapper pgtable: 4k pages, 39-bit VAs, pgdp=0000000004a9f000 [1.428038]

CVE	Severity	CVSS	CVSS Vector	Component	Description
					[ffffff8000000000] pgd=18000000f7fff403, p4d=18000000f7fff403, pud=18000000f7fff403, pmd=0000000000000000 [1.429014] Internal error: Oops: 0000000096000006 [#1] SMP [1.429525] Modules linked in: [1.429813] CPU: 3 UID: 0 PID: 1 Comm: swapper/0 Not tainted 6.17.0-rc4-dirty #343 PREEMPT [1.430559] Hardware name: Pine64 RockPro64 v2.1 (DT) [1.431012] pstate: 60000005 (nZCv daif -PAN -UAO -TCO -DIT -SSBS BTYPE=--) [1.431634] pc : sched_numa_find_nth_cpu+0x2a0/0x488 [1.432094] lr : sched_numa_find_nth_cpu+0x284/0x488 [1.432543] sp : ffffffc084e1b960 [1.432843] x29: ffffffc084e1b960 x28: ffffff80078a8800 x27: ffffffc0846eb1d0 [1.433495] x26: 0000000000000000 x25: 0000000000000000 x24: 0000000000000000 [1.434144] x23: 0000000000000000 x22: ffffffffff7f093 x21: ffffffc081de6378 [1.434792] x20: 0000000000000000 x19: 0000000ffff7f093 x18: 00000000ffffffff [1.435441] x17: 3030303866666666 x16: 66663d736b73616d x15: ffffffc104e1b5b7 [1.436091] x14: 0000000000000000 x13: ffffffc084712860 x12: 0000000000000372 [1.436739] x11: 0000000000000126 x10: ffffffc08476a860 x9 : ffffffc084712860 [1.437389] x8 : 00000000fffefff x7 : ffffffc08476a860 x6 : 0000000000000000 [1.438036] x5 : 000000000000bff4 x4 : 0000000000000000 x3 : 0000000000000000 [1.438683] x2 : 0000000000000000 x1 : ffffffc0846eb000 x0 : ffffff8000407b68 [1.439332] Call trace: [1.439559] sched_numa_find_nth_cpu+0x2a0/0x488 (P) [1.440016] smp_call_function_any+0xc8/0xd0 [1.440416] armv8_pmu_init+0x58/0x27c [1.440770] armv8_cortex_a72_pmu_init+0x20/0x2c [1.441199] arm_pmu_device_probe+0x1e4/0x5e8 [1.441603] armv8_pmu_device_probe+0x1c/0x28 [1.442007] platform_probe+0x5c/0xac [1.442347] really_probe+0xbc/0x298 [1.442683] __driver_probe_device+0x78/0x12c [1.443087] driver_probe_device+0xdc/0x160 [1.443475] __driver_attach+0x94/0x19c [1.443833] bus_for_each_dev+0x74/0xd4 [1.444190] driver_attach+0x24/0x30 [1.444525] bus_add_driver+0xe4/0x208 [1.444874] driver_register+0x60/0x128 [1.445233] __platform_driver_register+0x24/0x30 [1.445662] armv8_pmu_driver_init+0x28/0x4c [1.446059] do_one_initcall+0x44/0x25c [1.446416] kernel_init_freeable+0x1dc/0x3bc [1.446820] kernel_init+0x20/0x1d8 [1.447151] ret_from_fork+0x10/0x20 [1.447493] Code: 90022e21 f000e5f5 910de2b5 2a1703e2 (f8767803) [1.448040] ---[end trace 0000000000000000]--- [1.448483] note: swapper/0[1] exited with preempt_count 1 [1.449047] Kernel panic - not syncing: Attempted to kill init! exitcode=0x0000000b [1.449741] SMP: stopping secondary CPUs [1.450105] Kernel Offset: disabled [1.450419] CPU features: 0x000000,00080000,20002001,0400421b [---truncated---

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38700	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: scsi: libiscsi: Initialize iscsi_conn->dd_data only if memory is allocated In case of an ib_fast_reg_mr allocation failure during iSER setup, the machine hits a panic because iscsi_conn->dd_data is initialized unconditionally, even when no memory is allocated (dd_size == 0). This leads invalid pointer dereference during connection teardown. Fix by setting iscsi_conn->dd_data only if memory is actually allocated. Panic trace: ---------- iser: iser_create_fastreg_desc: Failed to allocate ib_fast_reg_mr err=-12 iser: iser_alloc_rx_descriptors: failed allocating rx descriptors / data buffers BUG: unable to handle page fault for address: fffffffffffffff8 RIP: 0010:swake_up_locked.part.5+0xa/0x40 Call Trace: complete+0x31/0x40 iscsi_iser_conn_stop+0x88/0xb0 [ib_iser] iscsi_stop_conn+0x66/0xc0 [scsi_transport_iscsi] iscsi_if_stop_conn+0x14a/0x150 [scsi_transport_iscsi] iscsi_if_rx+0x1135/0x1834 [scsi_transport_iscsi] ? netlink_lookup+0x12f/0x1b0 ? netlink_deliver_tap+0x2c/0x200 netlink_unicast+0x1ab/0x280 netlink_sendmsg+0x257/0x4f0 ? _copy_from_user+0x29/0x60 sock_sendmsg+0x5f/0x70
CVE-2025-38695	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: scsi: lpfc: Check for hdwq null ptr when cleaning up lpfc_vport structure If a call to lpfc_sli4_read_rev() from lpfc_sli4_hba_setup() fails, the resultant cleanup routine lpfc_sli4_vport_delete_fcp_xri_aborted() may occur before sli4_hba.hdwqs are allocated. This may result in a null pointer dereference when attempting to take the abts_io_buf_list_lock for the first hardware queue. Fix by adding a null ptr check on phba->sli4_hba.hdwq and early return because this situation means there must have been an error during port initialization.
CVE-2025-21723	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: scsi: mpi3mr: Fix possible crash when setting up bsg fails If bsg_setup_queue() fails, the bsg_queue is assigned a non-NULL value. Consequently, in mpi3mr_bsg_exit(), the condition "if(!mrioc->bsg_queue)" will not be satisfied, preventing execution from entering bsg_remove_queue(), which could lead to the following crash: BUG: kernel NULL pointer dereference, address: 000000000000041c Call Trace: mpi3mr_bsg_exit+0x1f/0x50 [mpi3mr] mpi3mr_remove+0x6f/0x340 [mpi3mr] pci_device_remove+0x3f/0xb0 device_release_driver_internal+0x19d/0x220 unbind_store+0xa4/0xb0 kernfs_fop_write_iter+0x11f/0x200 vfs_write+0x1fc/0x3e0 ksys_write+0x67/0xe0 do_syscall_64+0x38/0x80 entry_SYSCALL_64_after_hwframe+0x78/0xe2

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-71236	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: scsi: qla2xxx: Validate sp before freeing associated memory System crash with the following signature [154563.214890] nvme nvme2: NVME-FC{1}: controller connect complete [154564.169363] qla2xxx [0000:b0:00.1]-3002:2: nvme: Sched: Set ZIO exchange threshold to 3. [154564.169405] qla2xxx [0000:b0:00.1]-ffffff:2: SET ZIO Activity exchange threshold to 5. [154565.539974] qla2xxx [0000:b0:00.1]-5013:2: RSCN database changed – 0078 0080 0000. [154565.545744] qla2xxx [0000:b0:00.1]-5013:2: RSCN database changed – 0078 00a0 0000. [154565.545857] qla2xxx [0000:b0:00.1]-11a2:2: FEC=enabled (data rate). [154565.552760] qla2xxx [0000:b0:00.1]-11a2:2: FEC=enabled (data rate). [154565.553079] BUG: kernel NULL pointer dereference, address: 00000000000000f8 [154565.553080] #PF: supervisor read access in kernel mode [154565.553082] #PF: error_code(0x0000) - not-present page [154565.553084] PGD 80000010488ab067 P4D 80000010488ab067 PUD 104978a067 PMD 0 [154565.553089] Oops: 0000 1 PREEMPT SMP PTI [154565.553092] CPU: 10 PID: 858 Comm: qla2xxx_2_dpc Kdump: loaded Tainted: G OE ------ --- 5.14.0-503.11.1.el9_5.x86_64 #1 [154565.553096] Hardware name: HPE Synergy 660 Gen10/Synergy 660 Gen10 Compute Module, BIOS I43 09/30/2024 [154565.553097] RIP: 0010:qla_fab_async_scan.part.0+0x40b/0x870 [qla2xxx] [154565.553141] Code: 00 00 e8 58 a3 ec d4 49 89 e9 ba 12 20 00 00 4c 89 e6 49 c7 c0 00 ee a8 c0 48 c7 c1 66 c0 a9 c0 bf 00 80 00 10 e8 15 69 00 00 <4c> 8b 8d f8 00 00 00 4d 85 c9 74 35 49 8b 84 24 00 19 00 00 48 8b [154565.553143] RSP: 0018:ffffb4dbc8aebdd0 EFLAGS: 00010286 [154565.553145] RAX: 0000000000000000 RBX: ffff8ec2cf0908d0 RCX: 0000000000000002 [154565.553147] RDX: 0000000000000000 RSI: ffffffffc0a9c896 RDI: ffffb4dbc8aebd47 [154565.553148] RBP: 0000000000000000 R08: ffffb4dbc8aebd45 R09: 00000000000fff0a [154565.553150] R10: 0000000000000000 R11: 000000000000000f R12: ffff8ec2cf0908d0 [154565.553151] R13: ffff8ec2cf090900 R14: 0000000000000102 R15: ffff8ec2cf084000 [154565.553152] FS: 0000000000000000(0000) GS:ffff8ed27f800000(0000) knlGS:0000000000000000 [154565.553154] CS: 0010 DS: 0000 ES: 0000 CR0: 0000000080050033 [154565.553155] CR2: 00000000000000f8 CR3: 000000113ae0a005 CR4: 00000000007706f0 [154565.553157] DR0: 0000000000000000 DR1: 0000000000000000 DR2: 0000000000000000 [154565.553158] DR3: 0000000000000000 DR6: 00000000fffe0ff0 DR7: 0000000000000400 [154565.553159] PKRU: 55555554 [154565.553160] Call Trace: [154565.553162] [154565.553165] ? show_trace_log_lvl+0x1c4/0x2df [154565.553172] ? show_trace_log_lvl+0x1c4/0x2df [154565.553177] ? qla_fab_async_scan.part.0+0x40b/0x870 [qla2xxx] [154565.553215] ? __die_body.cold+0x8/0xd [154565.553218] ? page_fault_oops+0x134/0x170 [154565.553223] ? snprintf+0x49/0x70 [154565.553229] ? exc_page_fault+0x62/0x150 [154565.553238] ? asm_exc_page_fault+0x22/0x30 Check for sp being non NULL before freeing any associated memory

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-39676	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: scsi: qla4xxx: Prevent a potential error pointer dereference The qla4xxx_get_ep_fwdb() function is supposed to return NULL on error, but qla4xxx_ep_connect() returns error pointers. Propagating the error pointers will lead to an Oops in the caller, so change the error pointers to NULL.
CVE-2025-38399	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: scsi: target: Fix NULL pointer dereference in core_scsi3_decode_spec_i_port() The function core_scsi3_decode_spec_i_port(), in its error code path, unconditionally calls core_scsi3_lunacl_undepend_item() passing the dest_se_deve pointer, which may be NULL. This can lead to a NULL pointer dereference if dest_se_deve remains unset. SPC-3 PR SPEC_I_PT: Unable to locate dest_tpg Unable to handle kernel paging request at virtual address dfff800000000012 Call trace: core_scsi3_lunacl_undepend_item+0x2c/0xf0 [target_core_mod] (P) core_scsi3_decode_spec_i_port+0x120c/0x1c30 [target_core_mod] core_scsi3_emulate_pro_register+0x6b8/0xcd8 [target_core_mod] target_scsi3_emulate_pr_out+0x56c/0x840 [target_core_mod] Fix this by adding a NULL check before calling core_scsi3_lunacl_undepend_item()
CVE-2025-37826	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: scsi: ufs: core: Add NULL check in ufshcd_mcq_compl_pending_transfer() Add a NULL check for the returned hwq pointer by ufshcd_mcq_req_to_hwq(). This is similar to the fix in commit 74736103fb41 ("scsi: ufs: core: Fix ufshcd_abort_one racing issue").
CVE-2024-56620	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: scsi: ufs: qcom: Only free platform MSIs when ESI is enabled Otherwise, it will result in a NULL pointer dereference as below: Unable to handle kernel NULL pointer dereference at virtual address 0000000000000008 Call trace: mutex_lock+0xc/0x54 platform_device_msi_free_irqs_all+0x14/0x20 ufs_qcom_remove+0x34/0x48 [ufs_qcom] platform_remove+0x28/0x44 device_remove+0x4c/0x80 device_release_driver_internal+0xd8/0x178 driver_detach+0x50/0x9c bus_remove_driver+0x6c/0xbc driver_unregister+0x30/0x60 platform_driver_unregister+0x14/0x20 ufs_qcom_pltform_exit+0x18/0xb94 [ufs_qcom] __arm64_sys_delete_module+0x180/0x260 invoke_syscall+0x44/0x100 el0_svc_common.constprop.0+0xc0/0xe0 do_el0_svc+0x1c/0x28 el0_svc+0x34/0xdc el0t_64_sync_handler+0xc0/0xc4 el0t_64_sync+0x190/0x194

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-22062	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: sctp: add mutual exclusion in proc_sctp_do_udp_port() We must serialize calls to sctp_udp_sock_stop() and sctp_udp_sock_start() or risk a crash as syzbot reported: Oops: general protection fault, probably for non-canonical address 0xdfffffc000000000d: 0000 [#1] SMP KASAN PTI KASAN: null-ptr-deref in range [0x0000000000000068-0x000000000000006f] CPU: 1 UID: 0 PID: 6551 Comm: syz.1.44 Not tainted 6.14.0-syzkaller-g7f2ff7b62617 #0 PREEMPT(full) Hardware name: Google Google Compute Engine/Google Compute Engine, BIOS Google 02/12/2025 RIP: 0010:kernel_sock_shutdown+0x47/0x70 net/socket.c:3653 Call Trace: udp_tunnel_sock_release+0x68/0x80 net/ipv4/udp_tunnel_core.c:181 sctp_udp_sock_stop+0x71/0x160 net/sctp/protocol.c:930 proc_sctp_do_udp_port+0x264/0x450 net/sctp/sysctl.c:553 proc_sys_call_handler+0x3d0/0x5b0 fs/proc/proc_sysctl.c:601 iter_file_splice_write+0x91c/0x1150 fs/splice.c:738 do_splice_from fs/splice.c:935 [inline] direct_splice_actor+0x18f/0x6c0 fs/splice.c:1158 splice_direct_to_actor+0x342/0xa30 fs/splice.c:1102 do_splice_direct_actor fs/splice.c:1201 [inline] do_splice_direct+0x174/0x240 fs/splice.c:1227 do_sendfile+0xafd/0xe50 fs/read_write.c:1368 __do_sys_sendfile64 fs/read_write.c:1429 [inline] __se_sys_sendfile64 fs/read_write.c:1415 [inline] __x64_sys_sendfile64+0x1d8/0x220 fs/read_write.c:1415 do_syscall_x64 arch/x86/entry/syscall_64.c:63 [inline]

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2026-23125	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: sctp: move SCTP_CMD_ASSOC_SHKEY right after SCTP_CMD_PEER_INIT A null-ptr-deref was reported in the SCTP transmit path when SCTP-AUTH key initialization fails: ================================== ========== KASAN: null-ptr-deref in range [0x0000000000000018-0x000000000000001f] CPU: 0 PID: 16 Comm: ksoftirqd/0 Tainted: G W 6.6.0 #2 RIP: 0010:sctp_packet_bundle_auth net/sctp/output.c:264 [inline] RIP: 0010:sctp_packet_append_chunk+0xb36/0x1260 net/sctp/output.c:401 Call Trace: sctp_packet_transmit_chunk+0x31/0x250 net/sctp/output.c:189 sctp_outq_flush_data+0xa29/0x26d0 net/sctp/outqueue.c:1111 sctp_outq_flush+0xc80/0x1240 net/sctp/outqueue.c:1217 sctp_cmd_interpreter.isra.0+0x19a5/0x62c0 net/sctp/sm_sideeffect.c:1787 sctp_side_effects net/sctp/sm_sideeffect.c:1198 [inline] sctp_do_sm+0x1a3/0x670 net/sctp/sm_sideeffect.c:1169 sctp_assoc_bh_rcv+0x33e/0x640 net/sctp/associola.c:1052 sctp_inq_push+0x1dd/0x280 net/sctp/inqueue.c:88 sctp_rcv+0x11ae/0x3100 net/sctp/input.c:243 sctp6_rcv+0x3d/0x60 net/sctp/ipv6.c:1127 The issue is triggered when sctp_auth_asoc_init_active_key() fails in sctp_sf_do_5_1C_ack() while processing an INIT_ACK. In this case, the command sequence is currently: - SCTP_CMD_PEER_INIT - SCTP_CMD_TIMER_STOP (T1_INIT) - SCTP_CMD_TIMER_START (T1_COOKIE) - SCTP_CMD_NEW_STATE (COOKIE_ECHOED) - SCTP_CMD_ASSOC_SHKEY - SCTP_CMD_GEN_COOKIE_ECHO If SCTP_CMD_ASSOC_SHKEY fails, asoc->shkey remains NULL, while asoc->peer.auth_capable and asoc->peer.peer_chunks have already been set by SCTP_CMD_PEER_INIT. This allows a DATA chunk with auth = 1 and shkey = NULL to be queued by sctp_datamsg_from_user(). Since command interpretation stops on failure, no COOKIE_ECHO should been sent via SCTP_CMD_GEN_COOKIE_ECHO. However, the T1_COOKIE timer has already been started, and it may enqueue a COOKIE_ECHO into the outqueue later. As a result, the DATA chunk can be transmitted together with the COOKIE_ECHO in sctp_outq_flush_data(), leading to the observed issue. Similar to the other places where it calls sctp_auth_asoc_init_active_key() right after sctp_process_init(), this patch moves the SCTP_CMD_ASSOC_SHKEY immediately after SCTP_CMD_PEER_INIT, before stopping T1_INIT and starting T1_COOKIE. This ensures that if shared key generation fails, authenticated DATA cannot be sent. It also allows the T1_INIT timer to retransmit INIT, giving the client another chance to process INIT_ACK and retry key setup.
CVE-2025-37860	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: sfc: fix NULL dereferences in ef100_process_design_param() Since cited commit, ef100_probe_main() and hence also ef100_check_design_params() run before efx->net_dev is created; consequently, we cannot netif_set_tso_max_size() or _segs() at this point. Move those netif calls to ef100_probe_netdev(), and also replace netif_err within the design params code with pci_err.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-39692	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: smb: server: split ksmbd_rdma_stop_listening() out of ksmbd_rdma_destroy() We can't call destroy_workqueue(smb_direct_wq); before stop_sessions()! Otherwise already existing connections try to use smb_direct_wq as a NULL pointer.
CVE-2025-38487	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: soc: aspeed: lpc-snoop: Don't disable channels that aren't enabled Mitigate e.g. the following: # echo 1e789080.lpc-snoop > /sys/bus/platform/drivers/aspeed-lpc-snoop/unbind ... [120.363594] Unable to handle kernel NULL pointer dereference at virtual address 00000004 when write [120.373866] [00000004] *pgd=00000000 [120.377910] Internal error: Oops: 805 [#1] SMP ARM[120.383306] CPU: 1 UID: 0 PID: 315 Comm: sh Not tainted 6.15.0-rc1-00009-g926217bc7d7d-dirty #20 NONE ... [120.679543] Call trace: [120.679559] misc_deregister from aspeed_lpc_snoop_remove+0x84/0xac [120.692462] aspeed_lpc_snoop_remove from platform_remove+0x28/0x38 [120.700996] platform_remove from device_release_driver_internal+0x188/0x200 ...
CVE-2025-38648	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: spi: stm32: Check for cfg availability in stm32_spi_probe The stm32_spi_probe function now includes a check to ensure that the pointer returned by of_device_get_match_data is not NULL before accessing its members. This resolves a warning where a potential NULL pointer dereference could occur when accessing cfg->has_device_mode. Before accessing the 'has_device_mode' member, we verify that 'cfg' is not NULL. If 'cfg' is NULL, an error message is logged. This change ensures that the driver does not attempt to access configuration data if it is not available, thus preventing a potential system crash due to a NULL pointer dereference.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-39865	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: tee: fix NULL pointer dereference in tee_shm_put tee_shm_put have NULL pointer dereference: __optee_disable_shm_cache --> shm = reg_pair_to_ptr(...);//shm maybe return NULL tee_shm_free(shm); --> tee_shm_put(shm);//crash Add check in tee_shm_put to fix it. panic log: Unable to handle kernel paging request at virtual address 0000000000100cca Mem abort info: ESR = 0x0000000096000004 EC = 0x25: DABT (current EL), IL = 32 bits SET = 0, FnV = 0 EA = 0, S1PTW = 0 FSC = 0x04: level 0 translation fault Data abort info: ISV = 0, ISS = 0x00000004, ISS2 = 0x00000000 CM = 0, WnR = 0, TnD = 0, TagAccess = 0 GCS = 0, Overlay = 0, DirtyBit = 0, Xs = 0 user pgtable: 4k pages, 48-bit VAs, pgdp=0000002049d07000 [0000000000100cca] pgd=0000000000000000, p4d=0000000000000000 Internal error: Oops: 0000000096000004 [#1] SMP CPU: 2 PID: 14442 Comm: systemd-sleep Tainted: P OE ------- ---- 6.6.0-39-generic #38 Source Version: 938b255f6cb8817c95b0dd5c8c2944acfce94b07 Hardware name: greatwall GW-001Y1A-FTH, BIOS Great Wall BIOS V3.0 10/26/2022 pstate: 80000005 (Nzcv daif -PAN -UAO -TCO -DIT -SSBS BTYPE=--) pc : tee_shm_put+0x24/0x188 lr : tee_shm_free+0x14/0x28 sp : ffff001f98f9faf0 x29: ffff001f98f9faf0 x28: ffff0020df543cc0 x27: 0000000000000000 x26: ffff001f811344a0 x25: ffff8000818dac00 x24: ffff800082d8d048 x23: ffff001f850fcd18 x22: 0000000000000001 x21: ffff001f98f9fb88 x20: ffff001f83e76218 x19: ffff001f83e761e0 x18: 000000000000ffff x17: 303a30303a303030 x16: 0000000000000000 x15: 0000000000000003 x14: 0000000000000001 x13: 0000000000000000 x12: 0101010101010101 x11: 0000000000000001 x10: 0000000000000001 x9 : ffff800080e08d0c x8 : ffff001f98f9fb88 x7 : 0000000000000000 x6 : 0000000000000000 x5 : 0000000000000000 x4 : 0000000000000000 x3 : 0000000000000000 x2 : ffff001f83e761e0 x1 : 00000000ffff001f x0 : 0000000000100cca Call trace: tee_shm_put+0x24/0x188 tee_shm_free+0x14/0x28 __optee_disable_shm_cache+0xa8/0x108 optee_shutdown+0x28/0x38 platform_shutdown+0x28/0x40 device_shutdown+0x144/0x2b0 kernel_power_off+0x3c/0x80 hibernate+0x35c/0x388 state_store+0x64/0x80 kobj_attr_store+0x14/0x28 sysfs_kf_write+0x48/0x60 kernfs_fop_write_iter+0x128/0x1c0 vfs_write+0x270/0x370 ksys_write+0x6c/0x100 __arm64_sys_write+0x20/0x30 invoke_syscall+0x4c/0x120 el0_svc_common.constprop.0+0x44/0xf0 do_el0_svc+0x24/0x38 el0_svc+0x24/0x88 el0t_64_sync_handler+0x134/0x150 el0t_64_sync+0x14c/0x15

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-71125	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: tracing: Do not register unsupported perf events Synthetic events currently do not have a function to register perf events. This leads to calling the tracepoint register functions with a NULL function pointer which triggers: ------------[cut here]------------ WARNING: kernel/tracepoint.c:175 at tracepoint_add_func+0x357/0x370, CPU#2: perf/2272 Modules linked in: kvm_intel kvm irqbypass CPU: 2 UID: 0 PID: 2272 Comm: perf Not tainted 6.18.0-ftest-11964-ge022764176fc-dirty #323 PREEMPTLAZY Hardware name: QEMU Standard PC (Q35 + ICH9, 2009), BIOS 1.17.0-debian-1.17.0-1 04/01/2014 RIP: 0010:tracepoint_add_func+0x357/0x370 Code: 28 9c e8 4c 0b f5 ff eb 0f 4c 89 f7 48 c7 c6 80 4d 28 9c e8 ab 89 f4 ff 31 c0 5b 41 5c 41 5d 41 5e 41 5f 5d c3 cc cc cc cc cc <0f> 0b 49 c7 c6 ea ff ff ff e9 ee fe ff ff 0f 0b e9 f9 fe ff ff 0f RSP: 0018:ffffabc0c44d3c40 EFLAGS: 00010246 RAX: 0000000000000001 RBX: ffff9380aa9e4060 RCX: 0000000000000000 RDX: 000000000000000a RSI: ffffffff9e1d4a98 RDI: ffff937fcf5fd6c8 RBP: 0000000000000001 R08: 0000000000000007 R09: ffff937fcf5fc780 R10: 0000000000000003 R11: ffffffff9c193910 R12: 000000000000000a R13: ffffffff9e1e5888 R14: 0000000000000000 R15: ffffabc0c44d3c78 FS: 00007f6202f5f340(0000) GS:ffff93819f00f000(0000) knlGS:0000000000000000 CS: 0010 DS: 0000 ES: 0000 CR0: 0000000080050033 CR2: 000055d3162281a8 CR3: 0000000106a56003 CR4: 0000000000172ef0 Call Trace: tracepoint_probe_register+0x5d/0x90 synth_event_reg+0x3c/0x60 perf_trace_event_init+0x204/0x340 perf_trace_init+0x85/0xd0 perf_tp_event_init+0x2e/0x50 perf_try_init_event+0x6f/0x230 ? perf_event_alloc+0x4bb/0xdc0 perf_event_alloc+0x65a/0xdc0 __se_sys_perf_event_open+0x290/0x9f0 do_syscall_64+0x93/0x7b0 ? entry_SYSCALL_64_after_hwframe+0x76/0x7e ? trace_hardirqs_off+0x53/0xc0 entry_SYSCALL_64_after_hwframe+0x76/0x7e Instead, have the code return -ENODEV, which doesn't warn and has perf error out with: # perf record -e synthetic:futex_wait Error: The sys_perf_event_open() syscall returned with 19 (No such device) for event (synthetic:futex_wait). "dmesg | grep -i perf" may provide additional information. Ideally perf should support synthetic events, but for now just fix the warning. The support can come later.
CVE-2026-23088	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: tracing: Fix crash on synthetic stacktrace field usage When creating a synthetic event based on an existing synthetic event that had a stacktrace field and the new synthetic event used that field a kernel crash occurred: ~# cd /sys/kernel/tracing ~# echo 's:stack unsigned long stack[];' > dynamic_events ~# echo 'hist:keys=prev_pid:s0=common_stacktrace if prev_state & 3' >> events/sched/sched_switch/trigger ~# echo 'hist:keys=next_pid:s1=$s0:onmatch(sched.sched_switch).trace(stack,$s1)' >> events/sched/sched_switch/trigger The above creates a synthetic event that takes a stacktrace when a task schedules out in a non-running state and passes that stacktrace to the sched_switch event when that task schedules back in. It triggers the "stack" synthetic event that has a stacktrace as its field (called "stack"). ~# echo 's:syscall_stack s64 id; unsigned long stack[];' >> dynamic_events ~# echo 'hist:keys=common_pid:s2=stack' >> events/synthetic/stack/trigger ~# echo 'hist:keys=common_pid:s3=$s2,i0=id:onmatch(synthetic.stack).trace(syscall_stack,$i0,$s3)' >>

CVE	Severity	CVSS	CVSS Vector	Component	Description
					events/raw_syscalls/sys_exit/trigger The above makes another synthetic event called "syscall_stack" that attaches the first synthetic event (stack) to the sys_exit trace event and records the stacktrace from the stack event with the id of the system call that is exiting. When enabling this event (or using it in a historgram): ~# echo 1 > events/synthetic/syscall_stack/enable Produces a kernel crash! BUG: unable to handle page fault for address: 0000000000400010 #PF: supervisor read access in kernel mode #PF: error_code(0x0000) - not-present page PGD 0 P4D 0 Oops: Oops: 0000 [#1] SMP PTI CPU: 6 UID: 0 PID: 1257 Comm: bash Not tainted 6.16.3+deb14-amd64 #1 PREEMPT(lazy) Debian 6.16.3-1 Hardware name: QEMU Standard PC (Q35 + ICH9, 2009), BIOS 1.17.0-debian-1.17.0-1 04/01/2014 RIP: 0010:trace_event_raw_event_synth+0x90/0x380 Code: c5 00 00 00 00 85 d2 0f 84 e1 00 00 00 31 db eb 34 0f 1f 00 66 66 2e 0f 1f 84 00 00 00 00 00 66 66 2e 0f 1f 84 00 00 00 00 00 <49> 8b 04 24 48 83 c3 01 8d 0c c5 08 00 00 00 01 cd 41 3b 5d 40 0f RSP: 0018:ffffd2670388f958 EFLAGS: 00010202 RAX: ffff8ba1065cc100 RBX: 0000000000000000 RCX: 0000000000000000 RDX: 0000000000000001 RSI: fffff266ffda7b90 RDI: ffffd2670388f9b0 RBP: 0000000000000010 R08: ffff8ba104e76000 R09: ffffd2670388fa50 R10: ffff8ba102dd42e0 R11: ffffffff9a908970 R12: 0000000000400010 R13: ffff8ba10a246400 R14: ffff8ba10a710220 R15: fffff266ffda7b90 FS: 00007fa3bc63f740(0000) GS:ffff8ba2e0f48000(0000) knlGS:0000000000000000 CS: 0010 DS: 0000 ES: 0000 CR0: 0000000080050033 CR2: 0000000000400010 CR3: 0000000107f9e003 CR4: 0000000000172ef0 Call Trace: ? __tracing_map_insert+0x208/0x3a0 action_trace+0x67/0x70 event_hist_trigger+0x633/0x6d0 event_triggers_call+0x82/0x130 trace_event_buffer_commit+0x19d/0x250 trace_event_raw_event_sys_exit+0x62/0xb0 syscall_exit_work+0x9d/0x140 do_syscall_64+0x20a/0x2f0 ? trace_event_raw_event_sched_switch+0x12b/0x170 ? save_fpregs_to_fpstate+0x3e/0x90 ? _raw_spin_unlock+0xe/0x30 ? finish_task_switch.isra.0+0x97/0x2c0 ? __rseq_handle_notify_resume+0xad/0x4c0 ? __schedule+0x4b8/0xd00 ? restore_fpregs_from_fpstate+0x3c/0x90 ? switch_fpu_return+0x5b/0xe0 ? do_syscall_64+0x1ef/0x2f0 ? do_fault+0x2e9/0x540 ? __handle_mm_fault+0x7d1/0xf70 ? count_memcg_events+0x167/0x1d0 ? handle_mm_fault+0x1d7/0x2e0 ? do_user_addr_fault+0x2c3/0x7f0 entry_SYSCALL_64_after_hwframe+0x76/0x7e The reason is that the stacktrace field is not labeled as such, and is treated as a normal field and not as a dynamic event that it is. In trace_event_raw_event_synth() the event is field is still treated as a dynamic array, but the retrieval of the data is considered a normal field, and the reference is just the meta data: // Meta data is retrieved instead of a dynamic array ---truncated---

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2026-23063	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: uacce: ensure safe queue release with state management Directly calling `put_queue` carries risks since it cannot guarantee that resources of `uacce_queue` have been fully released beforehand. So adding a `stop_queue` operation for the UACCE_CMD_PUT_Q command and leaving the `put_queue` operation to the final resource release ensures safety. Queue states are defined as follows: - UACCE_Q_ZOMBIE: Initial state - UACCE_Q_INIT: After opening `uacce` - UACCE_Q_STARTED: After `start` is issued via `ioctl` When executing `poweroff -f` in virt while accelerator are still working, `uacce_fops_release` and `uacce_remove` may execute concurrently. This can cause `uacce_put_queue` within `uacce_fops_release` to access a NULL `ops` pointer. Therefore, add state checks to prevent accessing freed pointers.
CVE-2025-37759	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ublk: fix handling recovery & reissue in ublk_abort_queue() Commit 8284066946e6 ("ublk: grab request reference when the request is handled by userspace") doesn't grab request reference in case of recovery reissue. Then the request can be requeued & re-dispatch & failed when canceling uring command. If it is one zc request, the request can be freed before io_uring returns the zc buffer back, then cause kernel panic: [126.773061] BUG: kernel NULL pointer dereference, address: 00000000000000c8 [126.773657] #PF: supervisor read access in kernel mode [126.774052] #PF: error_code(0x0000) - not-present page [126.774455] PGD 0 P4D 0 [126.774698] Oops: Oops: 0000 [#1] SMP NOPTI [126.775034] CPU: 13 UID: 0 PID: 1612 Comm: kworker/u64:55 Not tainted 6.14.0_blk+ #182 PREEMPT(full) [126.775676] Hardware name: QEMU Standard PC (Q35 + ICH9, 2009), BIOS 1.16.3-1.fc39 04/01/2014 [126.776275] Workqueue: iou_exit io_ring_exit_work [126.776651] RIP: 0010:ublk_io_release+0x14/0x130 [ublk_drv] Fixes it by always grabbing request reference for aborting the request.
CVE-2024-56544	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: udmabuf: change folios array from kmalloc to kvmalloc When PAGE_SIZE 4096, MAX_PAGE_ORDER 10, 64bit machine, page_alloc only support 4MB. If above this, trigger this warn and return NULL. udmabuf can change size limit, if change it to 3072(3GB), and then alloc 3GB udmabuf, will fail create. [4080.876581] ------------[cut here]------------ [4080.876843] WARNING: CPU: 3 PID: 2015 at mm/page_alloc.c:4556 __alloc_pages+0x2c8/0x350 [4080.878839] RIP: 0010:__alloc_pages+0x2c8/0x350 [4080.879470] Call Trace: [4080.879473] [4080.879473] ? __alloc_pages+0x2c8/0x350 [4080.879475] ? __warn.cold+0x8e/0xe8 [4080.880647] ? __alloc_pages+0x2c8/0x350 [4080.880909] ? report_bug+0xff/0x140 [4080.881175] ? handle_bug+0x3c/0x80 [4080.881556] ? exc_invalid_op+0x17/0x70 [4080.881559] ? asm_exc_invalid_op+0x1a/0x20 [4080.882077] ? udmabuf_create+0x131/0x400 Because MAX_PAGE_ORDER, kmalloc can max alloc 4096 * (1 << 10), 4MB memory, each array entry is pointer(8byte), so can save 524288 pages(2GB). Further more, costly order(order 3) may not be guaranteed that it can be applied for, due to fragmentation. This patch change udmabuf array use kvmalloc_array, this can fallback alloc into vmalloc, which can guarantee allocation for any size and does not affect the performance of kmalloc allocations.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2024-46698	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: video/aperture: optionally match the device in sysfb_disable() In aperture_remove_conflicting_pci_devices(), we currently only call sysfb_disable() on vga class devices. This leads to the following problem when the pimary device is not VGA compatible: 1. A PCI device with a non-VGA class is the boot display 2. That device is probed first and it is not a VGA device so sysfb_disable() is not called, but the device resources are freed by aperture_detach_platform_device() 3. Non-primary GPU has a VGA class and it ends up calling sysfb_disable() 4. NULL pointer dereference via sysfb_disable() since the resources have already been freed by aperture_detach_platform_device() when it was called by the other device. Fix this by passing a device pointer to sysfb_disable() and checking the device to determine if we should execute it or not. v2: Fix build when CONFIG_SCREEN_INFO is not set v3: Move device check into the mutex Drop primary variable in aperture_remove_conflicting_pci_devices() Drop __init on pci sysfb_pci_dev_is_enabled()

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-39850	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: vxlan: Fix NPD in {arp,neigh}_reduce() when using nexthop objects When the "proxy" option is enabled on a VXLAN device, the device will suppress ARP requests and IPv6 Neighbor Solicitation messages if it is able to reply on behalf of the remote host. That is, if a matching and valid neighbor entry is configured on the VXLAN device whose MAC address is not behind the "any" remote (0.0.0.0 / ::). The code currently assumes that the FDB entry for the neighbor's MAC address points to a valid remote destination, but this is incorrect if the entry is associated with an FDB nexthop group. This can result in a NPD [1][3] which can be reproduced using [2][4]. Fix by checking that the remote destination exists before dereferencing it. [1] BUG: kernel NULL pointer dereference, address: 0000000000000000 [...] CPU: 4 UID: 0 PID: 365 Comm: arping Not tainted 6.17.0-rc2-virtme-g2a89cb21162c #2 PREEMPT(voluntary) Hardware name: QEMU Standard PC (i440FX + PIIX, 1996), BIOS 1.17.0-4.fc41 04/01/2014 RIP: 0010:vxlan_xmit+0xb58/0x15f0 [...] Call Trace: dev_hard_start_xmit+0x5d/0x1c0 __dev_queue_xmit+0x246/0xfd0 packet_sendmsg+0x113a/0x1850 __sock_sendmsg+0x38/0x70 __sys_sendto+0x126/0x180 __x64_sys_sendto+0x24/0x30 do_syscall_64+0xa4/0x260 entry_SYSCALL_64_after_hwframe+0x4b/0x53 [2] #!/bin/bash ip address add 192.0.2.1/32 dev lo ip nexthop add id 1 via 192.0.2.2 fdb ip nexthop add id 10 group 1 fdb ip link add name vx0 up type vxlan id 10010 local 192.0.2.1 dstport 4789 proxy ip neigh add 192.0.2.3 lladdr 00:11:22:33:44:55 nud perm dev vx0 bridge fdb add 00:11:22:33:44:55 dev vx0 self static nhid 10 arping -b -c 1 -s 192.0.2.1 -I vx0 192.0.2.3 [3] BUG: kernel NULL pointer dereference, address: 0000000000000000 [...] CPU: 13 UID: 0 PID: 372 Comm: ndisc6 Not tainted 6.17.0-rc2-virtmne-g6ee90cb26014 #3 PREEMPT(voluntary) Hardware name: QEMU Standard PC (i440FX + PIIX, 1v996), BIOS 1.17.0-4.fc41 04/01/2x014 RIP: 0010:vxlan_xmit+0x803/0x1600 [...] Call Trace: dev_hard_start_xmit+0x5d/0x1c0 __dev_queue_xmit+0x246/0xfd0 ip6_finish_output2+0x210/0x6c0 ip6_finish_output+0x1af/0x2b0 ip6_mr_output+0x92/0x3e0 ip6_send_skb+0x30/0x90 rawv6_sendmsg+0xe6e/0x12e0 __sock_sendmsg+0x38/0x70 __sys_sendto+0x126/0x180 __x64_sys_sendto+0x24/0x30 do_syscall_64+0xa4/0x260 entry_SYSCALL_64_after_hwframe+0x4b/0x53 RIP: 0033:0x7f383422ec77 [4] #!/bin/bash ip address add 2001:db8:1::1/128 dev lo ip nexthop add id 1 via 2001:db8:1::1 fdb ip nexthop add id 10 group 1 fdb ip link add name vx0 up type vxlan id 10010 local 2001:db8:1::1 dstport 4789 proxy ip neigh add 2001:db8:1::3 lladdr 00:11:22:33:44:55 nud perm dev vx0 bridge fdb add 00:11:22:33:44:55 dev vx0 self static nhid 10 ndisc6 -r 1 -s 2001:db8:1::1 -w 1 2001:db8:1::3 vx0

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-39851	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: vxlan: Fix NPD when refreshing an FDB entry with a nexthop object VXLAN FDB entries can point to either a remote destination or an FDB nexthop group. The latter is usually used in EVPN deployments where learning is disabled. However, when learning is enabled, an incoming packet might try to refresh an FDB entry that points to an FDB nexthop group and therefore does not have a remote. Such packets should be dropped, but they are only dropped after dereferencing the non-existent remote, resulting in a NPD [1] which can be reproduced using [2]. Fix by dropping such packets earlier. Remove the misleading comment from first_remote_rcu(). [1] BUG: kernel NULL pointer dereference, address: 0000000000000000 [...] CPU: 13 UID: 0 PID: 361 Comm: mausezahn Not tainted 6.17.0-rc1-virtme-g9f6b606b6b37 #1 PREEMPT(voluntary) Hardware name: QEMU Standard PC (i440FX + PIIX, 1996), BIOS 1.17.0-4.fc41 04/01/2014 RIP: 0010:vxlan_snoop+0x98/0x1e0 [...] Call Trace: vxlan_encap_bypass+0x209/0x240 encap_bypass_if_local+0xb1/0x100 vxlan_xmit_one+0x1375/0x17e0 vxlan_xmit+0x6b4/0x15f0 dev_hard_start_xmit+0x5d/0x1c0 __dev_queue_xmit+0x246/0xfd0 packet_sendmsg+0x113a/0x1850 __sock_sendmsg+0x38/0x70 __sys_sendto+0x126/0x180 __x64_sys_sendto+0x24/0x30 do_syscall_64+0xa4/0x260 entry_SYSCALL_64_after_hwframe+0x4b/0x53 [2] #!/bin/bash ip address add 192.0.2.1/32 dev lo ip address add 192.0.2.2/32 dev lo ip nexthop add id 1 via 192.0.2.3 fdb ip nexthop add id 10 group 1 fdb ip link add name vx0 up type vxlan id 10010 local 192.0.2.1 dstport 12345 localbypass ip link add name vx1 up type vxlan id 10020 local 192.0.2.2 dstport 54321 learning bridge fdb add 00:11:22:33:44:55 dev vx0 self static dst 192.0.2.2 port 54321 vni 10020 bridge fdb add 00:aa:bb:cc:dd:ee dev vx1 self static nhid 10 mausezahn vx0 -a 00:aa:bb:cc:dd:ee -b 00:11:22:33:44:55 -c 1 -q
CVE-2025-38605	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: wifi: ath12k: Pass ab pointer directly to ath12k_dp_tx_get_encap_type() In ath12k_dp_tx_get_encap_type(), the arvif parameter is only used to retrieve the ab pointer. In vdev delete sequence the arvif->ar could become NULL and that would trigger kernel panic. Since the caller ath12k_dp_tx() already has a valid ab pointer, pass it directly to avoid panic and unnecessary dereferencing. PC points to "ath12k_dp_tx+0x228/0x988 [ath12k]" LR points to "ath12k_dp_tx+0xc8/0x988 [ath12k]". The Backtrace obtained is as follows: ath12k_dp_tx+0x228/0x988 [ath12k] ath12k_mac_tx_check_max_limit+0x608/0x920 [ath12k] ieee80211_process_measurement_req+0x320/0x348 [mac80211] ieee80211_tx_dequeue+0x9ac/0x1518 [mac80211] ieee80211_tx_dequeue+0xb14/0x1518 [mac80211] ieee80211_tx_prepare_skb+0x224/0x254 [mac80211] ieee80211_xmit+0xec/0x100 [mac80211] __ieee80211_subif_start_xmit+0xc50/0xf40 [mac80211] ieee80211_subif_start_xmit+0x2e8/0x308 [mac80211] netdev_start_xmit+0x150/0x18c dev_hard_start_xmit+0x74/0xc0 Tested-on: QCN9274 hw2.0 PCI WLAN.WBE.1.3.1-00173-QCAHKSWPL_SILICONZ-1

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38604	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: wifi: rtl818x: Kill URBs before clearing tx status queue In rtl8187_stop() move the call of usb_kill_anchored_urbs() before clearing b_tx_status.queue. This change prevents callbacks from using already freed skb due to anchor was not killed before freeing such skb. BUG: kernel NULL pointer dereference, address: 0000000000000080 #PF: supervisor read access in kernel mode #PF: error_code(0x0000) - not-present page PGD 0 P4D 0 Oops: Oops: 0000 [#1] SMP NOPTI CPU: 7 UID: 0 PID: 0 Comm: swapper/7 Not tainted 6.15.0 #8 PREEMPT(voluntary) Hardware name: QEMU Standard PC (i440FX + PIIX, 1996), BIOS 0.0.0 02/06/2015 RIP: 0010:ieee80211_tx_status_irqsafe+0x21/0xc0 [mac80211] Call Trace: rtl8187_tx_cb+0x116/0x150 [rtl8187] __usb_hcd_giveback_urb+0x9d/0x120 usb_giveback_urb_bh+0xbb/0x140 process_one_work+0x19b/0x3c0 bh_worker+0x1a7/0x210 tasklet_action+0x10/0x30 handle_softirqs+0xf0/0x340 __irq_exit_rcu+0xcd/0xf0 common_interrupt+0x85/0xa0 Tested on RTL8187BvE device. Found by Linux Verification Center (linuxtesting.org) with SVACE.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38646	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: wifi: rtw89: avoid NULL dereference when RX problematic packet on unsupported 6 GHz band With a quite rare chance, RX report might be problematic to make SW think a packet is received on 6 GHz band even if the chip does not support 6 GHz band actually. Since SW won't initialize stuffs for unsupported bands, NULL dereference will happen then in the sequence, rtw89_vif_rx_stats_iter() -> rtw89_core_cancel_6ghz_probe_tx(). So, add a check to avoid it. The following is a crash log for this case. BUG: kernel NULL pointer dereference, address: 0000000000000032 #PF: supervisor read access in kernel mode #PF: error_code(0x0000) - not-present page PGD 0 P4D 0 Oops: 0000 [#1] PREEMPT SMP NOPTI CPU: 1 PID: 1907 Comm: irq/131-rtw89_p Tainted: G U 6.6.56-05896-g89f5fb0eb30b #1 (HASH:1400 4) Hardware name: Google Telith/Telith, BIOS Google_Telith.15217.747.0 11/12/2024 RIP: 0010:rtw89_vif_rx_stats_iter+0xd2/0x310 [rtw89_core] Code: 4c 89 7d c8 48 89 55 c0 49 8d 44 24 02 48 89 45 b8 45 31 ff eb 11 41 c6 45 3a 01 41 b7 01 4d 8b 6d 00 4d 39 f5 74 42 8b 43 10 <41> 33 45 32 0f b7 4b 14 66 41 33 4d 36 0f b7 c9 09 c1 74 d8 4d 85 RSP: 0018:ffff9f3080138ca0 EFLAGS: 00010246 RAX: 00000000b8bf5770 RBX: ffff91b5e8c639c0 RCX: 0000000000000011 RDX: ffff91b582de1be8 RSI: 0000000000000000 RDI: ffff91b5e8c639e6 RBP: ffff9f3080138d00 R08: 0000000000000000 R09: 0000000000000000 R10: ffff91b59de70000 R11: ffffffffc069be50 R12: ffff91b5e8c639e4 R13: 0000000000000000 R14: ffff91b5828020b8 R15: 0000000000000000 FS: 0000000000000000(0000) GS:ffff91b8efa40000(0000) knlGS:0000000000000000 CS: 0010 DS: 0000 ES: 0000 CR0: 0000000080050033 CR2: 0000000000000032 CR3: 00000002bf838000 CR4: 0000000000750ee0 PKRU: 55555554 Call Trace: ? __die_body+0x68/0xb0 ? page_fault_oops+0x379/0x3e0 ? exc_page_fault+0x4f/0xa0 ? asm_exc_page_fault+0x22/0x30 ? __pfx_rtw89_vif_rx_stats_iter+0x10/0x10 [rtw89_core (HASH:1400 5)] ? rtw89_vif_rx_stats_iter+0xd2/0x310 [rtw89_core (HASH:1400 5)] __iterate_interfaces+0x59/0x110 [mac80211 (HASH:1400 6)] ? __pfx_rtw89_vif_rx_stats_iter+0x10/0x10 [rtw89_core (HASH:1400 5)] ? __pfx_rtw89_vif_rx_stats_iter+0x10/0x10 [rtw89_core (HASH:1400 5)] ieee80211_iterate_active_interfaces_atomic+0x36/0x50 [mac80211 (HASH:1400 6)] rtw89_core_rx_to_mac80211+0xfd/0x1b0 [rtw89_core (HASH:1400 5)] rtw89_core_rx+0x43a/0x980 [rtw89_core (HASH:1400 5)]

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38513	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: wifi: zd1211rw: Fix potential NULL pointer dereference in zd_mac_tx_to_dev() There is a potential NULL pointer dereference in zd_mac_tx_to_dev(). For example, the following is possible: T0 T1 zd_mac_tx_to_dev() /* len == skb_queue_len(q) */ while (len > ZD_MAC_MAX_ACK_WAITERS) { filter_ack() spin_lock_irqsave(&q->lock, flags); /* position == skb_queue_len(q) */ for (i=1; itype == NL80211_IFTYPE_AP) skb = __skb_dequeue(q); spin_unlock_irqrestore(&q->lock, flags); skb_dequeue() -> NULL Since there is a small gap between checking skb queue length and skb being unconditionally dequeued in zd_mac_tx_to_dev(), skb_dequeue() can return NULL. Then the pointer is passed to zd_mac_tx_status() where it is dereferenced. In order to avoid potential NULL pointer dereference due to situations like above, check if skb is not NULL before passing it to zd_mac_tx_status(). Found by Linux Verification Center (linuxtesting.org) with SVACE.
CVE-2025-38503	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: btrfs: fix assertion when building free space tree When building the free space tree with the block group tree feature enabled, we can hit an assertion failure like this: BTRFS info (device loop0 state M): rebuilding free space tree assertion failed: ret == 0, in fs/btrfs/free-space-tree.c:1102 ------------[cut here]------------ kernel BUG at fs/btrfs/free-space-tree.c:1102! Internal error: Oops - BUG: 00000000f2000800 [#1] SMP Modules linked in: CPU: 1 UID: 0 PID: 6592 Comm: syz-executor322 Not tainted 6.15.0-rc7-syzkaller-gd7fa1af5b33e #0 PREEMPT Hardware name: Google Google Compute Engine/Google Compute Engine, BIOS Google 05/07/2025 pstate: 60400005 (nZCv daif +PAN -UAO -TCO -DIT -SSBS BTYPE=--) pc : populate_free_space_tree+0x514/0x518 fs/btrfs/free-space-tree.c:1102 lr : populate_free_space_tree+0x514/0x518 fs/btrfs/free-space-tree.c:1102 sp : ffff8000a4ce7600 x29: ffff8000a4ce76e0 x28: ffff0000c9bc6000 x27: ffff0000ddfff3d8 x26: ffff0000ddfff378 x25: dfff800000000000 x24: 0000000000000001 x23: ffff8000a4ce7660 x22: ffff70001499cecc x21: ffff0000e1d8c160 x20: ffff0000e1cb7800 x19: ffff0000e1d8c0b0 x18: 00000000ffffffff x17: ffff800092f39000 x16: ffff80008ad27e48 x15: ffff700011e740c0 x14: 1ffff00011e740c0 x13: 0000000000000004 x12: ffffffffffffffff x11: ffff700011e740c0 x10: 0000000000ff0100 x9 : 94ef24f55d2dbc00 x8 : 94ef24f55d2dbc00 x7 : 0000000000000001 x6 : 0000000000000001 x5 : ffff8000a4ce6f98 x4 : ffff80008f415ba0 x3 : ffff800080548ef0 x2 : 0000000000000000 x1 : 0000000100000000 x0 : 000000000000003e Call trace: populate_free_space_tree+0x514/0x518 fs/btrfs/free-space-tree.c:1102 (P) btrfs_rebuild_free_space_tree+0x14c/0x54c fs/btrfs/free-space-tree.c:1337 btrfs_start_pre_rw_mount+0xa78/0xe10 fs/btrfs/disk-io.c:3074 btrfs_remount_rw fs/btrfs/super.c:1319 [inline] btrfs_reconfigure+0x828/0x2418 fs/btrfs/super.c:1543 reconfigure_super+0x1d4/0x6f0 fs/super.c:1083 do_remount fs/namespace.c:3365 [inline] path_mount+0xb34/0xde0 fs/namespace.c:4200 do_mount fs/namespace.c:4221 [inline] __do_sys_mount fs/namespace.c:4432 [inline] __se_sys_mount fs/namespace.c:4409 [inline] __arm64_sys_mount+0x3e8/0x468 fs/namespace.c:4409 __invoke_syscall arch/arm64/kernel/syscall.c:35 [inline] invoke_syscall+0x98/0x2b8 arch/arm64/kernel/syscall.c:49 el0_svc_common+0x130/0x23c arch/arm64/kernel/syscall.c:132 do_el0_svc+0x48/0x58 arch/arm64/kernel/syscall.c:151

CVE	Severity	CVSS	CVSS Vector	Component	Description
					el0_svc+0x58/0x17c arch/arm64/kernel/entry-common.c:767 el0t_64_sync_handler+0x78/0x108 arch/arm64/kernel/entry-common.c:786 el0t_64_sync+0x198/0x19c arch/arm64/kernel/entry.S:600 Code: f0047182 91178042 528089c3 9771d47b (d4210000) ---[end trace 0000000000000000]--- This happens because we are processing an empty block group, which has no extents allocated from it, there are no items for this block group, including the block group item since block group items are stored in a dedicated tree when using the block group tree feature. It also means this is the block group with the highest start offset, so there are no higher keys in the extent root, hence btrfs_search_slot_for_read() returns 1 (no higher key found). Fix this by asserting 'ret' is 0 only if the block group tree feature is not enabled, in which case we should find a block group item for the block group since it's stored in the extent root and block group item keys are greater than extent item keys (the value for BTRFS_BLOCK_GROUP_ITEM_KEY is 192 and for BTRFS_EXTENT_ITEM_KEY and BTRFS_METADATA_ITEM_KEY the values are 168 and 169 respectively). In case 'ret' is 1, we just need to add a record to the free space tree which spans the whole block group, and we can achieve this by making 'ret == 0' as the while loop's condition.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2024-42252	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: closures: Change BUG_ON() to WARN_ON() If a BUG_ON() can be hit in the wild, it shouldn't be a BUG_ON() For reference, this has popped up once in the CI, and we'll need more info to debug it: 03240 ------------[cut here]------------ 03240 kernel BUG at lib/closure.c:21! 03240 kernel BUG at lib/closure.c:21! 03240 Internal error: Oops - BUG: 00000000f2000800 [#1] SMP 03240 Modules linked in: 03240 CPU: 15 PID: 40534 Comm: kworker/u80:1 Not tainted 6.10.0-rc4-ktest-ga56da69799bd #25570 03240 Hardware name: linux,dummy-virt (DT) 03240 Workqueue: btree_update btree_interior_update_work 03240 pstate: 00001005 (nzcv daif -PAN -UAO -TCO -DIT +SSBS BTYPE=---) 03240 pc : closure_put+0x224/0x2a0 03240 lr : closure_put+0x24/0x2a0 03240 sp : ffff0000d12071c0 03240 x29: ffff0000d12071c0 x28: dfff800000000000 x27: ffff0000d1207360 03240 x26: 0000000000000040 x25: 0000000000000040 x24: 0000000000000040 03240 x23: ffff0000c1f20180 x22: 0000000000000000 x21: ffff0000c1f20168 03240 x20: 0000000040000000 x19: ffff0000c1f20140 x18: 0000000000000001 03240 x17: 0000000000003aa0 x16: 0000000000003ad0 x15: 1fffe0001c326974 03240 x14: 0000000000000a1e x13: 0000000000000000 x12: 1fffe000183e402d 03240 x11: ffff6000183e402d x10: dfff800000000000 x9 : ffff6000183e402e 03240 x8 : 0000000000000001 x7 : 00009fffe7c1bfd3 x6 : ffff0000c1f2016b 03240 x5 : ffff0000c1f20168 x4 : ffff6000183e402e x3 : ffff800081391954 03240 x2 : 0000000000000001 x1 : 0000000000000000 x0 : 00000000a8000000 03240 Call trace: 03240 closure_put+0x224/0x2a0 03240 bch2_check_for_deadlock+0x910/0x1028 03240 bch2_six_check_for_deadlock+0x1c/0x30 03240 six_lock_slowpath.isra.0+0x29c/0xed0 03240 six_lock_ip_waiter+0xa8/0xf8 03240 __bch2_btree_node_lock_write+0x14c/0x298 03240 bch2_trans_lock_write+0x6d4/0xb10 03240 __bch2_trans_commit+0x135c/0x5520 03240 btree_interior_update_work+0x1248/0x1c10 03240 process_scheduled_works+0x53c/0xd90 03240 worker_thread+0x370/0x8c8 03240 kthread+0x258/0x2e8 03240 ret_from_fork+0x10/0x20 03240 Code: aa1303e0 d63f0020 a94363f7 17ffff8c (d4210000) 03240 ---[end trace 0000000000000000]--- 03240 Kernel panic - not syncing: Oops - BUG: Fatal exception 03240 SMP: stopping secondary CPUs 03241 SMP: failed to stop secondary CPUs 13,15 03241 Kernel Offset: disabled 03241 CPU features: 0x00,00000003,80000008,4240500b 03241 Memory Limit: none 03241 ---[end Kernel panic - not syncing: Oops - BUG: Fatal exception]--- 03246 ========= FAILED TIMEOUT copygc_torture_no_checksum in 7200s
CVE-2025-38701	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ext4: do not BUG when INLINE_DATA_FL lacks system.data xattr A syzbot fuzzed image triggered a BUG_ON in ext4_update_inline_data() when an inode had the INLINE_DATA_FL flag set but was missing the system.data extended attribute. Since this can happen due to a maicioulyfuzzed file system, we shouldn't BUG, but rather, report it as a corrupted file system. Add similar replacements of BUG_ON with EXT4_ERROR_INODE() ii ext4_create_inline_data() and ext4_inline_data_truncate().

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38712	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: hfsplus: don't use BUG_ON() in hfsplus_create_attributes_file() When the volume header contains erroneous values that do not reflect the actual state of the filesystem, hfsplus_fill_super() assumes that the attributes file is not yet created, which later results in hitting BUG_ON() when hfsplus_create_attributes_file() is called. Replace this BUG_ON() with -EIO error with a message to suggest running fsck tool.
CVE-2025-71085	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ipv6: BUG() in pskb_expand_head() as part of calipso_skbuff_setattr() There exists a kernel oops caused by a BUG_ON(nhead < 0) at net/core/skbuff.c:2232 in pskb_expand_head(). This bug is triggered as part of the calipso_skbuff_setattr() routine when skb_cow() is passed headroom > INT_MAX (i.e. (int) (skb_headroom(skb) + len_delta) < 0). The root cause of the bug is due to an implicit integer cast in __skb_cow(). The check (headroom > skb_headroom(skb)) is meant to ensure that delta = headroom - skb_headroom(skb) is never negative, otherwise we will trigger a BUG_ON in pskb_expand_head(). However, if headroom > INT_MAX and delta <= -NET_SKB_PAD, the check passes, delta becomes negative, and pskb_expand_head() is passed a negative value for nhead. Fix the trigger condition in calipso_skbuff_setattr(). Avoid passing "negative" headroom sizes to skb_cow() within calipso_skbuff_setattr() by only using skb_cow() to grow headroom. PoC: Using `netlabelctl` tool: netlabelctl map del default netlabelctl calipso add pass doi:7 netlabelctl map add default address:0::1/128 protocol:calipso,7 Then run the following PoC: int fd = socket(AF_INET6, SOCK_DGRAM, IPPROTO_UDP); // setup msghdr int cmsg_size = 2; int cmsg_len = 0x60; struct msghdr msg; struct sockaddr_in6 dest_addr; struct cmsghdr * cmsg = (struct cmsghdr *) calloc(1, sizeof(struct cmsghdr) + cmsg_len); msg.msg_name = &dest_addr; msg.msg_namelen = sizeof(dest_addr); msg.msg_iov = NULL; msg.msg_iovlen = 0; msg.msg_control = cmsg; msg.msg_controllen = cmsg_len; msg.msg_flags = 0; // setup sockaddr dest_addr.sin6_family = AF_INET6; dest_addr.sin6_port = htons(31337); dest_addr.sin6_flowinfo = htonl(31337); dest_addr.sin6_addr = in6addr_loopback; dest_addr.sin6_scope_id = 31337; // setup cmsghdr cmsg->cmsg_len = cmsg_len; cmsg->cmsg_level = IPPROTO_IPV6; cmsg->cmsg_type = IPV6_HOPOPTS; char * hop_hdr = (char *)cmsg + sizeof(struct cmsghdr); hop_hdr[1] = 0x9; //set hop size - (0x9 + 1) * 8 = 80 sendmsg(fd, &msg, 0);
CVE-2026-22990	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: libceph: replace overzealous BUG_ON in osdmap_apply_incremental() If the osdmap is (maliciously) corrupted such that the incremental osdmap epoch is different from what is expected, there is no need to BUG. Instead, just declare the incremental osdmap to be invalid.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2026-23238	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: romfs: check sb_set_blocksize() return value romfs_fill_super() ignores the return value of sb_set_blocksize(), which can fail if the requested block size is incompatible with the block device's configuration. This can be triggered by setting a loop device's block size larger than PAGE_SIZE using ioctl(LOOP_SET_BLOCK_SIZE, 32768), then mounting a romfs filesystem on that device. When sb_set_blocksize(sb, ROMBSIZE) is called with ROMBSIZE=4096 but the device has logical_block_size=32768, bdev_validate_blocksize() fails because the requested size is smaller than the device's logical block size. sb_set_blocksize() returns 0 (failure), but romfs ignores this and continues mounting. The superblock's block size remains at the device's logical block size (32768). Later, when sb_bread() attempts I/O with this oversized block size, it triggers a kernel BUG in folio_set_bh(): kernel BUG at fs/buffer.c:1582! BUG_ON(size > PAGE_SIZE); Fix by checking the return value of sb_set_blocksize() and failing the mount with -EINVAL if it returns 0.
CVE-2025-38544	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: rxrpc: Fix bug due to prealloc collision When userspace is using AF_RXRPC to provide a server, it has to preallocate incoming calls and assign to them call IDs that will be used to thread related recvmsg() and sendmsg() together. The preallocated call IDs will automatically be attached to calls as they come in until the pool is empty. To the kernel, the call IDs are just arbitrary numbers, but userspace can use the call ID to hold a pointer to prepared structs. In any case, the user isn't permitted to create two calls with the same call ID (call IDs become available again when the call ends) and EBADSLT should result from sendmsg() if an attempt is made to preallocate a call with an in-use call ID. However, the cleanup in the error handling will trigger both assertions in rxrpc_cleanup_call() because the call isn't marked complete and isn't marked as having been released. Fix this by setting the call state in rxrpc_service_prealloc_one() and then marking it as being released before calling the cleanup function.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-39801	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: usb: dwc3: Remove WARN_ON for device endpoint command timeouts This commit addresses a rarely observed endpoint command timeout which causes kernel panic due to warn when 'panic_on_warn' is enabled and unnecessary call trace prints when 'panic_on_warn' is disabled. It is seen during fast software-controlled connect/disconnect testcases. The following is one such endpoint command timeout that we observed: 1. Connect ======= ->dwc3_thread_interrupt - >dwc3_ep0_interrupt - >configfs_composite_setup ->composite_setup ->usb_ep_queue ->dwc3_gadget_ep0_queue - >__dwc3_gadget_ep0_queue - >__dwc3_ep0_do_control_data - >dwc3_send_gadget_ep_cmd 2. Disconnect ========== ->dwc3_thread_interrupt - >dwc3_gadget_disconnect_interrupt - >dwc3_ep0_reset_state - >dwc3_ep0_end_control_data - >dwc3_send_gadget_ep_cmd In the issue scenario, in Exynos platforms, we observed that control transfers for the previous connect have not yet been completed and end transfer command sent as a part of the disconnect sequence and processing of USB_ENDPOINT_HALT feature request from the host timeout. This maybe an expected scenario since the controller is processing EP commands sent as a part of the previous connect. It maybe better to remove WARN_ON in all places where device endpoint commands are sent to avoid unnecessary kernel panic due to warn.
CVE-2025-38335	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: Input: gpio-keys - fix a sleep while atomic with PREEMPT_RT When enabling PREEMPT_RT, the gpio_keys_irq_timer() callback runs in hard irq context, but the input_event() takes a spin_lock, which isn't allowed there as it is converted to a rt_spin_lock(). [4054.289999] BUG: sleeping function called from invalid context at kernel/locking/spinlock_rt.c:48 [4054.290028] in_atomic(): 1, irqs_disabled(): 1, non_block: 0, pid: 0, name: swapper/0 ... [4054.290195] __might_resched+0x13c/0x1f4 [4054.290209] rt_spin_lock+0x54/0x11c [4054.290219] input_event+0x48/0x80 [4054.290230] gpio_keys_irq_timer+0x4c/0x78 [4054.290243] __hrtimer_run_queues+0x1a4/0x438 [4054.290257] hrtimer_interrupt+0xe4/0x240 [4054.290269] arch_timer_handler_phys+0x2c/0x44 [4054.290283] handle_percpu_devid_irq+0x8c/0x14c [4054.290297] handle_irq_desc+0x40/0x58 [4054.290307] generic_handle_domain_irq+0x1c/0x28 [4054.290316] gic_handle_irq+0x44/0xcc Considering the gpio_keys_irq_isr() can run in any context, e.g. it can be threaded, it seems there's no point in requesting the timer isr to run in hard irq context. Relax the hrtimer not to use the hard context.
CVE-2025-71104	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: KVM: x86: Fix VM hard lockup after prolonged inactivity with periodic HV timer When advancing the target expiration for the guest's APIC timer in periodic mode, set the expiration to "now" if the target expiration is in the past (similar to what is done in update_target_expiration()). Blindly adding the period to the previous target expiration can result in KVM generating a practically unbounded number of hrtimer IRQs due to programming an expired timer over and over. In extreme scenarios, e.g. if userspace pauses/suspends a VM for an extended duration, this can even cause hard lockups in the host. Currently, the bug only affects Intel CPUs when

CVE	Severity	CVSS	CVSS Vector	Component	Description
					using the hypervisor timer (HV timer), a.k.a. the VMX preemption timer. Unlike the software timer, a.k.a. hrtimer, which KVM keeps running even on exits to userspace, the HV timer only runs while the guest is active. As a result, if the vCPU does not run for an extended duration, there will be a huge gap between the target expiration and the current time the vCPU resumes running. Because the target expiration is incremented by only one period on each timer expiration, this leads to a series of timer expirations occurring rapidly after the vCPU/VM resumes. More critically, when the vCPU first triggers a periodic HV timer expiration after resuming, advancing the expiration by only one period will result in a target expiration in the past. As a result, the delta may be calculated as a negative value. When the delta is converted into an absolute value (tscdeadline is an unsigned u64), the resulting value can overflow what the HV timer is capable of programming. I.e. the large value will exceed the VMX Preemption Timer's maximum bit width of cpu_preemption_timer_multi + 32, and thus cause KVM to switch from the HV timer to the software timer (hrtimers). After switching to the software timer, periodic timer expiration callbacks may be executed consecutively within a single clock interrupt handler, because hrtimers honors KVM's request for an expiration in the past and immediately re-invokes KVM's callback after reprogramming. And because the interrupt handler runs with IRQs disabled, restarting KVM's hrtimer over and over until the target expiration is advanced to "now" can result in a hard lockup. E.g. the following hard lockup was triggered in the host when running a Windows VM (only relevant because it used the APIC timer in periodic mode) after resuming the VM from a long suspend (in the host). NMI watchdog: Watchdog detected hard LOCKUP on cpu 45 ... RIP: 0010:advance_periodic_target_expiration+0x4d/0x80 [kvm] ... RSP: 0018:ff4f88f5d98d8ef0 EFLAGS: 00000046 RAX: fff0103f91be678e RBX: fff0103f91be678e RCX: 00843a7d9e127bcc RDX: 0000000000000002 RSI: 0052ca4003697505 RDI: ff440d5bfbdbd500 RBP: ff440d5956f99200 R08: ff2ff2a42deb6a84 R09: 000000000002a6c0 R10: 0122d794016332b3 R11: 0000000000000000 R12: ff440db1af39cfc0 R13: ff440db1af39cfc0 R14: ffffffffc0d4a560 R15: ff440db1af39d0f8 FS: 00007f04a6ffd700(0000) GS:ff440db1af380000(0000) knlGS:000000e38a3b8000 CS: 0010 DS: 0000 ES: 0000 CR0: 0000000080050033 CR2: 000000d5651feff8 CR3: 000000684e038002 CR4: 0000000000773ee0 PKRU: 55555554 Call Trace: apic_timer_fn+0x31/0x50 [kvm] __hrtimer_run_queues+0x100/0x280 hrtimer_interrupt+0x100/0x210 ? ttwu_do_wakeup+0x19/0x160 smp_apic_timer_interrupt+0x6a/0x130 apic_timer_interrupt+0xf/0x20 Moreover, if the suspend duration of the virtual machine is not long enough to trigger a hard lockup in this scenario, since commit 98c25ead5eda ("KVM: VMX: Move preemption timer <=> hrtimer dance to common x86"), KVM will continue using the software timer until the guest reprograms the APIC timer in some way. Since the periodic timer does not require frequent APIC timer register programming, the guest may continue to use the software timer in ---truncated---

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-39767	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: LoongArch: Optimize module load time by optimizing PLT/GOT counting When enabling CONFIG_KASAN, CONFIG_PREEMPT_VOLUNTARY_BUILD and CONFIG_PREEMPT_VOLUNTARY at the same time, there will be soft deadlock, the relevant logs are as follows: rcu: INFO: rcu_sched self-detected stall on CPU ... Call Trace: [<9000000000024f9e4>] show_stack+0x5c/0x180 [<90000000002482f4>] dump_stack_lvl+0x94/0xbc [<9000000000224544>] rcu_dump_cpu_stacks+0x1fc/0x280 [<900000000037ac80>] rcu_sched_clock_irq+0x720/0xf88 [<9000000000396c34>] update_process_times+0xb4/0x150 [<90000000003b2474>] tick_nohz_handler+0xf4/0x250 [<9000000000397e28>] __hrtimer_run_queues+0x1d0/0x428 [<9000000000399b2c>] hrtimer_interrupt+0x214/0x538 [<9000000000253634>] constant_timer_interrupt+0x64/0x80 [<9000000000349938>] __handle_irq_event_percpu+0x78/0x1a0 [<9000000000349a78>] handle_irq_event_percpu+0x18/0x88 [<9000000000354c00>] handle_percpu_irq+0x90/0xf0 [<9000000000348c74>] handle_irq_desc+0x94/0xb8 [<9000000001012b28>] handle_cpu_irq+0x68/0xa0 [<9000000001def8c0>] handle_loongarch_irq+0x30/0x48 [<9000000001def958>] do_vint+0x80/0xd0 [<9000000000268a0c>] kasan_mem_to_shadow.part.0+0x2c/0x2a0 [<90000000006344f4>] __asan_load8+0x4c/0x120 [<900000000025c0d0>] module_frob_arch_sections+0x5c8/0x6b8 [<90000000003895f0>] load_module+0x9e0/0x2958 [<900000000038b770>] __do_sys_init_module+0x208/0x2d0 [<9000000001df0c34>] do_syscall+0x94/0x190 [<900000000024d6fc>] handle_syscall+0xbc/0x158 After analysis, this is because the slow speed of loading the amdgpu module leads to the long time occupation of the cpu and then the soft deadlock. When loading a module, module_frob_arch_sections() tries to figure out the number of PLTs/GOTs that will be needed to handle all the RELAs. It will call the count_max_entries() to find in an out-of-order date which counting algorithm has O(n^2) complexity. To make it faster, we sort the relocation list by info and addend. That way, to check for a duplicate relocation, it just needs to compare with the previous entry. This reduces the complexity of the algorithm to O(n log n), as done in commit d4e0340919fb ("arm64/module: Optimize module load time by optimizing PLT counting"). This gives sinificant reduction in module load time for modules with large number of relocations. After applying this patch, the soft deadlock problem has been solved, and the kernel starts normally without "Call Trace". Using the default configuration to test some modules, the results are as follows: Module Size ip_tables 36K fat 143K radeon 2.5MB amdgpu 16MB Without this patch: Module Module load time (ms) Count(PLTs/GOTs) ip_tables 18 59/6 fat 0 162/14 radeon 54 1221/84 amdgpu 1411 4525/1098 With this patch: Module Module load time (ms) Count(PLTs/GOTs) ip_tables 18 59/6 fat 0 162/14 radeon 22 1221/84 amdgpu 45 4525/1098
CVE-2025-	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal	In the Linux kernel, the following vulnerability has

37745 CVE	Severity	CVSS	CVSS Vector	6.6.96+git Component	been resolved: PM: hibernate: Avoid deadlock in hibernate_compressor_param_set() syzbot Description
					reported a deadlock in lock_system_sleep() (see below). The write operation to "/sys/module/hibernate/parameters/compressor" conflicts with the registration of ieee80211 device, resulting in a deadlock when attempting to acquire system_transition_mutex under param_lock. To avoid this deadlock, change hibernate_compressor_param_set() to use mutex_trylock() for attempting to acquire system_transition_mutex and return -EBUSY when it fails. Task flags need not be saved or adjusted before calling mutex_trylock(&system_transition_mutex) because the caller is not going to end up waiting for this mutex and if it runs concurrently with system suspend in progress, it will be frozen properly when it returns to user space. syzbot report: syz-executor895/5833 is trying to acquire lock: ffffffff8e0828c8 (system_transition_mutex) {+.+.}-{4:4}, at: lock_system_sleep+0x87/0xa0 kernel/power/main.c:56 but task is already holding lock: ffffffff8e07dc68 (param_lock){+.+.}-{4:4}, at: kernel_param_lock kernel/params.c:607 [inline] ffffffff8e07dc68 (param_lock){+.+.}-{4:4}, at: param_attr_store+0xe6/0x300 kernel/params.c:586 which lock already depends on the new lock. the existing dependency chain (in reverse order) is: -> #3 (param_lock){+.+.}-{4:4}: __mutex_lock_common kernel/locking/mutex.c:585 [inline] __mutex_lock+0x19b/0xb10 kernel/locking/mutex.c:730 ieee80211_rate_control_ops_get net/mac80211/rate.c:220 [inline] rate_control_alloc net/mac80211/rate.c:266 [inline] ieee80211_init_rate_ctrl_alg+0x18d/0x6b0 net/mac80211/rate.c:1015 ieee80211_register_hw+0x20cd/0x4060 net/mac80211/main.c:1531 mac80211_hwsim_new_radio+0x304e/0x54e0 drivers/net/wireless/virtual/mac80211_hwsim.c:5558 init_mac80211_hwsim+0x432/0x8c0 drivers/net/wireless/virtual/mac80211_hwsim.c:6910 do_one_initcall+0x128/0x700 init/main.c:1257 do_initcall_level init/main.c:1319 [inline] do_initcalls init/main.c:1335 [inline] do_basic_setup init/main.c:1354 [inline] kernel_init_freeable+0x5c7/0x900 init/main.c:1568 kernel_init+0x1c/0x2b0 init/main.c:1457 ret_from_fork+0x45/0x80 arch/x86/kernel/process.c:148 ret_from_fork_asm+0x1a/0x30 arch/x86/entry/entry_64.S:244 -> #2 (rtnl_mutex) {+.+.}-{4:4}: __mutex_lock_common kernel/locking/mutex.c:585 [inline] __mutex_lock+0x19b/0xb10 kernel/locking/mutex.c:730 wg_pm_notification drivers/net/wireguard/device.c:80 [inline] wg_pm_notification+0x49/0x180 drivers/net/wireguard/device.c:64 notifier_call_chain+0xb7/0x410 kernel/notifier.c:85 notifier_call_chain_robust kernel/notifier.c:120 [inline] blocking_notifier_call_chain_robust kernel/notifier.c:345 [inline] blocking_notifier_call_chain_robust+0xc9/0x170 kernel/notifier.c:333 pm_notifier_call_chain_robust+0x27/0x60 kernel/power/main.c:102 snapshot_open+0x189/0x2b0 kernel/power/user.c:77 misc_open+0x35a/0x420 drivers/char/misc.c:179 chrdev_open+0x237/0x6a0 fs/char_dev.c:414 do_dentry_open+0x735/0x1c40 fs/open.c:956 vfs_open+0x82/0x3f0 fs/open.c:1086 do_open fs/namei.c:3830 [inline] path_openat+0x1e88/0x2d80 fs/namei.c:3989 do_filp_open+0x20c/0x470 fs/namei.c:4016 do_sys_openat2+0x17a/0x1e0 fs/open.c:1428 do_sys_open fs/open.c:1443 [inline] __do_sys_openat fs/open.c:1459 [inline] __se_sys_openat fs/open.c:1454 [inline] __x64_sys_openat+0x175/0x210 fs/open.c:1454 do_syscall_x64 arch/x86/entry/common.c:52

CVE	Severity	CVSS	CVSS Vector	Component	Description
					[inline] do_syscall_64+0xcd/0x250 arch/x86/entry/common.c:83 entry_SYSCALL_64_after_hwframe+0x77/0x7f -> #1 ((pm_chain_head).rwsem){++++}-{4:4}: down_read+0x9a/0x330 kernel/locking/rwsem.c:1524 blocking_notifier_call_chain_robust kerne --- truncated---
CVE-2025-37907	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: accel/ivpu: Fix locking order in ivpu_job_submit Fix deadlock in job submission and abort handling. When a thread aborts currently executing jobs due to a fault, it first locks the global lock protecting submitted_jobs (#1). After the last job is destroyed, it proceeds to release the related context and locks file_priv (#2). Meanwhile, in the job submission thread, the file_priv lock (#2) is taken first, and then the submitted_jobs lock (#1) is obtained when a job is added to the submitted jobs list. CPU0 CPU1 ---- ---- (for example due to a fault) (jobs submissions keep coming) lock(&vdev->submitted_jobs_lock) #1 ivpu_jobs_abort_all() job_destroy() lock(&file_priv->lock) #2 lock(&vdev->submitted_jobs_lock) #1 file_priv_release() lock(&vdev->context_list_lock) lock(&file_priv->lock) #2 This order of locking causes a deadlock. To resolve this issue, change the order of locking in ivpu_job_submit().

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38520	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: drm/amdkfd: Don't call mmput from MMU notifier callback If the process is exiting, the mmput inside mmu notifier callback from compactd or fork or numa balancing could release the last reference of mm struct to call exit_mmap and free_pgtable, this triggers deadlock with below backtrace. The deadlock will leak kfd process as mmu notifier release is not called and cause VRAM leaking. The fix is to take mm reference mmget_non_zero when adding prange to the deferred list to pair with mmput in deferred list work. If prange split and add into pchild list, the pchild work_item.mm is not used, so remove the mm parameter from svm_range_unmap_split and svm_range_add_child. The backtrace of hung task: INFO: task python:348105 blocked for more than 64512 seconds. Call Trace: __schedule+0x1c3/0x550 schedule+0x46/0xb0 rwsem_down_write_slowpath+0x24b/0x4c0 unlink_anon_vmas+0xb1/0x1c0 free_pgtables+0xa9/0x130 exit_mmap+0xbc/0x1a0 mmput+0x5a/0x140 svm_range_cpu_invalidate_pagetables+0x2b/0x40 [amdgpu] mn_itree_invalidate+0x72/0xc0 __mmu_notifier_invalidate_range_start+0x48/0x60 try_to_unmap_one+0x10fa/0x1400 rmap_walk_anon+0x196/0x460 try_to_unmap+0xbb/0x210 migrate_page_unmap+0x54d/0x7e0 migrate_pages_batch+0x1c3/0xae0 migrate_pages_sync+0x98/0x240 migrate_pages+0x25c/0x520 compact_zone+0x29d/0x590 compact_zone_order+0xb6/0xf0 try_to_compact_pages+0xbe/0x220 __alloc_pages_direct_compact+0x96/0x1a0 __alloc_pages_slowpath+0x410/0x930 __alloc_pages_nodemask+0x3a9/0x3e0 do_huge_pmd_anonymous_page+0xd7/0x3e0 __handle_mm_fault+0x5e3/0x5f0 handle_mm_fault+0xf7/0x2e0 hmm_vma_fault.isra.0+0x4d/0xa0 walk_pmd_range.isra.0+0xa8/0x310 walk_pud_range+0x167/0x240 walk_pgd_range+0x55/0x100 __walk_page_range+0x87/0x90 walk_page_range+0xf6/0x160 hmm_range_fault+0x4f/0x90 amdgpu_hmm_range_get_pages+0x123/0x230 [amdgpu] amdgpu_ttm_tt_get_user_pages+0xb1/0x150 [amdgpu] init_user_pages+0xb1/0x2a0 [amdgpu] amdgpu_amdkfd_gpuvm_alloc_memory_of_gpu+0x543/0x7d0 [amdgpu] kfd_ioctl_alloc_memory_of_gpu+0x24c/0x4e0 [amdgpu] kfd_ioctl+0x29d/0x500 [amdgpu] (cherry picked from commit a29e067bd38946f752b0ef855f3dfff87e77bec7)
CVE-2025-68223	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: drm/radeon: delete radeon_fence_process in is_signaled, no deadlock Delete the attempt to progress the queue when checking if fence is signaled. This avoids deadlock. dma-fence_ops::signaled can be called with the fence lock in unknown state. For radeon, the fence lock is also the wait queue lock. This can cause a self deadlock when signaled() tries to make forward progress on the wait queue. But advancing the queue is unneeded because incorrectly returning false from signaled() is perfectly acceptable. (cherry picked from commit 527ba26e50ec2ca2be9c7c82f3ad42998a75d0db)

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2024-44956	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: drm/xe/preempt_fence: enlarge the fence critical section It is really easy to introduce subtle deadlocks in preempt_fence_work_func() since we operate on single global ordered-wq for signalling our preempt fences behind the scenes, so even though we signal a particular fence, everything in the callback should be in the fence critical section, since blocking in the callback will prevent other published fences from signalling. If we enlarge the fence critical section to cover the entire callback, then lockdep should be able to understand this better, and complain if we grab a sensitive lock like vm->lock, which is also held when waiting on preempt fences.
CVE-2024-47736	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: erofs: handle overlapped pclusters out of crafted images properly syzbot reported a task hang issue due to a deadlock case where it is waiting for the folio lock of a cached folio that will be used for cache I/Os. After looking into the crafted fuzzed image, I found it's formed with several overlapped big pclusters as below: Ext: logical offset | length : physical offset | length 0: 0.. 16384 | 16384 : 151552.. 167936 | 16384 1: 16384.. 32768 | 16384 : 155648.. 172032 | 16384 2: 32768.. 49152 | 16384 : 537223168.. 537239552 | 16384 ... Here, extent 0/1 are physically overlapped although it's entirely _impossible_ for normal filesystem images generated by mkfs. First, managed folios containing compressed data will be marked as up-to-date and then unlocked immediately (unlike in-place folios) when compressed I/Os are complete. If physical blocks are not submitted in the incremental order, there should be separate BIOs to avoid dependency issues. However, the current code mis-arranges z_erofs_fill_bio_vec() and BIO submission which causes unexpected BIO waits. Second, managed folios will be connected to their own pclusters for efficient inter-queries. However, this is somewhat hard to implement easily if overlapped big pclusters exist. Again, these only appear in fuzzed images so let's simply fall back to temporary short-lived pages for correctness. Additionally, it justifies that referenced managed folios cannot be truncated for now and reverts part of commit 2080ca1ed3e4 ("erofs: tidy up `struct z_erofs_bvec`") for simplicity although it shouldn't be any difference.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-22127	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: f2fs: fix potential deadloop in prepare_compress_overwrite() Jan Prusakowski reported a kernel hang issue as below: When running xfstests on linux-next kernel (6.14.0-rc3, 6.12) I encountered a problem in generic/475 test where fsstress process gets blocked in __f2fs_write_data_pages() and the test hangs. The options I used are: MKFS_OPTIONS -- -O compression -O extra_attr -O project_quota -O quota /dev/vdc MOUNT_OPTIONS -- -o acl,user_xattr -o discard,compress_extension=* /dev/vdc /vdc INFO: task kworker/u8:0:11 blocked for more than 122 seconds. Not tainted 6.14.0-rc3-xfstests-lockdep #1 "echo 0 > /proc/sys/kernel/hung_task_timeout_secs" disables this message. task:kworker/u8:0 state:D stack:0 pid:11 tgid:11 ppid:2 task_flags:0x4208160 flags:0x00004000 Workqueue: writeback wb_workfn (flush-253:0) Call Trace: __schedule+0x309/0x8e0 schedule+0x3a/0x100 schedule_preempt_disabled+0x15/0x30 __mutex_lock+0x59a/0xdb0 __f2fs_write_data_pages+0x3ac/0x400 do_writepages+0xe8/0x290 __writeback_single_inode+0x5c/0x360 writeback_sb_inodes+0x22f/0x570 wb_writeback+0xb0/0x410 wb_do_writeback+0x47/0x2f0 wb_workfn+0x5a/0x1c0 process_one_work+0x223/0x5b0 worker_thread+0x1d5/0x3c0 kthread+0xfd/0x230 ret_from_fork+0x31/0x50 ret_from_fork_asm+0x1a/0x30 The root cause is: once generic/475 starts toload error table to dm device, f2fs_prepare_compress_overwrite() will loop reading compressed cluster pages due to IO error, meanwhile it has held .writepages lock, it can block all other writeback tasks. Let's fix this issue w/ below changes: - add f2fs_handle_page_eio() in prepare_compress_overwrite() to detect IO error. - detect cp_error earler in f2fs_read_multi_pages().
CVE-2024-26686	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: fs/proc: do_task_stat: use sig->stats_lock to gather the threads/children stats lock_task_sighand() can trigger a hard lockup. If NR_CPUS threads call do_task_stat() at the same time and the process has NR_THREADS, it will spin with irqs disabled O(NR_CPUS * NR_THREADS) time. Change do_task_stat() to use sig->stats_lock to gather the statistics outside of ->siglock protected section, in the likely case this code will run lockless.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38650	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: hfsplus: remove mutex_lock check in hfsplus_free_extents Syzbot reported an issue in hfsplus filesystem: ------------[cut here]------------ WARNING: CPU: 0 PID: 4400 at fs/hfsplus/extents.c:346 hfsplus_free_extents+0x700/0xad0 Call Trace: hfsplus_file_truncate+0x768/0xbb0 fs/hfsplus/extents.c:606 hfsplus_write_begin+0xc2/0xd0 fs/hfsplus/inode.c:56 cont_expand_zero fs/buffer.c:2383 [inline] cont_write_begin+0x2cf/0x860 fs/buffer.c:2446 hfsplus_write_begin+0x86/0xd0 fs/hfsplus/inode.c:52 generic_cont_expand_simple+0x151/0x250 fs/buffer.c:2347 hfsplus_setattr+0x168/0x280 fs/hfsplus/inode.c:263 notify_change+0xe38/0x10f0 fs/attr.c:420 do_truncate+0x1fb/0x2e0 fs/open.c:65 do_sys_ftruncate+0x2eb/0x380 fs/open.c:193 do_syscall_x64 arch/x86/entry/common.c:50 [inline] do_syscall_64+0x3d/0xb0 arch/x86/entry/common.c:80 entry_SYSCALL_64_after_hwframe+0x63/0xcd To avoid deadlock, Commit 31651c607151 ("hfsplus: avoid deadlock on file truncation") unlock extree before hfsplus_free_extents(), and add check wheather extree is locked in hfsplus_free_extents(). However, when operations such as hfsplus_file_release, hfsplus_setattr, hfsplus_unlink, and hfsplus_get_block are executed concurrently in different files, it is very likely to trigger the WARN_ON, which will lead syzbot and xfstest to consider it as an abnormality. The comment above this warning also describes one of the easy triggering situations, which can easily trigger and cause xfstest&syzbot to report errors. [task A] [task B] ->hfsplus_file_release ->hfsplus_file_truncate ->hfs_find_init ->mutex_lock ->mutex_unlock ->hfsplus_write_begin ->hfsplus_get_block ->hfsplus_file_extend ->hfsplus_ext_read_extent ->hfs_find_init ->mutex_lock ->hfsplus_free_extents WARN_ON(mutex_is_locked) !!! Several threads could try to lock the shared extents tree. And warning can be triggered in one thread when another thread has locked the tree. This is the wrong behavior of the code and we need to remove the warning.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38311	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: iavf: get rid of the crit lock Get rid of the crit lock. That frees us from the error prone logic of try_locks. Thanks to netdev_lock() by Jakub it is now easy, and in most cases we were protected by it already - replace crit lock by netdev lock when it was not the case. Lockdep reports that we should cancel the work under crit_lock [splat1], and that was the scheme we have mostly followed since [1] by Slawomir. But when that is done we still got into deadlocks [splat2]. So instead we should look at the bigger problem, namely "weird locking/scheduling" of the iavf. The first step to fix that is to remove the crit lock. I will followup with a -next series that simplifies scheduling/tasks. Cancel the work without netdev lock (weird unlock+lock scheme), to fix the [splat2] (which would be totally ugly if we would kept the crit lock). Extend protected part of iavf_watchdog_task() to include scheduling more work. Note that the removed comment in iavf_reset_task() was misplaced, it belonged to inside of the removed if condition, so it's gone now. [splat1] - w/o this patch - The deadlock during VF removal: WARNING: possible circular locking dependency detected sh/3825 is trying to acquire lock: ((work_completion)(&(&adapter->watchdog_task)->work)){+.+.}-{0:0}, at: start_flush_work+0x1a1/0x470 but task is already holding lock: (&adapter->crit_lock){+.+.}-{4:4}, at: iavf_remove+0xd1/0x690 [iavf] which lock already depends on the new lock. [splat2] - when cancelling work under crit lock, w/o this series, see [2] for the band aid attempt WARNING: possible circular locking dependency detected sh/3550 is trying to acquire lock: ((wq_completion)iavf){+.+.}-{0:0}, at: touch_wq_lockdep_map+0x26/0x90 but task is already holding lock: (&dev->lock){+.+.}-{4:4}, at: iavf_remove+0xa6/0x6e0 [iavf] which lock already depends on the new lock. [1] fc2e6b3b132a ("iavf: Rework mutexes for better synchronisation") [2] https://github.com/pkitszel/linux/commit/52dddbfc2bb60294083f5711a158a
CVE-2025-37968	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: iio: light: opt3001: fix deadlock due to concurrent flag access The threaded IRQ function in this driver is reading the flag twice: once to lock a mutex and once to unlock it. Even though the code setting the flag is designed to prevent it, there are subtle cases where the flag could be true at the mutex_lock stage and false at the mutex_unlock stage. This results in the mutex not being unlocked, resulting in a deadlock. Fix it by making the opt3001_irq() code generally more robust, reading the flag into a variable and using the variable value at both stages.
CVE-2024-41080	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: io_uring: fix possible deadlock in io_register_iowq_max_workers() The io_register_iowq_max_workers() function calls io_put_sq_data(), which acquires the sqd->lock without releasing the uring_lock. Similar to the commit 009ad9f0c6ee ("io_uring: drop ctx->uring_lock before acquiring sqd->lock"), this can lead to a potential deadlock situation. To resolve this issue, the uring_lock is released before calling io_put_sq_data(), and then it is re-acquired after the function call. This change ensures that the locks are acquired in the correct order, preventing the possibility of a deadlock.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2026-23103	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ipvlan: Make the addrs_lock be per port Make the addrs_lock be per port, not per ipvlan dev. Initial code seems to be written in the assumption, that any address change must occur under RTNL. But it is not so for the case of IPv6. So 1) Introduce per-port addrs_lock. 2) It was needed to fix places where it was forgotten to take lock (ipvlan_open/ipvlan_close) This appears to be a very minor problem though. Since it's highly unlikely that ipvlan_add_addr() will be called on 2 CPU simultaneously. But nevertheless, this could cause: 1) False-negative of ipvlan_addr_busy(): one interface iterated through all port->ipvlans + ipvlan->addrs under some ipvlan spinlock, and another added IP under its own lock. Though this is only possible for IPv6, since looks like only ipvlan_addr6_event() can be called without rtnl_lock. 2) Race since ipvlan_ht_addr_add(port) is called under different ipvlan->addrs_lock locks This should not affect performance, since add/remove IP is a rare situation and spinlock is not taken on fast paths.
CVE-2025-39782	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: jbd2: prevent softlockup in jbd2_log_do_checkpoint() Both jbd2_log_do_checkpoint() and jbd2_journal_shrink_checkpoint_list() periodically release j_list_lock after processing a batch of buffers to avoid long hold times on the j_list_lock. However, since both functions contend for j_list_lock, the combined time spent waiting and processing can be significant. jbd2_journal_shrink_checkpoint_list() explicitly calls cond_resched() when need_resched() is true to avoid softlockups during prolonged operations. But jbd2_log_do_checkpoint() only exits its loop when need_resched() is true, relying on potentially sleeping functions like __flush_batch() or wait_on_buffer() to trigger rescheduling. If those functions do not sleep, the kernel may hit a softlockup. watchdog: BUG: soft lockup - CPU#3 stuck for 156s! [kworker/u129:2:373] CPU: 3 PID: 373 Comm: kworker/u129:2 Kdump: loaded Not tainted 6.6.0+ #10 Hardware name: Huawei TaiShan 2280 /BC11SPCD, BIOS 1.27 06/13/2017 Workqueue: writeback wb_workfn (flush-7:2) pstate: 20000005 (nzCv daif -PAN -UAO -TCO -DIT -SSBS BTYPE=--) pc : native_queued_spin_lock_slowpath+0x358/0x418 lr : jbd2_log_do_checkpoint+0x31c/0x438 [jbd2] Call trace: native_queued_spin_lock_slowpath+0x358/0x418 jbd2_log_do_checkpoint+0x31c/0x438 [jbd2] __jbd2_log_wait_for_space+0xfc/0x2f8 [jbd2] add_transaction_credits+0x3bc/0x418 [jbd2] start_this_handle+0xf8/0x560 [jbd2] jbd2__journal_start+0x118/0x228 [jbd2] __ext4_journal_start_sb+0x110/0x188 [ext4] ext4_do_writepages+0x3dc/0x740 [ext4] ext4_writepages+0xa4/0x190 [ext4] do_writepages+0x94/0x228 __writeback_single_inode+0x48/0x318 writeback_sb_inodes+0x204/0x590 __writeback_inodes_wb+0x54/0xf8 wb_writeback+0x2cc/0x3d8 wb_do_writeback+0x2e0/0x2f8 wb_workfn+0x80/0x2a8 process_one_work+0x178/0x3e8 worker_thread+0x234/0x3b8 kthread+0xf0/0x108 ret_from_fork+0x10/0x20 So explicitly call cond_resched() in jbd2_log_do_checkpoint() to avoid softlockup.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-37802	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ksmbd: fix WARNING "do not call blocking ops when !TASK_RUNNING" wait_event_timeout() will set the state of the current task to TASK_UNINTERRUPTIBLE, before doing the condition check. This means that ksmbd_durable_scavenger_alive() will try to acquire the mutex while already in a sleeping state. The scheduler warns us by giving the following warning: do not call blocking ops when !TASK_RUNNING; state=2 set at [<0000000061515a6f>] prepare_to_wait_event+0x9f/0x6c0 WARNING: CPU: 2 PID: 4147 at kernel/sched/core.c:10099 __might_sleep+0x12f/0x160 mutex lock is not needed in ksmbd_durable_scavenger_alive().
CVE-2024-35808	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: md/dm-raid: don't call md_reap_sync_thread() directly Currently md_reap_sync_thread() is called from raid_message() directly without holding 'reconfig_mutex', this is definitely unsafe because md_reap_sync_thread() can change many fields that is protected by 'reconfig_mutex'. However, hold 'reconfig_mutex' here is still problematic because this will cause deadlock, for example, commit 130443d60b1b ("md: refactor idle/frozen_sync_thread() to fix deadlock"). Fix this problem by using stop_sync_thread() to unregister sync_thread, like md/raid did.
CVE-2024-57977	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: memcg: fix soft lockup in the OOM process A soft lockup issue was found in the product with about 56,000 tasks were in the OOM cgroup, it was traversing them when the soft lockup was triggered. watchdog: BUG: soft lockup - CPU#2 stuck for 23s! [VM Thread:1503066] CPU: 2 PID: 1503066 Comm: VM Thread Kdump: loaded Tainted: G Hardware name: Huawei Cloud OpenStack Nova, BIOS RIP: 0010:console_unlock+0x343/0x540 RSP: 0000:ffffb751447db9a0 EFLAGS: 00000247 ORIG_RAX: ffffffffffffff13 RAX: 0000000000000001 RBX: 0000000000000000 RCX: 00000000ffffffff RDX: 0000000000000000 RSI: 0000000000000004 RDI: 0000000000000247 RBP: ffffffffafc71f90 R08: 0000000000000000 R09: 0000000000000040 R10: 0000000000000080 R11: 0000000000000000 R12: ffffffffafc74bd0 R13: ffffffffaf60a220 R14: 0000000000000247 R15: 0000000000000000 CS: 0010 DS: 0000 ES: 0000 CR0: 0000000080050033 CR2: 00007f2fe6ad91f0 CR3: 00000004b2076003 CR4: 0000000000360ee0 DR0: 0000000000000000 DR1: 0000000000000000 DR2: 0000000000000000 DR3: 0000000000000000 DR6: 00000000fffe0ff0 DR7: 0000000000000400 Call Trace: vprintk_emit+0x193/0x280 printk+0x52/0x6e dump_task+0x114/0x130 mem_cgroup_scan_tasks+0x76/0x100 dump_header+0x1fe/0x210 oom_kill_process+0xd1/0x100 out_of_memory+0x125/0x570 mem_cgroup_out_of_memory+0xb5/0xd0 try_charge+0x720/0x770 mem_cgroup_try_charge+0x86/0x180 mem_cgroup_try_charge_delay+0x1c/0x40 do_anonymous_page+0xb5/0x390 handle_mm_fault+0xc4/0x1f0 This is because thousands of processes are in the OOM cgroup, it takes a long time to traverse all of them. As a result, this lead to soft lockup in the OOM process. To fix this issue, call 'cond_resched' in the 'mem_cgroup_scan_tasks' function per 1000 iterations. For global OOM, call 'touch_softlockup_watchdog' per 1000 iterations to avoid this issue.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-39736	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: mm/kmemleak: avoid deadlock by moving pr_warn() outside kmemleak_lock When netpoll is enabled, calling pr_warn_once() while holding kmemleak_lock in mem_pool_alloc() can cause a deadlock due to lock inversion with the netconsole subsystem. This occurs because pr_warn_once() may trigger netpoll, which eventually leads to __alloc_skb() and back into kmemleak code, attempting to reacquire kmemleak_lock. This is the path for the deadlock. mem_pool_alloc() -> raw_spin_lock_irqsave(&kmemleak_lock, flags); -> pr_warn_once() -> netconsole subsystem -> netpoll -> __alloc_skb -> __create_object -> raw_spin_lock_irqsave(&kmemleak_lock, flags); Fix this by setting a flag and issuing the pr_warn_once() after kmemleak_lock is released.
CVE-2025-39910	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: mm/vmalloc, mm/kasan: respect gfp mask in kasan_populate_vmalloc() kasan_populate_vmalloc() and its helpers ignore the caller's gfp_mask and always allocate memory using the hardcoded GFP_KERNEL flag. This makes them inconsistent with vmalloc(), which was recently extended to support GFP_NOFS and GFP_NOIO allocations. Page table allocations performed during shadow population also ignore the external gfp_mask. To preserve the intended semantics of GFP_NOFS and GFP_NOIO, wrap the apply_to_page_range() calls into the appropriate memalloc scope. xfs calls vmalloc with GFP_NOFS, so this bug could lead to deadlock. There was a report here https://lkml.kernel.org/r/686ea951.050a0220.385921.0016.GAE@google.com This patch: - Extends kasan_populate_vmalloc() and helpers to take gfp_mask; - Passes gfp_mask down to alloc_pages_bulk() and __get_free_page(); - Enforces GFP_NOFS/NOIO semantics with memalloc_*_save()/restore() around apply_to_page_range(); - Updates vmalloc.c and percpu allocator call sites accordingly.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-39843	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: mm: slub: avoid wake up kswapd in set_track_prepare set_track_prepare() can incur lock recursion. The issue is that it is called from hrtimer_start_range_ns holding the per_cpu(hrtimer_bases)[n].lock, but when enabled CONFIG_DEBUG_OBJECTS_TIMERS, may wake up kswapd in set_track_prepare, and try to hold the per_cpu(hrtimer_bases)[n].lock. Avoid deadlock caused by implicitly waking up kswapd by passing in allocation flags, which do not contain __GFP_KSWAPD_RECLAIMin the debug_objects_fill_pool() case. Inside stack depot they are processed by gfp_nested_mask(). Since ___slab_alloc() has preemption disabled, we mask out __GFP_DIRECT_RECLAIMfrom the flags there. The oops looks something like: BUG: spinlock recursion on CPU#3, swapper/3/0 lock: 0xffffff8a4bf29c80, .magic: dead4ead, .owner: swapper/3/0, .owner_cpu: 3 Hardware name: Qualcomm Technologies, Inc. Popsicle based on SM8850 (DT) Call trace: spin_bug+0x0 _raw_spin_lock_irqsave+0x80 hrtimer_try_to_cancel+0x94 task_contending+0x10c enqueue_dl_entity+0x2a4 dl_server_start+0x74 enqueue_task_fair+0x568 enqueue_task+0xac do_activate_task+0x14c ttwu_do_activate+0xcc try_to_wake_up+0x6c8 default_wake_function+0x20 autoremove_wake_function+0x1c __wake_up+0xac wakeup_kswapd+0x19c wake_all_kswapds+0x78 __alloc_pages_slowpath+0x1ac __alloc_pages_noprof+0x298 stack_depot_save_flags+0x6b0 stack_depot_save+0x14 set_track_prepare+0x5c ___slab_alloc+0xccc __kmalloc_cache_noprof+0x470 __set_page_owner+0x2bc post_alloc_hook[jt]+0x1b8 prep_new_page+0x28 get_page_from_freelist+0x1edc __alloc_pages_noprof+0x13c alloc_slab_page+0x244 allocate_slab+0x7c ___slab_alloc+0x8e8 kmem_cache_alloc_noprof+0x450 debug_objects_fill_pool+0x22c debug_object_activate+0x40 enqueue_hrtimer[jt]+0xdc hrtimer_start_range_ns+0x5f8 ...
CVE-2025-38491	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: mptcp: make fallback action and fallback decision atomic Syzkaller reported the following splat: WARNING: CPU: 1 PID: 7704 at net/mptcp/protocol.h:1223 __mptcp_do_fallback net/mptcp/protocol.h:1223 [inline] WARNING: CPU: 1 PID: 7704 at net/mptcp/protocol.h:1223 mptcp_do_fallback net/mptcp/protocol.h:1244 [inline] WARNING: CPU: 1 PID: 7704 at net/mptcp/protocol.h:1223 check_fully_established net/mptcp/options.c:982 [inline] WARNING: CPU: 1 PID: 7704 at net/mptcp/protocol.h:1223 mptcp_incoming_options+0x21a8/0x2510 net/mptcp/options.c:1153 Modules linked in: CPU: 1 UID: 0 PID: 7704 Comm: syz.3.1419 Not tainted 6.16.0-rc3-gbd5ce2324dba #20 PREEMPT(voluntary) Hardware name: QEMU Ubuntu 24.04 PC (i440FX + PIIX, 1996), BIOS 1.16.3-debian-1.16.3-2 04/01/2014 RIP: 0010:__mptcp_do_fallback net/mptcp/protocol.h:1223 [inline] RIP: 0010:mptcp_do_fallback net/mptcp/protocol.h:1244 [inline] RIP: 0010:check_fully_established net/mptcp/options.c:982 [inline] RIP: 0010:mptcp_incoming_options+0x21a8/0x2510 net/mptcp/options.c:1153 Code: 24 18 e8 bb 2a 00 fd e9 1b df ff ff e8 b1 21 0f 00 e8 ec 5f c4 fc 44 0f b7 ac 24 b0 00 00 00 e9 54 f1 ff ff e8 d9 5f c4 fc 90 <0f> 0b 90 e9 b8 f4 ff ff e8 8b 2a 00 fd e9 8d e6 ff ff e8 81 2a 00 RSP: 0018:ffff8880a3f08448 EFLAGS: 00010246 RAX: 0000000000000000 RBX: ffff8880180a8000 RCX: ffffffff84afcf45 RDX:

CVE	Severity	CVSS	CVSS Vector	Component	Description
					ffff888090223700 RSI: ffffffff84afdaa7 RDI: 0000000000000001 RBP: ffff8888017955780 R08: 0000000000000001 R09: 0000000000000000 R10: 0000000000000000 R11: 0000000000000000 R12: 0000000000000000 R13: ffff8880180a8910 R14: ffff8880a3e9d058 R15: 0000000000000000 FS: 00005555791b8500(0000) GS:ffff88811c495000(0000) knlGS:0000000000000000 CS: 0010 DS: 0000 ES: 0000 CR0: 0000000080050033 CR2: 000000110c2800b7 CR3: 0000000058e44000 CR4: 0000000000350ef0 Call Trace: tcp_reset+0x26f/0x2b0 net/ipv4/tcp_input.c:4432 tcp_validate_incoming+0x1057/0x1b60 net/ipv4/tcp_input.c:5975 tcp_rcv_established+0x5b5/0x21f0 net/ipv4/tcp_input.c:6166 tcp_v4_do_rcv+0x5dc/0xa70 net/ipv4/tcp_ipv4.c:1925 tcp_v4_rcv+0x3473/0x44a0 net/ipv4/tcp_ipv4.c:2363 ip_protocol_deliver_rcu+0xba/0x480 net/ipv4/ip_input.c:205 ip_local_deliver_finish+0x2f1/0x500 net/ipv4/ip_input.c:233 NF_HOOK include/linux/netfilter.h:317 [inline] NF_HOOK include/linux/netfilter.h:311 [inline] ip_local_deliver+0x1be/0x560 net/ipv4/ip_input.c:254 dst_input include/net/dst.h:469 [inline] ip_rcv_finish net/ipv4/ip_input.c:447 [inline] NF_HOOK include/linux/netfilter.h:317 [inline] NF_HOOK include/linux/netfilter.h:311 [inline] ip_rcv+0x514/0x810 net/ipv4/ip_input.c:567 __netif_receive_skb_one_core+0x197/0x1e0 net/core/dev.c:5975 __netif_receive_skb+0x1f/0x120 net/core/dev.c:6088 process_backlog+0x301/0x1360 net/core/dev.c:6440 __napi_poll.constprop.0+0xba/0x550 net/core/dev.c:7453 napi_poll net/core/dev.c:7517 [inline] net_rx_action+0xb44/0x1010 net/core/dev.c:7644 handle_softirqs+0x1d0/0x770 kernel/softirq.c:579 do_softirq+0x3f/0x90 kernel/softirq.c:480 __local_bh_enable_ip+0xed/0x110 kernel/softirq.c:407 local_bh_enable include/linux/bottom_half.h:33 [inline] inet_csk_listen_stop+0x2c5/0x1070 net/ipv4/inet_connection_sock.c:1524 mptcp_check_listen_stop.part.0+0x1cc/0x220 net/mptcp/protocol.c:2985 mptcp_check_listen_stop net/mptcp/mib.h:118 [inline] __mptcp_close+0x9b9/0xbd0 net/mptcp/protocol.c:3000 mptcp_close+0x2f/0x140 net/mptcp/protocol.c:3066 inet_release+0xed/0x200 net/ipv4/af_inet.c:435 inet6_release+0x4f/0x70 net/ipv6/af_inet6.c:487 __sock_release+0xb3/0x270 net/socket.c:649 sock_close+0x1c/0x30 net/socket.c:1439 __fput+0x402/0xb70 fs/file_table.c:465 task_work_run+0x150/0x240 kernel/task_work.c:227 resume_user_mode_work include/linux/resume_user_mode.h:50 [inline] exit_to_user_mode_loop+0xd4 ---truncated---

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-39832	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net/mlx5: Fix lockdep assertion on sync reset unload event Fix lockdep assertion triggered during sync reset unload event. When the sync reset flow is initiated using the devlink reload fw_activate option, the PF already holds the devlink lock while handling unload event. In this case, delegate sync reset unload event handling back to the devlink callback process to avoid double-locking and resolve the lockdep warning. Kernel log: WARNING: CPU: 9 PID: 1578 at devl_assert_locked+0x31/0x40 [...] Call Trace: mlx5_unload_one_devl_locked+0x2c/0xc0 [mlx5_core] mlx5_sync_reset_unload_event+0xaf/0x2f0 [mlx5_core] process_one_work+0x222/0x640 worker_thread+0x199/0x350 kthread+0x10b/0x230 ? __pfx_worker_thread+0x10/0x10 ? __pfx_kthread+0x10/0x10 ret_from_fork+0x8e/0x100 ? __pfx_kthread+0x10/0x10 ret_from_fork_asm+0x1a/0x30
CVE-2024-27010	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net/sched: Fix mirred deadlock on device recursion When the mirred action is used on a classful egress qdisc and a packet is mirrored or redirected to self we hit a qdisc lock deadlock. See trace below. [..... other info removed for brevity....] [82.890906] [82.890906] == [82.890906] WARNING: possible recursive locking detected [82.890906] 6.8.0-05205-g77fadd89fe2d-dirty #213 Tainted: G W [82.890906] -- [82.890906] ping/418 is trying to acquire lock: [82.890906] ffff888006994110 (&sch->q.lock){+.-.}-{3:3}, at: __dev_queue_xmit+0x1778/0x3550 [82.890906] [82.890906] but task is already holding lock: [82.890906] ffff888006994110 (&sch->q.lock){+.-.}-{3:3}, at: __dev_queue_xmit+0x1778/0x3550 [82.890906] [82.890906] other info that might help us debug this: [82.890906] Possible unsafe locking scenario: [82.890906] [82.890906] CPU0 [82.890906] ---- [82.890906] lock(&sch->q.lock); [82.890906] lock(&sch->q.lock); [82.890906] [82.890906] *** DEADLOCK *** [82.890906] [..... other info removed for brevity....] Example setup (eth0->eth0) to recreate tc qdisc add dev eth0 root handle 1: htb default 30 tc filter add dev eth0 handle 1: protocol ip prio 2 matchall \ action mirred egress redirect dev eth0 Another example(eth0->eth1->eth0) to recreate tc qdisc add dev eth0 root handle 1: htb default 30 tc filter add dev eth0 handle 1: protocol ip prio 2 matchall \ action mirred egress redirect dev eth1 tc qdisc add dev eth1 root handle 1: htb default 30 tc filter add dev eth1 handle 1: protocol ip prio 2 matchall \ action mirred egress redirect dev eth0 We fix this by adding an owner field (CPU id) to struct Qdisc set after root qdisc is entered. When the softirq enters it a second time, if the qdisc owner is the same CPU, the packet is dropped to break the loop.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38553	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net/sched: Restrict conditions for adding duplicating netems to qdisc tree netem_enqueue's duplication prevention logic breaks when a netem resides in a qdisc tree with other netems - this can lead to a soft lockup and OOM loop in netem_dequeue, as seen in [1]. Ensure that a duplicating netem cannot exist in a tree with other netems. Previous approaches suggested in discussions in chronological order: 1) Track duplication status or ttl in the sk_buff struct. Considered too specific a use case to extend such a struct, though this would be a resilient fix and address other previous and potential future DOS bugs like the one described in loopy fun [2]. 2) Restrict netem_enqueue recursion depth like in act_mirred with a per cpu variable. However, netem_dequeue can call enqueue on its child, and the depth restriction could be bypassed if the child is a netem. 3) Use the same approach as in 2, but add metadata in netem_skb_cb to handle the netem_dequeue case and track a packet's involvement in duplication. This is an overly complex approach, and Jamal notes that the skb cb can be overwritten to circumvent this safeguard. 4) Prevent the addition of a netem to a qdisc tree if its ancestral path contains a netem. However, filters and actions can cause a packet to change paths when re-enqueued to the root from netem duplication, leading us to the current solution: prevent a duplicating netem from inhabiting the same tree as other netems. [1] https://lore.kernel.org/netdev/8DuRWwfqjoRDLDmBMIfbrsZg9Gx50DHJc1ilxsEBNe2D6NMbigR_eIRlG0LOjMc3r10nUUZtArXx4oZBldUfZQrwjcQhdinnMis_0G7VEk=@willsroot.io/ [2] https://lwn.net/Articles/719297/
CVE-2025-39773	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net: bridge: fix soft lockup in br_multicast_query_expired() When set multicast_query_interval to a large value, the local variable 'time' in br_multicast_send_query() may overflow. If the time is smaller than jiffies, the timer will expire immediately, and then call mod_timer() again, which creates a loop and may trigger the following soft lockup issue. watchdog: BUG: soft lockup - CPU#1 stuck for 221s! [rb_consumer:66] CPU: 1 UID: 0 PID: 66 Comm: rb_consumer Not tainted 6.16.0+ #259 PREEMPT(none) Call Trace: __netdev_alloc_skb+0x2e/0x3a0 br_ip6_multicast_alloc_query+0x212/0x1b70 __br_multicast_send_query+0x376/0xac0 br_multicast_send_query+0x299/0x510 br_multicast_query_expired.constprop.0+0x16d/0x1b0 call_timer_fn+0x3b/0x2a0 __run_timers+0x619/0x950 run_timer_softirq+0x11c/0x220 handle_softirqs+0x18e/0x560 __irq_exit_rcu+0x158/0x1a0 sysvec_apic_timer_interrupt+0x76/0x90 This issue can be reproduced with: ip link add br0 type bridge echo 1 > /sys/class/net/br0/bridge/multicast_querier echo 0xffffffffffffffff > /sys/class/net/br0/bridge/multicast_query_interval ip link set dev br0 up The multicast_startup_query_interval can also cause this issue. Similar to the commit 99b40610956a ("net: bridge: mcast: add and enforce query interval minimum"), add check for the query interval maximum to fix this issue.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-71079	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net: nfc: fix deadlock between nfc_unregister_device and rfkill_fop_write A deadlock can occur between nfc_unregister_device() and rfkill_fop_write() due to lock ordering inversion between device_lock and rfkill_global_mutex. The problematic lock order is: Thread A (rfkill_fop_write): rfkill_fop_write() mutex_lock(&rfkill_global_mutex) rfkill_set_block() nfc_rfkill_set_block() nfc_dev_down() device_lock(&dev->dev) <- waits for device_lock Thread B (nfc_unregister_device): nfc_unregister_device() device_lock(&dev->dev) rfkill_unregister() mutex_lock(&rfkill_global_mutex) <- waits for rfkill_global_mutex This creates a classic ABBA deadlock scenario. Fix this by moving rfkill_unregister() and rfkill_destroy() outside the device_lock critical section. Store the rfkill pointer in a local variable before releasing the lock, then call rfkill_unregister() after releasing device_lock. This change is safe because rfkill_fop_write() holds rfkill_global_mutex while calling the rfkill callbacks, and rfkill_unregister() also acquires rfkill_global_mutex before cleanup. Therefore, rfkill_unregister() will wait for any ongoing callback to complete before proceeding, and device_del() is only called after rfkill_unregister() returns, preventing any use-after-free. The similar lock ordering in nfc_register_device() (device_lock -> rfkill_global_mutex via rfkill_register) is safe because during registration the device is not yet in rfkill_list, so no concurrent rfkill operations can occur on this device.
CVE-2025-39885	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ocfs2: fix recursive semaphore deadlock in fiemap call syzbot detected a OCFS2 hang due to a recursive semaphore on a FS_IOC_FIEMAP of the extent list on a specially crafted mmap file. context_switch kernel/sched/core.c:5357 [inline] __schedule+0x1798/0x4cc0 kernel/sched/core.c:6961 __schedule_loop kernel/sched/core.c:7043 [inline] schedule+0x165/0x360 kernel/sched/core.c:7058 schedule_preempt_disabled+0x13/0x30 kernel/sched/core.c:7115 rwsem_down_write_slowpath+0x872/0xfe0 kernel/locking/rwsem.c:1185 __down_write_common kernel/locking/rwsem.c:1317 [inline] __down_write kernel/locking/rwsem.c:1326 [inline] down_write+0x1ab/0x1f0 kernel/locking/rwsem.c:1591 ocfs2_page_mkwrite+0x2ff/0xc40 fs/ocfs2/mmap.c:142 do_page_mkwrite+0x14d/0x310 mm/memory.c:3361 wp_page_shared mm/memory.c:3762 [inline] do_wp_page+0x268d/0x5800 mm/memory.c:3981 handle_pte_fault mm/memory.c:6068 [inline] __handle_mm_fault+0x1033/0x5440 mm/memory.c:6195 handle_mm_fault+0x40a/0x8e0 mm/memory.c:6364 do_user_addr_fault+0x764/0x1390 arch/x86/mm/fault.c:1387 handle_page_fault arch/x86/mm/fault.c:1476 [inline] exc_page_fault+0x76/0xf0 arch/x86/mm/fault.c:1532 asm_exc_page_fault+0x26/0x30 arch/x86/include/asm/idtentry.h:623 RIP: 0010:copy_user_generic arch/x86/include/asm/uaccess_64.h:126 [inline] RIP: 0010:raw_copy_to_user arch/x86/include/asm/uaccess_64.h:147 [inline] RIP: 0010:_inline_copy_to_user include/linux/uaccess.h:197 [inline] RIP: 0010:_copy_to_user+0x85/0xb0 lib/usercopy.c:26 Code: e8 00 bc f7 fc 4d 39 fc 72 3d 4d 39 ec 77 38 e8 91 b9 f7 fc 4c 89 f7 89 de e8 47 25 5b fd 0f 01 cb 4c 89 ff 48 89 d9 4c 89 f6 a4 0f 1f 00 48 89

CVE	Severity	CVSS	CVSS Vector	Component	Description
					cb 0f 01 ca 48 89 d8 5b 41 5c 41 5d 41 5e 41 RSP: 0018:ffffc9000403f950 EFLAGS: 00050256 RAX: ffffffff84c7f101 RBX: 0000000000000038 RCX: 0000000000000038 RDX: 0000000000000000 RSI: ffffc9000403f9e0 RDI: 0000200000000060 RBP: ffffc9000403fa90 R08: ffffc9000403fa17 R09: 1ffff92000807f42 R10: dfffffc0000000000 R11: fffff52000807f43 R12: 0000200000000098 R13: 00007fffffffff000 R14: ffffc9000403f9e0 R15: 0000200000000060 copy_to_user include/linux/uaccess.h:225 [inline] fiemap_fill_next_extent+0x1c0/0x390 fs/ioctl.c:145 ocfs2_fiemap+0x888/0xc90 fs/ocfs2/extent_map.c:806 ioctl_fiemap fs/ioctl.c:220 [inline] do_vfs_ioctl+0x1173/0x1430 fs/ioctl.c:532 __do_sys_ioctl fs/ioctl.c:596 [inline] __se_sys_ioctl+0x82/0x170 fs/ioctl.c:584 do_syscall_x64 arch/x86/entry/syscall_64.c:63 [inline] do_syscall_64+0xfa/0x3b0 arch/x86/entry/syscall_64.c:94 entry_SYSCALL_64_after_hwframe+0x77/0x7f RIP: 0033:0x7f5f13850fd9 RSP: 002b:00007ffe3b3518b8 EFLAGS: 00000246 ORIG_RAX: 0000000000000010 RAX: ffffffffffffffda RBX: 0000200000000000 RCX: 00007f5f13850fd9 RDX: 0000200000000040 RSI: 00000000c020660b RDI: 0000000000000004 RBP: 6165627472616568 R08: 0000000000000000 R09: 0000000000000000 R10: 0000000000000000 R11: 0000000000000246 R12: 00007ffe3b3518f0 R13: 00007ffe3b351b18 R14: 431bde82d7b634db R15: 00007f5f1389a03b ocfs2_fiemap() takes a read lock of the ip_alloc_sem semaphore (since v2.6.22-527-g7307de80510a) and calls fiemap_fill_next_extent() to read the extent list of this running mmap executable. The user supplied buffer to hold the fiemap information page faults calling ocfs2_page_mkwrite() which will take a write lock (since v2.6.27-38-g00dc417fa3e7) of the same semaphore. This recursive semaphore will hold filesystem locks and causes a hang of the fileystem. The ip_alloc_sem protects the inode extent list and size. Release the read semphore before calling fiemap_fill_next_extent() in ocfs2_fiemap() and ocfs2_fiemap_inline(). This does an unnecessary semaphore lock/unlock on the last extent but simplifies the error path.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2024-57994	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ptr_ring: do not block hard interrupts in ptr_ring_resize_multiple() Jakub added a lockdep_assert_no_hardirq() check in __page_pool_put_page() to increase test coverage. syzbot found a splat caused by hard irq blocking in ptr_ring_resize_multiple() [1] As current users of ptr_ring_resize_multiple() do not require hard irqs being masked, replace it to only block BH. Rename helpers to better reflect they are safe against BH only. - ptr_ring_resize_multiple() to ptr_ring_resize_multiple_bh() - skb_array_resize_multiple() to skb_array_resize_multiple_bh() [1] WARNING: CPU: 1 PID: 9150 at net/core/page_pool.c:709 __page_pool_put_page net/core/page_pool.c:709 [inline] WARNING: CPU: 1 PID: 9150 at net/core/page_pool.c:709 page_pool_put_unrefed_netmem+0x157/0xa40 net/core/page_pool.c:780 Modules linked in: CPU: 1 UID: 0 PID: 9150 Comm: syz.1.1052 Not tainted 6.11.0-rc3-syzkaller-00202-gf8669d7b5f5d #0 Hardware name: Google Google Compute Engine/Google Compute Engine, BIOS Google 08/06/2024 RIP: 0010:__page_pool_put_page net/core/page_pool.c:709 [inline] RIP: 0010:page_pool_put_unrefed_netmem+0x157/0xa40 net/core/page_pool.c:780 Code: 74 0e e8 7c aa fb f7 eb 43 e8 75 aa fb f7 eb 3c 65 8b 1d 38 a8 6a 76 31 ff 89 de e8 a3 ae fb f7 85 db 74 0b e8 5a aa fb f7 90 <0f> 0b 90 eb 1d 65 8b 1d 15 a8 6a 76 31 ff 89 de e8 84 ae fb f7 85 RSP: 0018:ffffc9000bda6b58 EFLAGS: 00010083 RAX: ffffffff8997e523 RBX: 0000000000000000 RCX: 0000000000040000 RDX: ffffc9000fbd0000 RSI: 0000000000001842 RDI: 0000000000001843 RBP: 0000000000000000 R08: ffffffff8997df2c R09: 1ffffd40003a000d R10: dffffc0000000000 R11: fffff940003a000e R12: ffffea0001d00040 R13: ffff88802e8a4000 R14: dffffc0000000000 R15: 00000000ffffffff FS: 00007fb7aaf716c0(0000) GS:ffff8880b9300000(0000) knlGS:0000000000000000 CS: 0010 DS: 0000 ES: 0000 CR0: 0000000080050033 CR2: 00007fa15a0d4b72 CR3: 00000000561b0000 CR4: 00000000003506f0 DR0: 0000000000000000 DR1: 0000000000000000 DR2: 0000000000000000 DR3: 0000000000000000 DR6: 00000000fffe0ff0 DR7: 0000000000000400 Call Trace: tun_ptr_free drivers/net/tun.c:617 [inline] __ptr_ring_swap_queue include/linux/ptr_ring.h:571 [inline] ptr_ring_resize_multiple_noprof include/linux/ptr_ring.h:643 [inline] tun_queue_resize drivers/net/tun.c:3694 [inline] tun_device_event+0xaaf/0x1080 drivers/net/tun.c:3714 notifier_call_chain+0x19f/0x3e0 kernel/notifier.c:93 call_netdevice_notifiers_extack net/core/dev.c:2032 [inline] call_netdevice_notifiers net/core/dev.c:2046 [inline] dev_change_tx_queue_len+0x158/0x2a0 net/core/dev.c:9024 do_setlink+0xff6/0x41f0 net/core/rtnetlink.c:2923 rtnl_setlink+0x40d/0x5a0 net/core/rtnetlink.c:3201 rtnetlink_rcv_msg+0x73f/0xcf0 net/core/rtnetlink.c:6647 netlink_rcv_skb+0x1e3/0x430 net/netlink/af_netlink.c:2550

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38359	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: s390/mm: Fix in_atomic() handling in do_secure_storage_access() Kernel user spaces accesses to not exported pages in atomic context incorrectly try to resolve the page fault. With debug options enabled call traces like this can be seen: BUG: sleeping function called from invalid context at kernel/locking/rwsem.c:1523 in_atomic(): 1, irqs_disabled(): 0, non_block: 0, pid: 419074, name: qemu-system-s39 preempt_count: 1, expected: 0 RCU nest depth: 0, expected: 0 INFO: lockdep is turned off. Preemption disabled at: [<00000383ea47cfa2>] copy_page_from_iter_atomic+0xa2/0x8a0 CPU: 12 UID: 0 PID: 419074 Comm: qemu-system-s39 Tainted: G W 6.16.0-20250531.rc0.git0.69b3a602feac.63.fc42.s390x+debug #1 PREEMPT Tainted: [W]=WARN Hardware name: IBM 3931 A01 703 (LPAR) Call Trace: [<00000383e990d282>] dump_stack_lvl+0xa2/0xe8 [<00000383e99bf152>] __might_resched+0x292/0x2d0 [<00000383eaa7c374>] down_read+0x34/0x2d0 [<00000383e99432f8>] do_secure_storage_access+0x108/0x360 [<00000383eaa724b0>] __do_pgm_check+0x130/0x220 [<00000383eaa842e4>] pgm_check_handler+0x114/0x160 [<00000383ea47d028>] copy_page_from_iter_atomic+0x128/0x8a0 ([<00000383ea47d016>] copy_page_from_iter_atomic+0x116/0x8a0) [<00000383e9c45eae>] generic_perform_write+0x16e/0x310 [<00000383e9eb87f4>] ext4_buffered_write_iter+0x84/0x160 [<00000383e9da0de4>] vfs_write+0x1c4/0x460 [<00000383e9da123c>] ksys_write+0x7c/0x100 [<00000383eaa7284e>] __do_syscall+0x15e/0x280 [<00000383eaa8417e>] system_call+0x6e/0x90 INFO: lockdep is turned off. It is not allowed to take the mmap_lock while in atomic context. Therefore handle such a secure storage access fault as if the accessed page is not mapped: the uaccess function will return -EFAULT, and the caller has to deal with this. Usually this means that the access is retried in process context, which allows to resolve the page fault (or in this case export the page).
CVE-2025-38711	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: smb/server: avoid deadlock when linking with ReplaceIfExists If smb2_create_link() is called with ReplaceIfExists set and the name does exist then a deadlock will happen. ksmbd_vfs_kern_path_locked() will return with success and the parent directory will be locked. ksmbd_vfs_remove_file() will then remove the file. ksmbd_vfs_link() will then be called while the parent is still locked. It will try to lock the same parent and will deadlock. This patch moves the ksmbd_vfs_kern_path_unlock() call to *before* ksmbd_vfs_link() and then simplifies the code, removing the file_present flag variable.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-68823	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ublk: fix deadlock when reading partition table When one process(such as udev) opens ublk block device (e.g., to read the partition table via bdev_open()), a deadlock[1] can occur: 1. bdev_open() grabs disk->open_mutex 2. The process issues read I/O to ublk backend to read partition table 3. In __ublk_complete_rq(), blk_update_request() or blk_mq_end_request() runs bio->bi_end_io() callbacks 4. If this triggers fput() on file descriptor of ublk block device, the work may be deferred to current task's task work (see fput() implementation) 5. This eventually calls blkdev_release() from the same context 6. blkdev_release() tries to grab disk->open_mutex again 7. Deadlock: same task waiting for a mutex it already holds The fix is to run blk_update_request() and blk_mq_end_request() with bottom halves disabled. This forces blkdev_release() to run in kernel work-queue context instead of current task work context, and allows ublk server to make forward progress, and avoids the deadlock. [axboe: rewrite comment in ublk]
CVE-2025-37880	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: um: work around sched_yield not yielding in time-travel mode sched_yield by a userspace may not actually cause scheduling in time-travel mode as no time has passed. In the case seen it appears to be a badly implemented userspace spinlock in ASAN. Unfortunately, with time-travel it causes an extreme slowdown or even deadlock depending on the kernel configuration (CONFIG_UML_MAX_USERSPACE_ITERATIONS). Work around it by accounting time to the process whenever it executes a sched_yield syscall.
CVE-2025-38404	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: usb: typec: displayport: Fix potential deadlock The deadlock can occur due to a recursive lock acquisition of `cros_typec_altmode_data::mutex`. The call chain is as follows: 1. cros_typec_altmode_work() acquires the mutex 2. typec_altmode_vdm() -> dp_altmode_vdm() -> 3. typec_altmode_exit() -> cros_typec_altmode_exit() 4. cros_typec_altmode_exit() attempts to acquire the mutex again To prevent this, defer the `typec_altmode_exit()` call by scheduling it rather than calling it directly from within the mutex-protected context.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38643	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: wifi: cfg80211: Add missing lock in cfg80211_check_and_end_cac() Callers of wdev_chandef() must hold the wiphy mutex. But the worker cfg80211_propagate_cac_done_wk() never takes the lock. Which triggers the warning below with the mesh_peer_connected_dfs test from hostapd and not (yet) released mac80211 code changes: WARNING: CPU: 0 PID: 495 at net/wireless/chan.c:1552 wdev_chandef+0x60/0x165 Modules linked in: CPU: 0 UID: 0 PID: 495 Comm: kworker/u4:2 Not tainted 6.14.0-rc5-wt-g03960e6f9d47 #33 13c287eeabfe1efea01c0bcc863723ab082e17cf Workqueue: cfg80211 cfg80211_propagate_cac_done_wk Stack: 00000000 00000001 ffffff00 6093267c 00000000 6002ec30 6d577c50 60037608 00000000 67e8d108 6063717b 00000000 Call Trace: [<6002ec30>] ? _printk+0x0/0x98 [<6003c2b3>] show_stack+0x10e/0x11a [<6002ec30>] ? _printk+0x0/0x98 [<60037608>] dump_stack_lvl+0x71/0xb8 [<6063717b>] ? wdev_chandef+0x60/0x165 [<6003766d>] dump_stack+0x1e/0x20 [<6005d1b7>] __warn+0x101/0x20f [<6005d3a8>] warn_slowpath_fmt+0xe3/0x15d [<600b0c5c>] ? mark_lock.part.0+0x0/0x4ec [<60751191>] ? __this_cpu_preempt_check+0x0/0x16 [<600b11a2>] ? mark_held_locks+0x5a/0x6e [<6005d2c5>] ? warn_slowpath_fmt+0x0/0x15d [<60052e53>] ? unblock_signals+0x3a/0xe7 [<60052f2d>] ? um_set_signals+0x2d/0x43 [<60751191>] ? __this_cpu_preempt_check+0x0/0x16 [<607508b2>] ? lock_is_held_type+0x207/0x21f [<6063717b>] wdev_chandef+0x60/0x165 [<605f89b4>] regulatory_propagate_dfs_state+0x247/0x43f [<60052f00>] ? um_set_signals+0x0/0x43 [<605e6bfd>] cfg80211_propagate_cac_done_wk+0x3a/0x4a [<6007e460>] process_scheduled_works+0x3bc/0x60e [<6007d0ec>] ? move_linked_works+0x4d/0x81 [<6007d120>] ? assign_work+0x0/0xaa [<6007f81f>] worker_thread+0x220/0x2dc [<600786ef>] ? set_pf_worker+0x0/0x57 [<60087c96>] ? to_kthread+0x0/0x43 [<6008ab3c>] kthread+0x2d3/0x2e2 [<6007f5ff>] ? worker_thread+0x0/0x2dc [<6006c05b>] ? calculate_sigpending+0x0/0x56 [<6003b37d>] new_thread_handler+0x4a/0x64 irq event stamp: 614611 hardirqs last enabled at (614621): [<00000000600bc96b>] __up_console_sem+0x82/0xaf hardirqs last disabled at (614630): [<00000000600bc92c>] __up_console_sem+0x43/0xaf softirqs last enabled at (614268): [<00000000606c55c6>] __ieee80211_wake_queue+0x933/0x985 softirqs last disabled at (614266): [<00000000606c52d6>] __ieee80211_wake_queue+0x643/0x985
CVE-2025-39795	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: block: avoid possible overflow for chunk_sectors check in blk_stack_limits() In blk_stack_limits(), we check that the t->chunk_sectors value is a multiple of the t->physical_block_size value. However, by finding the chunk_sectors value in bytes, we may overflow the unsigned int which holds chunk_sectors, so change the check to be based on sectors.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38614	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: eventpoll: Fix semi-unbounded recursion Ensure that epoll instances can never form a graph deeper than EP_MAX_NESTS+1 links. Currently, ep_loop_check_proc() ensures that the graph is loop-free and does some recursion depth checks, but those recursion depth checks don't limit the depth of the resulting tree for two reasons: - They don't look upwards in the tree. - If there are multiple downwards paths of different lengths, only one of the paths is actually considered for the depth check since commit 28d82dc1c4ed ("epoll: limit paths"). Essentially, the current recursion depth check in ep_loop_check_proc() just serves to prevent it from recursing too deeply while checking for loops. A more thorough check is done in reverse_path_check() after the new graph edge has already been created; this checks, among other things, that no paths going upwards from any non-epoll file with a length of more than 5 edges exist. However, this check does not apply to non-epoll files. As a result, it is possible to recurse to a depth of at least roughly 500, tested on v6.15. (I am unsure if deeper recursion is possible; and this may have changed with commit 8c44dac8add7 ("eventpoll: Fix priority inversion problem").) To fix it: 1. In ep_loop_check_proc(), note the subtree depth of each visited node, and use subtree depths for the total depth calculation even when a subtree has already been visited. 2. Add ep_get_upwards_depth_proc() for similarly determining the maximum depth of an upwards walk. 3. In ep_loop_check(), use these values to limit the total path length between epoll nodes to EP_MAX_NESTS edges.
CVE-2025-38493	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: tracing/osnoise: Fix crash in timerlat_dump_stack() We have observed kernel panics when using timerlat with stack saving, with the following dmesg output: memcpy: detected buffer overflow: 88 byte write of buffer size 0 WARNING: CPU: 2 PID: 8153 at lib/string_helpers.c:1032 __fortify_report+0x55/0xa0 CPU: 2 UID: 0 PID: 8153 Comm: timerlatu/2 Kdump: loaded Not tainted 6.15.3-200.fc42.x86_64 #1 PREEMPT(lazy) Call Trace: ? trace_buffer_lock_reserve+0x2a/0x60 __fortify_panic+0xd/0xf __timerlat_dump_stack.cold+0xd/0xd timerlat_dump_stack.part.0+0x47/0x80 timerlat_fd_read+0x36d/0x390 vfs_read+0xe2/0x390 ? syscall_exit_to_user_mode+0x1d5/0x210 ksys_read+0x73/0xe0 do_syscall_64+0x7b/0x160 ? exc_page_fault+0x7e/0x1a0 entry_SYSCALL_64_after_hwframe+0x76/0x7e __timerlat_dump_stack() constructs the ftrace stack entry like this: struct stack_entry *entry; ... memcpy(&entry->caller, fstack->calls, size); entry->size = fstack->nr_entries; Since commit e7186af7fb26 ("tracing: Add back FORTIFY_SOURCE logic to kernel_stack event structure"), struct stack_entry marks its caller field with __counted_by(size). At the time of the memcpy, entry->size contains garbage from the ringbuffer, which under some circumstances is zero, triggering a kernel panic by buffer overflow. Populate the size field before the memcpy so that the out-of-bounds check knows the correct size. This is analogous to __ftrace_trace_stack().
CVE-2024-25741	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	printer_write in drivers/usb/gadget/function/f_printer.c in the Linux kernel through 6.7.4 does not properly call usb_ep_queue, which might allow attackers to cause a denial of service or have unspecified other impact.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2024-25739	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	create_empty_lvol in drivers/mtd/ubi/vtbl.c in the Linux kernel through 6.7.4 can attempt to allocate zero bytes, and crash, because of a missing check for ubi->leb_size.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-71232	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: scsi: qla2xxx: Free sp in error path to fix system crash System crash seen during load/unload test in a loop, [61110.449331] qla2xxx [0000:27:00.0]-0042:0: Disabled MSI-X. [61110.467494] ========== = [61110.467498] BUG qla2xxx_srbs (Tainted: G OE -------- ---): Objects remaining in qla2xxx_srbs on __kmem_cache_shutdown() [61110.467501] ---------- [61110.467502] Slab 0x000000000ffc8162 objects=51 used=1 fp=0x00000000e25d3d85 flags=0x57ffffc0010200(slab|head|node=1|zone=2|lastcpupid=0x1fffff) [61110.467509] CPU: 53 PID: 455206 Comm: rmmod Kdump: loaded Tainted: G OE -------- --- 5.14.0-284.11.1.el9_2.x86_64 #1 [61110.467513] Hardware name: HPE ProLiant DL385 Gen10 Plus v2/ProLiant DL385 Gen10 Plus v2, BIOS A42 08/17/2023 [61110.467515] Call Trace: [61110.467516] [61110.467519] dump_stack_lvl+0x34/0x48 [61110.467526] slab_err.cold+0x53/0x67 [61110.467534] __kmem_cache_shutdown+0x16e/0x320 [61110.467540] kmem_cache_destroy+0x51/0x160 [61110.467544] qla2x00_module_exit+0x93/0x99 [qla2xxx] [61110.467607] ? __do_sys_delete_module.constprop.0+0x178/0x280 [61110.467613] ? syscall_trace_enter.constprop.0+0x145/0x1d0 [61110.467616] ? do_syscall_64+0x5c/0x90 [61110.467619] ? exc_page_fault+0x62/0x150 [61110.467622] ? entry_SYSCALL_64_after_hwframe+0x63/0xcd [61110.467626] [61110.467627] Disabling lock debugging due to kernel taint [61110.467635] Object 0x0000000026f7e6e6 @offset=16000 [61110.467639] ------------[cut here]------------ [61110.467639] kmem_cache_destroy qla2xxx_srbs: Slab cache still has objects when called from qla2x00_module_exit+0x93/0x99 [qla2xxx] [61110.467659] WARNING: CPU: 53 PID: 455206 at mm/slab_common.c:520 kmem_cache_destroy+0x14d/0x160 [61110.467718] CPU: 53 PID: 455206 Comm: rmmod Kdump: loaded Tainted: G B OE -------- --- 5.14.0-284.11.1.el9_2.x86_64 #1 [61110.467720] Hardware name: HPE ProLiant DL385 Gen10 Plus v2/ProLiant DL385 Gen10 Plus v2, BIOS A42 08/17/2023 [61110.467721] RIP: 0010:kmem_cache_destroy+0x14d/0x160 [61110.467724] Code: 99 7d 07 00 48 89 ef e8 e1 6a 07 00 eb b3 48 8b 55 60 48 8b 4c 24 20 48 c7 c6 70 fc 66 90 48 c7 c7 f8 ef a1 90 e8 e1 ed 7c 00 <0f> 0b eb 93 c3 cc cc cc cc 66 2e 0f 1f 84 00 00 00 00 00 55 48 89 [61110.467725] RSP: 0018:ffffa304e489fe80 EFLAGS: 00010282 [61110.467727] RAX: 0000000000000000 RBX: ffffffffc0d9a860 RCX: 0000000000000027 [61110.467729] RDX: ffff8fd5ff9598a8 RSI: 0000000000000001 RDI: ffff8fd5ff9598a0 [61110.467730] RBP: ffff8fb6aaf78700 R08: 0000000000000000 R09: 0000000100d863b7 [61110.467731] R10: ffffa304e489fd20 R11: ffffffff913bef48 R12: 0000000040002000 [61110.467731] R13: 0000000000000000 R14: 0000000000000000 R15: 0000000000000000 [61110.467733] FS: 00007f64c89fb740(0000) GS:ffff8fd5ff940000(0000) knlGS:0000000000000000 [61110.467734] CS: 0010 DS: 0000 ES: 0000 CR0: 0000000080050033 [61110.467735] CR2: 00007f0f02bfe000 CR3: 00000020ad6dc005 CR4: 0000000000770ee0 [61110.467736] PKRU: 55555554 [61110.467737] Call Trace: [61110.467738] [61110.467739] qla2x00_module_exit+0x93/0x99 [qla2xxx] [61110.467755] ? __do_sys_delete_module.constprop.0+0x178/0x280 Free sp in the error path to fix the crash.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-71109	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: MIPS: ftrace: Fix memory corruption when kernel is located beyond 32 bits Since commit e424054000878 ("MIPS: Tracing: Reduce the overhead of dynamic Function Tracer"), the macro UASM_i_LA_mostly has been used, and this macro can generate more than 2 instructions. At the same time, the code in ftrace assumes that no more than 2 instructions can be generated, which is why it stores them in an int[2] array. However, as previously noted, the macro UASM_i_LA_mostly (and now UASM_i_LA) causes a buffer overflow when _mcount is beyond 32 bits. This leads to corruption of the variables located in the __read_mostly section. This corruption was observed because the variable __cpu_primary_thread_mask was corrupted, causing a hang very early during boot. This fix prevents the corruption by avoiding the generation of instructions if they could exceed 2 instructions in length. Fortunately, insn_la_mcount is only used if the instrumented code is located outside the kernel code section, so dynamic ftrace can still be used, albeit in a more limited scope. This is still preferable to corrupting memory and/or crashing the kernel.
CVE-2024-26811	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ksmbd: validate payload size in ipc response If installing malicious ksmbd-tools, ksmbd.mountd can return invalid ipc response to ksmbd kernel server. ksmbd should validate payload size of ipc response from ksmbd.mountd to avoid memory overrun or slab-out-of-bounds. This patch validate 3 ipc response that has payload.
CVE-2025-39718	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: vsock/virtio: Validate length in packet header before skb_put() When receiving a vsock packet in the guest, only the virtqueue buffer size is validated prior to virtio_vsock_skb_rx_put(). Unfortunately, virtio_vsock_skb_rx_put() uses the length from the packet header as the length argument to skb_put(), potentially resulting in SKB overflow if the host has gone wonky. Validate the length as advertised by the packet header before calling virtio_vsock_skb_rx_put().

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2024-42071	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ionic: use dev_consume_skb_any outside of napi If we're not in a NAPI softirq context, we need to be careful about how we call napi_consume_skb(), specifically we need to call it with budget==0 to signal to it that we're not in a safe context. This was found while running some configuration stress testing of traffic and a change queue config loop running, and this curious note popped out: [4371.402645] BUG: using smp_processor_id() in preemptible [00000000] code: ethtool/20545 [4371.402897] caller is napi_skb_cache_put+0x16/0x80 [4371.403120] CPU: 25 PID: 20545 Comm: ethtool Kdump: loaded Tainted: G OE 6.10.0-rc3-netnext+ #8 [4371.403302] Hardware name: HPE ProLiant DL360 Gen10/ProLiant DL360 Gen10, BIOS U32 01/23/2021 [4371.403460] Call Trace: [4371.403613] [4371.403758] dump_stack_lvl+0x4f/0x70 [4371.403904] check_preemption_disabled+0xc1/0xe0 [4371.404051] napi_skb_cache_put+0x16/0x80 [4371.404199] ionic_tx_clean+0x18a/0x240 [ionic] [4371.404354] ionic_tx_cq_service+0xc4/0x200 [ionic] [4371.404505] ionic_tx_flush+0x15/0x70 [ionic] [4371.404653] ? ionic_lif_qcq_deinit.isra.23+0x5b/0x70 [ionic] [4371.404805] ionic_txrx_deinit+0x71/0x190 [ionic] [4371.404956] ionic_reconfigure_queues+0x5f5/0xff0 [ionic] [4371.405111] ionic_set_ringparam+0x2e8/0x3e0 [ionic] [4371.405265] ethnl_set_rings+0x1f1/0x300 [4371.405418] ethnl_default_set_doit+0xbb/0x160 [4371.405571] genl_family_rcv_msg_doit+0xff/0x130 [...] I found that ionic_tx_clean() calls napi_consume_skb() which calls napi_skb_cache_put(), but before that last call is the note /* Zero budget indicate non-NAPI context called us, like netpoll */ and DEBUG_NET_WARN_ON_ONCE(!in_softirq()); Those are pretty big hints that we're doing it wrong. We can pass a context hint down through the calls to let ionic_tx_clean() know what we're doing so it can call napi_consume_skb() correctly.
CVE-2024-36288	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: SUNRPC: Fix loop termination condition in gss_free_in_token_pages() The in_token->pages[] array is not NULL terminated. This results in the following KASAN splat: KASAN: maybe wild-memory-access in range [0x04a2013400000008-0x04a201340000000f]

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2024-47794	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: bpf: Prevent tailcall infinite loop caused by freplace There is a potential infinite loop issue that can occur when using a combination of tail calls and freplace. In an upcoming selftest, the attach target for entry_freplace of tailcall_freplace.c is subprog_tc of tc_bpf2bpf.c, while the tail call in entry_freplace leads to entry_tc. This results in an infinite loop: entry_tc -> subprog_tc -> entry_freplace --tailcall-> entry_tc. The problem arises because the tail_call_cnt in entry_freplace resets to zero each time entry_freplace is executed, causing the tail call mechanism to never terminate, eventually leading to a kernel panic. To fix this issue, the solution is twofold: 1. Prevent updating a program extended by an freplace program to a prog_array map. 2. Prevent extending a program that is already part of a prog_array map with an freplace program. This ensures that: * If a program or its subprogram has been extended by an freplace program, it can no longer be updated to a prog_array map. * If a program has been added to a prog_array map, neither it nor its subprograms can be extended by an freplace program. Moreover, an extension program should not be tailcalled. As such, return -EINVAL if the program has a type of BPF_PROG_TYPE_EXT when adding it to a prog_array map. Additionally, fix a minor code style issue by replacing eight spaces with a tab for proper formatting.
CVE-2025-38692	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: exfat: add cluster chain loop check for dir An infinite loop may occur if the following conditions occur due to file system corruption. (1) Condition for exfat_count_dir_entries() to loop infinitely. - The cluster chain includes a loop. - There is no UNUSED entry in the cluster chain. (2) Condition for exfat_create_upcase_table() to loop infinitely. - The cluster chain of the root directory includes a loop. - There are no UNUSED entry and up-case table entry in the cluster chain of the root directory. (3) Condition for exfat_load_bitmap() to loop infinitely. - The cluster chain of the root directory includes a loop. - There are no UNUSED entry and bitmap entry in the cluster chain of the root directory. (4) Condition for exfat_find_dir_entry() to loop infinitely. - The cluster chain includes a loop. - The unused directory entries were exhausted by some operation. (5) Condition for exfat_check_dir_empty() to loop infinitely. - The cluster chain includes a loop. - The unused directory entries were exhausted by some operation. - All files and sub-directories under the directory are deleted. This commit adds checks to break the above infinite loop.
CVE-2025-38587	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ipv6: fix possible infinite loop in fib6_info_uses_dev() fib6_info_uses_dev() seems to rely on RCU without an explicit protection. Like the prior fix in rt6_nlmsg_size(), we need to make sure fib6_del_route() or fib6_add_rt2node() have not removed the anchor from the list, or we risk an infinite loop.
CVE-2025-38588	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ipv6: prevent infinite loop in rt6_nlmsg_size() While testing prior patch, I was able to trigger an infinite loop in rt6_nlmsg_size() in the following place: list_for_each_entry_rcu(sibling, &f6i->fib6_siblings, fib6_siblings) { rt6_nh_nlmsg_size(sibling->fib6_nh, &nexthop_len); } This is because fib6_del_route() and fib6_add_rt2node() uses list_del_rcu(), which can confuse rcu readers, because they might no longer see the head of the list. Restart the loop if f6i->fib6_nsiblings is zero.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2026-23220	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ksmbd: fix infinite loop caused by next_smb2_rcv_hdr_off reset in error paths The problem occurs when a signed request fails smb2 signature verification check. In __process_request(), if check_sign_req() returns an error, set_smb2_rsp_status(work, STATUS_ACCESS_DENIED) is called. set_smb2_rsp_status() set work->next_smb2_rcv_hdr_off as zero. By resetting next_smb2_rcv_hdr_off to zero, the pointer to the next command in the chain is lost. Consequently, is_chained_smb2_message() continues to point to the same request header instead of advancing. If the header's NextCommand field is non-zero, the function returns true, causing __handle_ksmbd_work() to repeatedly process the same failed request in an infinite loop. This results in the kernel log being flooded with "bad smb2 signature" messages and high CPU usage. This patch fixes the issue by changing the return value from SERVER_HANDLER_CONTINUE to SERVER_HANDLER_ABORT. This ensures that the processing loop terminates immediately rather than attempting to continue from an invalidated offset.
CVE-2024-46701	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: libfs: fix infinite directory reads for offset dir After we switch tmpfs dir operations from simple_dir_operations to simple_offset_dir_operations, every rename happened will fill new dentry to dest dir's maple tree(&SHMEM_I(inode)->dir_offsets->mt) with a free key starting with octx->newx_offset, and then set newx_offset equals to free key + 1. This will lead to infinite readdir combine with rename happened at the same time, which fail generic/736 in xfstests(detail show as below). 1. create 5000 files(1 2 3...) under one dir 2. call readdir(man 3 readdir) once, and get one entry 3. rename(entry, "TEMPFILE"), then rename("TEMPFILE", entry) 4. loop 2~3, until readdir return nothing or we loop too many times(tmpfs break test with the second condition) We choose the same logic what commit 9b378f6ad48cf ("btrfs: fix infinite directory reads") to fix it, record the last_index when we open dir, and do not emit the entry which index >= last_index. The file->private_data now used in offset dir can use directly to do this, and we also update the last_index when we llseek the dir file. [brauner: only update last_index after seek when offset is zero like Jan suggested]
CVE-2024-58097	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: wifi: ath11k: fix RCU stall while reaping monitor destination ring While processing the monitor destination ring, MSDUs are reaped from the link descriptor based on the corresponding buf_id. However, sometimes the driver cannot obtain a valid buffer corresponding to the buf_id received from the hardware. This causes an infinite loop in the destination processing, resulting in a kernel crash. kernel log: ath11k_pci 0000:58:00.0: data msdu_pop: invalid buf_id 309 ath11k_pci 0000:58:00.0: data dp_rx_monitor_link_desc_return failed ath11k_pci 0000:58:00.0: data msdu_pop: invalid buf_id 309 ath11k_pci 0000:58:00.0: data dp_rx_monitor_link_desc_return failed Fix this by skipping the problematic buf_id and reaping the next entry, replacing the break with the next MSDU processing. Tested-on: WCN6855 hw2.0 PCI WLAN.HSP.1.1-03125-QCAHSPSWPL_V1_V2_SILICONZ_LITE-3.6510.30 Tested-on: QCN9074 hw1.0 PCI WLAN.HK.2.7.0.1-01744-QCAHKSWPL_SILICONZ-1

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-71096	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: RDMA/core: Check for the presence of LS_NLA_TYPE_DGID correctly The netlink response for RDMA_NL_LS_OP_IP_RESOLVE should always have a LS_NLA_TYPE_DGID attribute, it is invalid if it does not. Use the nl parsing logic properly and call nla_parse_deprecated() to fill the nlattrs array and then directly index that array to get the data for the DGID. Just fail if it is NULL. Remove the for loop searching for the nla, and squash the validation and parsing into one function. Fixes an uninitialized read from the stack triggered by userspace if it does not provide the DGID to a kernel initiated RDMA_NL_LS_OP_IP_RESOLVE query. BUG: KMSAN: uninit-value in hex_byte_pack include/linux/hex.h:13 [inline] BUG: KMSAN: uninit-value in ip6_string+0xef4/0x13a0 lib/vsprintf.c:1490 hex_byte_pack include/linux/hex.h:13 [inline] ip6_string+0xef4/0x13a0 lib/vsprintf.c:1490 ip6_addr_string+0x18a/0x3e0 lib/vsprintf.c:1509 ip_addr_string+0x245/0xee0 lib/vsprintf.c:1633 pointer+0xc09/0x1bd0 lib/vsprintf.c:2542 vsnprintf+0xf8a/0x1bd0 lib/vsprintf.c:2930 vprintk_store+0x3ae/0x1530 kernel/printk/printk.c:2279 vprintk_emit+0x307/0xcd0 kernel/printk/printk.c:2426 vprintk_default+0x3f/0x50 kernel/printk/printk.c:2465 vprintk+0x36/0x50 kernel/printk/printk_safe.c:82 _printk+0x17e/0x1b0 kernel/printk/printk.c:2475 ib_nl_process_good_ip_rsep drivers/infiniband/core/addr.c:128 [inline] ib_nl_handle_ip_res_resp+0x963/0x9d0 drivers/infiniband/core/addr.c:141 rdma_nl_rcv_msg drivers/infiniband/core/netlink.c:-1 [inline] rdma_nl_rcv_skb drivers/infiniband/core/netlink.c:239 [inline] rdma_nl_rcv+0xefa/0x11c0 drivers/infiniband/core/netlink.c:259 netlink_unicast_kernel net/netlink/af_netlink.c:1320 [inline] netlink_unicast+0xf04/0x12b0 net/netlink/af_netlink.c:1346 netlink_sendmsg+0x10b3/0x1250 net/netlink/af_netlink.c:1896 sock_sendmsg_nosec net/socket.c:714 [inline] __sock_sendmsg+0x333/0x3d0 net/socket.c:729 ____sys_sendmsg+0x7e0/0xd80 net/socket.c:2617 ___sys_sendmsg+0x271/0x3b0 net/socket.c:2671 __sys_sendmsg+0x1aa/0x300 net/socket.c:2703 __compat_sys_sendmsg net/compat.c:346 [inline] __do_compat_sys_sendmsg net/compat.c:353 [inline] __se_compat_sys_sendmsg net/compat.c:350 [inline] __ia32_compat_sys_sendmsg+0xa4/0x100 net/compat.c:350 ia32_sys_call+0x3f6c/0x4310 arch/x86/include/generated/asm/syscalls_32.h:371 do_syscall_32_irqs_on arch/x86/entry/syscall_32.c:83 [inline] __do_fast_syscall_32+0xb0/0x150 arch/x86/entry/syscall_32.c:306 do_fast_syscall_32+0x38/0x80 arch/x86/entry/syscall_32.c:331 do_SYSENTER_32+0x1f/0x30 arch/x86/entry/syscall_32.c:3

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38608	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: bpf, ktls: Fix data corruption when using bpf_msg_pop_data() in ktls When sending plaintext data, we initially calculated the corresponding ciphertext length. However, if we later reduced the plaintext data length via socket policy, we failed to recalculate the ciphertext length. This results in transmitting buffers containing uninitialized data during ciphertext transmission. This causes uninitialized bytes to be appended after a complete "Application Data" packet, leading to errors on the receiving end when parsing TLS record.
CVE-2025-38382	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: btrfs: fix iteration of extrefs during log replay At __inode_add_ref() when processing extrefs, if we jump into the next label we have an undefined value of victim_name.len, since we haven't initialized it before we did the goto. This results in an invalid memory access in the next iteration of the loop since victim_name.len was not initialized to the length of the name of the current extref. Fix this by initializing victim_name.len with the current extref's name length.
CVE-2025-38478	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: comedi: Fix initialization of data for instructions that write to subdevice Some Comedi subdevice instruction handlers are known to access instruction data elements beyond the first `insn->n` elements in some cases. The `do_insn_ioctl()` and `do_insnlist_ioctl()` functions allocate at least `MIN_SAMPLES` (16) data elements to deal with this, but they do not initialize all of that. For Comedi instruction codes that write to the subdevice, the first `insn->n` data elements are copied from user-space, but the remaining elements are left uninitialized. That could be a problem if the subdevice instruction handler reads the uninitialized data. Ensure that the first `MIN_SAMPLES` elements are initialized before calling these instruction handlers, filling the uncopied elements with 0. For `do_insnlist_ioctl()`, the same data buffer elements are used for handling a list of instructions, so ensure the first `MIN_SAMPLES` elements are initialized for each instruction that writes to the subdevice.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38480	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: comedi: Fix use of uninitialized data in insn_rw_emulate_bits() For Comedi `INSN_READ` and `INSN_WRITE` instructions on "digital" subdevices (subdevice types `COMEDI_SUBD_DI`, `COMEDI_SUBD_DO`, and `COMEDI_SUBD_DIO`), it is common for the subdevice driver not to have `insn_read` and `insn_write` handler functions, but to have an `insn_bits` handler function for handling Comedi `INSN_BITS` instructions. In that case, the subdevice's `insn_read` and/or `insn_write` function handler pointers are set to point to the `insn_rw_emulate_bits()` function by `__comedi_device_postconfig()`. For `INSN_WRITE`, `insn_rw_emulate_bits()` currently assumes that the supplied `data[0]` value is a valid copy from user memory. It will at least exist because `do_insnlist_ioctl()` and `do_insn_ioctl()` in "comedi_fops.c" ensure at lease `MIN_SAMPLES` (16) elements are allocated. However, if `insn->n` is 0 (which is allowable for `INSN_READ` and `INSN_WRITE` instructions, then `data[0]` may contain uninitialized data, and certainly contains invalid data, possibly from a different instruction in the array of instructions handled by `do_insnlist_ioctl()`. This will result in an incorrect value being written to the digital output channel (or to the digital input/output channel if configured as an output), and may be reflected in the internal saved state of the channel. Fix it by returning 0 early if `insn->n` is 0, before reaching the code that accesses `data[0]`. Previously, the function always returned 1 on success, but it is supposed to be the number of data samples actually read or written up to `insn->n`, which is 0 in this case.
CVE-2025-39684	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: comedi: Fix use of uninitialized memory in do_insn_ioctl() and do_insnlist_ioctl() syzbot reports a KMSAN kernel-infoleak in `do_insn_ioctl()`. A kernel buffer is allocated to hold `insn->n` samples (each of which is an `unsigned int`). For some instruction types, `insn->n` samples are copied back to user-space, unless an error code is being returned. The problem is that not all the instruction handlers that need to return data to userspace fill in the whole `insn->n` samples, so that there is an information leak. There is a similar syzbot report for `do_insnlist_ioctl()`, although it does not have a reproducer for it at the time of writing. One culprit is `insn_rw_emulate_bits()` which is used as the handler for `INSN_READ` or `INSN_WRITE` instructions for subdevices that do not have a specific handler for that instruction, but do have an `INSN_BITS` handler. For `INSN_READ` it only fills in at most 1 sample, so if `insn->n` is greater than 1, the remaining `insn->n - 1` samples copied to userspace will be uninitialized kernel data. Another culprit is `vm80xx_ai_insn_read()` in the "vm80xx" driver. It never returns an error, even if it fails to fill the buffer. Fix it in `do_insn_ioctl()` and `do_insnlist_ioctl()` by making sure that uninitialized parts of the allocated buffer are zeroed before handling each instruction. Thanks to Arnaud Lecomte for their fix to `do_insn_ioctl()`. That fix replaced the call to `kmalloc_array()` with `kcalloc()`, but it is not always necessary to clear the whole buffer.
CVE-2025-39931	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: crypto: af_alg - Set merge to zero early in af_alg_sendmsg If an error causes af_alg_sendmsg to abort, ctx->merge may contain a garbage value from the previous loop. This may then trigger a crash on the next entry into af_alg_sendmsg when it attempts to do a merge that can't be done. Fix this by setting ctx->merge to zero near the start of the loop.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-71113	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: crypto: af_alg - zero initialize memory allocated via sock_kmalloc Several crypto user API contexts and requests allocated with sock_kmalloc() were left uninitialized, relying on callers to set fields explicitly. This resulted in the use of uninitialized data in certain error paths or when new fields are added in the future. The ACVP patches also contain two user-space interface files: algif_kpp.c and algif_akcipher.c. These too rely on proper initialization of their context structures. A particular issue has been observed with the newly added 'inflight' variable introduced in af_alg_ctx by commit: 67b164a871af ("crypto: af_alg - Disallow multiple in-flight AIO requests") Because the context is not memset to zero after allocation, the inflight variable has contained garbage values. As a result, af_alg_alloc_areq() has incorrectly returned -EBUSY randomly when the garbage value was interpreted as true: https://github.com/gregkh/linux/blame/master/crypto/af_alg.c#L1209 The check directly tests ctx->inflight without explicitly comparing against true/false. Since inflight is only ever set to true or false later, an uninitialized value has triggered -EBUSY failures. Zero-initializing memory allocated with sock_kmalloc() ensures inflight and other fields start in a known state, removing random issues caused by uninitialized data.
CVE-2024-49990	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: drm/xe/hdcp: Check GSC structure validity Sometimes xe_gsc is not initialized when checked at HDCP capability check. Add gsc structure check to avoid null pointer error.
CVE-2024-50014	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ext4: fix access to uninitialised lock in fc replay path The following kernel trace can be triggered with fstest generic/629 when executed against a filesystem with fast-commit feature enabled: INFO: trying to register non-static key. The code is fine but needs lockdep annotation, or maybe you didn't initialize this object before use? turning off the locking correctness validator. CPU: 0 PID: 866 Comm: mount Not tainted 6.10.0+ #11 Hardware name: QEMU Standard PC (i440FX + PIIX, 1996), BIOS rel-1.16.2-3-gd478f380-prebuilt.qemu.org 04/01/2014 Call Trace: dump_stack_lvl+0x66/0x90 register_lock_class+0x759/0x7d0 __lock_acquire+0x85/0x2630 ? __find_get_block+0xb4/0x380 lock_acquire+0xd1/0x2d0 ? __ext4_journal_get_write_access+0xd5/0x160 _raw_spin_lock+0x33/0x40 ? __ext4_journal_get_write_access+0xd5/0x160 __ext4_journal_get_write_access+0xd5/0x160 ext4_reserve_inode_write+0x61/0xb0 __ext4_mark_inode_dirty+0x79/0x270 ? ext4_ext_replay_set_iblocks+0x2f8/0x450 ext4_ext_replay_set_iblocks+0x330/0x450 ext4_fc_replay+0x14c8/0x1540 ? jread+0x88/0x2e0 ? rcu_is_watching+0x11/0x40 do_one_pass+0x447/0xd00 jbd2_journal_recover+0x139/0x1b0 jbd2_journal_load+0x96/0x390 ext4_load_and_init_journal+0x253/0xd40 ext4_fill_super+0x2cc6/0x3180 ... In the replay path there's an attempt to lock sbi->s_bdev_wb_lock in function ext4_check_bdev_write_error(). Unfortunately, at this point this spinlock has not been initialized yet. Moving it's initialization to an earlier point in __ext4_fill_super() fixes this splat.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-68365	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: fs/ntfs3: Initialize allocated memory before use KMSAN reports: Multiple uninitialized values detected: - KMSAN: uninit-value in ntfs_read_hdr (3) - KMSAN: uninit-value in bcmp (3) Memory is allocated by __getname(), which is a wrapper for kmem_cache_alloc(). This memory is used before being properly cleared. Change kmem_cache_alloc() to kmem_cache_zalloc() to properly allocate and clear memory before use.
CVE-2025-38531	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: iio: common: st_sensors: Fix use of uninitialize device structs Throughout the various probe functions &indio_dev->dev is used before it is initialized. This caused a kernel panic in st_sensors_power_enable() when the call to devm_regulator_bulk_get_enable() fails and then calls dev_err_probe() with the uninitialized device. This seems to only cause a panic with dev_err_probe(), dev_err(), dev_warn() and dev_info() don't seem to cause a panic, but are fixed as well. The issue is reported and traced here: [1]
CVE-2026-23123	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: interconnect: debugfs: initialize src_node and dst_node to empty strings The debugfs_create_str() API assumes that the string pointer is either NULL or points to valid kmalloc() memory. Leaving the pointer uninitialized can cause problems. Initialize src_node and dst_node to empty strings before creating the debugfs entries to guarantee that reads and writes are safe.
CVE-2026-23003	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: ip6_tunnel: use skb_vlan_inet_prepare() in __ip6_tnl_rcv() Blamed commit did not take care of VLAN encapsulations as spotted by syzbot [1]. Use skb_vlan_inet_prepare() instead of pskb_inet_may_pull(). [1] BUG: KMSAN: uninit-value in __INET_ECN_decapsulate include/net/inet_ecn.h:253 [inline] BUG: KMSAN: uninit-value in INET_ECN_decapsulate include/net/inet_ecn.h:275 [inline] BUG: KMSAN: uninit-value in IP6_ECN_decapsulate+0x7a8/0x1fa0 include/net/inet_ecn.h:321 __INET_ECN_decapsulate include/net/inet_ecn.h:253 [inline] INET_ECN_decapsulate include/net/inet_ecn.h:275 [inline] IP6_ECN_decapsulate+0x7a8/0x1fa0 include/net/inet_ecn.h:321 ip6ip6_dscp_ecn_decapsulate+0x16f/0x1b0 net/ipv6/ip6_tunnel.c:729 __ip6_tnl_rcv+0xed9/0x1b50 net/ipv6/ip6_tunnel.c:860 ip6_tnl_rcv+0xc3/0x100 net/ipv6/ip6_tunnel.c:903 gre_rcv+0x1529/0x1b90 net/ipv6/ip6_gre.c:-1 ip6_protocol_deliver_rcu+0x1c89/0x2c60 net/ipv6/ip6_input.c:438 ip6_input_finish+0x1f4/0x4a0 net/ipv6/ip6_input.c:489 NF_HOOK include/linux/netfilter.h:318 [inline] ip6_input+0x9c/0x330 net/ipv6/ip6_input.c:500 ip6_mc_input+0x7ca/0xc10 net/ipv6/ip6_input.c:590 dst_input include/net/dst.h:474 [inline] ip6_rcv_finish+0x958/0x990 net/ipv6/ip6_input.c:79 NF_HOOK include/linux/netfilter.h:318 [inline] ipv6_rcv+0xf1/0x3c0 net/ipv6/ip6_input.c:311 __netif_receive_skb_one_core net/core/dev.c:6139 [inline] __netif_receive_skb+0x1df/0xac0 net/core/dev.c:6252 netif_receive_skb_internal net/core/dev.c:6338 [inline] netif_receive_skb+0x57/0x630 net/core/dev.c:6397 tun_rx_batched+0x1df/0x980 drivers/net/tun.c:1485 tun_get_user+0x5c0e/0x6c60

CVE	Severity	CVSS	CVSS Vector	Component	Description
					drivers/net/tun.c:1953 tun_chr_write_iter+0x3e9/0x5c0 drivers/net/tun.c:1999 new_sync_write fs/read_write.c:593 [inline] vfs_write+0xbe2/0x15d0 fs/read_write.c:686 ksys_write fs/read_write.c:738 [inline] __do_sys_write fs/read_write.c:749 [inline] __se_sys_write fs/read_write.c:746 [inline] __x64_sys_write+0x1fb/0x4d0 fs/read_write.c:746 x64_sys_call+0x30ab/0x3e70 arch/x86/include/generated/asm/syscalls_64.h:2 do_syscall_x64 arch/x86/entry/syscall_64.c:63 [inline] do_syscall_64+0xd3/0xf80 arch/x86/entry/syscall_64.c:94 entry_SYSCALL_64_after_hwframe+0x77/0x7f Uninit was created at: slab_post_alloc_hook mm/slub.c:4960 [inline] slab_alloc_node mm/slub.c:5263 [inline] kmem_cache_alloc_node_noprof+0x9e7/0x17a0 mm/slub.c:5315 kmalloc_reserve+0x13c/0x4b0 net/core/skbuff.c:586 __alloc_skb+0x805/0x1040 net/core/skbuff.c:690 alloc_skb include/linux/skbuff.h:1383 [inline] alloc_skb_with_frags+0xc5/0xa60 net/core/skbuff.c:6712 sock_alloc_send_pskb+0xacc/0xc60 net/core/sock.c:2995 tun_alloc_skb drivers/net/tun.c:1461 [inline] tun_get_user+0x1142/0x6c60 drivers/net/tun.c:1794 tun_chr_write_iter+0x3e9/0x5c0 drivers/net/tun.c:1999 new_sync_write fs/read_write.c:593 [inline] vfs_write+0xbe2/0x15d0 fs/read_write.c:686 ksys_write fs/read_write.c:738 [inline] __do_sys_write fs/read_write.c:749 [inline] __se_sys_write fs/read_write.c:746 [inline] __x64_sys_write+0x1fb/0x4d0 fs/read_write.c:746 x64_sys_call+0x30ab/0x3e70 arch/x86/include/generated/asm/syscalls_64.h:2 do_syscall_x64 arch/x86/entry/syscall_64.c:63 [inline] do_syscall_64+0xd3/0xf80 arch/x86/entry/syscall_64.c:94 entry_SYSCALL_64_after_hwframe+0x77/0x7f CPU: 0 UID: 0 PID: 6465 Comm: syz.0.17 Not tainted syzkaller #0 PREEMPT(none) Hardware name: Google Google Compute Engine/Google Compute Engine, BIOS Google 10/25/2025

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-39833	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: mISDN: hfcpci: Fix warning when deleting uninitialized timer With CONFIG_DEBUG_OBJECTS_TIMERS unloading hfcpci module leads to the following splat: [250.215892] ODEBUG: assert_init not available (active state 0) object: ffffffffc01a3dc0 object type: timer_list hint: 0x0 [250.217520] WARNING: CPU: 0 PID: 233 at lib/debugobjects.c:612 debug_print_object+0x1b6/0x2c0 [250.218775] Modules linked in: hfcpci(-) mISDN_core [250.219537] CPU: 0 UID: 0 PID: 233 Comm: rmmod Not tainted 6.17.0-rc2-g6f713187ac98 #2 PREEMPT(voluntary) [250.220940] Hardware name: QEMU Ubuntu 24.04 PC (i440FX + PIIX, 1996), BIOS 1.16.3-debian-1.16.3-2 04/01/2014 [250.222377] RIP: 0010:debug_print_object+0x1b6/0x2c0 [250.223131] Code: fc ff df 48 89 fa 48 c1 ea 03 80 3c 02 00 75 4f 41 56 48 8b 14 dd a0 4e 01 9f 48 89 ee 48 c7 c7 20 46 01 9f e8 cb 84d [250.225805] RSP: 0018:ffff888015ea7c08 EFLAGS: 00010286 [250.226608] RAX: 0000000000000000 RBX: 0000000000000005 RCX: ffffffff9be93a95 [250.227708] RDX: 1ffff1100d945138 RSI: 0000000000000008 RDI: ffff88806ca289c0 [250.228993] RBP: ffffffff9f014a00 R08: 0000000000000001 R09: ffffed1002bd4f39 [250.230043] R10: ffff888015ea79cf R11: 0000000000000001 R12: 0000000000000001 [250.231185] R13: ffffffff9eea0520 R14: 0000000000000000 R15: ffff888015ea7cc8 [250.232454] FS: 00007f3208f01540(0000) GS:ffff8880caf5a000(0000) knlGS:0000000000000000 [250.233851] CS: 0010 DS: 0000 ES: 0000 CR0: 0000000080050033 [250.234856] CR2: 00007f32090a7421 CR3: 0000000004d63000 CR4: 00000000000006f0 [250.236117] Call Trace: [250.236599] [250.236967] ? trace_irq_enable.constprop.0+0xd4/0x130 [250.237920] debug_object_assert_init+0x1f6/0x310 [250.238762] ? __pfx_debug_object_assert_init+0x10/0x10 [250.239658] ? __lock_acquire+0xdea/0x1c70 [250.240369] __try_to_del_timer_sync+0x69/0x140 [250.241172] ? __pfx___try_to_del_timer_sync+0x10/0x10 [250.242058] ? __timer_delete_sync+0xc6/0x120 [250.242842] ? lock_acquire+0x30/0x80 [250.243474] ? __timer_delete_sync+0xc6/0x120 [250.244262] __timer_delete_sync+0x98/0x120 [250.245015] HFC_cleanup+0x10/0x20 [hfcpci] [250.245704] __do_sys_delete_module+0x348/0x510 [250.246461] ? __pfx___do_sys_delete_module+0x10/0x10 [250.247338] do_syscall_64+0xc1/0x360 [250.247924] entry_SYSCALL_64_after_hwframe+0x77/0x7f Fix this by initializing hfc_tl timer with DEFINE_TIMER macro. Also, use mod_timer instead of manual timeout update.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38441	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: netfilter: flowtable: account for Ethernet header in nf_flow_pppoe_proto() syzbot found a potential access to uninit-value in nf_flow_pppoe_proto() Blamed commit forgot the Ethernet header. BUG: KMSAN: uninit-value in nf_flow_offload_inet_hook+0x7e4/0x940 net/netfilter/nf_flow_table_inet.c:27 nf_flow_offload_inet_hook+0x7e4/0x940 net/netfilter/nf_flow_table_inet.c:27 nf_hook_entry_hookfn include/linux/netfilter.h:157 [inline] nf_hook_slow+0xe1/0x3d0 net/netfilter/core.c:623 nf_hook_ingress include/linux/netfilter_netdev.h:34 [inline] nf_ingress net/core/dev.c:5742 [inline] __netif_receive_skb_core+0x4aff/0x70c0 net/core/dev.c:5837 __netif_receive_skb_one_core net/core/dev.c:5975 [inline] __netif_receive_skb+0xcc/0xac0 net/core/dev.c:6090 netif_receive_skb_internal net/core/dev.c:6176 [inline] netif_receive_skb+0x57/0x630 net/core/dev.c:6235 tun_rx_batched+0x1df/0x980 drivers/net/tun.c:1485 tun_get_user+0x4ee0/0x6b40 drivers/net/tun.c:1938 tun_chr_write_iter+0x3e9/0x5c0 drivers/net/tun.c:1984 new_sync_write fs/read_write.c:593 [inline] vfs_write+0xb4b/0x1580 fs/read_write.c:686 ksys_write fs/read_write.c:738 [inline] __do_sys_write fs/read_write.c:749 [inline]

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38472	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: netfilter: nf_conntrack: fix crash due to removal of uninitialised entry A crash in conntrack was reported while trying to unlink the conntrack entry from the hash bucket list: [exception RIP: __nf_ct_delete_from_lists+172] [..] #7 [ff539b5a2b043aa0] nf_ct_delete at ffffffffc124d421 [nf_conntrack] #8 [ff539b5a2b043ad0] nf_ct_gc_expired at ffffffffc124d999 [nf_conntrack] #9 [ff539b5a2b043ae0] __nf_conntrack_find_get at ffffffffc124efbc [nf_conntrack] [..] The nf_conn struct is marked as allocated from slab but appears to be in a partially initialised state: ct hlist pointer is garbage; looks like the ct hash value (hence crash). ct->status is equal to IPS_CONFIRMED|IPS_DYING, which is expected ct->timeout is 30000 (=30s), which is unexpected. Everything else looks like normal udp conntrack entry. If we ignore ct->status and pretend its 0, the entry matches those that are newly allocated but not yet inserted into the hash: - ct hlist pointers are overloaded and store/cache the raw tuple hash - ct->timeout matches the relative time expected for a new udp flow rather than the absolute 'jiffies' value. If it were not for the presence of IPS_CONFIRMED, __nf_conntrack_find_get() would have skipped the entry. Theory is that we did hit following race: cpu x cpu y cpu z found entry E found entry E E is expired nf_ct_delete() return E to rcu slab init_conntrack E is re-inited, ct->status set to 0 reply tuplehash hnnode.pprev stores hash value. cpu y found E right before it was deleted on cpu x. E is now re-inited on cpu z. cpu y was preempted before checking for expiry and/or confirm bit. ->refcnt set to 1 E now owned by skb ->timeout set to 30000 If cpu y were to resume now, it would observe E as expired but would skip E due to missing CONFIRMED bit. nf_conntrack_confirm gets called sets: ct->status |= CONFIRMED This is wrong: E is not yet added to hashtable. cpu y resumes, it observes E as expired but CONFIRMED: nf_ct_expired() -> yes (ct->timeout is 30s) confirmed bit set. cpu y will try to delete E from the hashtable: nf_ct_delete() -> set DYING bit __nf_ct_delete_from_lists Even this scenario doesn't guarantee a crash: cpu z still holds the table bucket lock(s) so y blocks: wait for spinlock held by z CONFIRMED is set but there is no guarantee ct will be added to hash: "chaintoolong" or "clash resolution" logic both skip the insert step. reply hnnode.pprev still stores the hash value. unlocks spinlock return NF_DROP In case CPU z does insert the entry into the hashtable, cpu y will unlink E again right away but no crash occurs. Without 'cpu y' race, 'garbage' hlist is of no consequence: ct refcnt remains at 1, eventually skb will be free'd and E gets destroyed via: nf_conntrack_put -> nf_conntrack_destroy -> nf_ct_destroy. To resolve this, move the IPS_CONFIRMED assignment after the table insertion but before the unlock. Pablo points out that the confirm-bit-store could be reordered to happen before hlist add resp. the timeout fixup, so switch to set_bit and before_atomic memory barrier to prevent this. It doesn't matter if other CPUs can observe a newly inserted entry right before the CONFIRMED bit was set: Such event cannot be distinguished from above "E is the old incarnation" case: the entry will be skipped. Also change nf_ct_should_gc() to first check the confirmed bit. The gc sequence is: 1. Check if entry has expired, if not skip to next entry 2. Obtain a reference to the expired entry. 3. Call nf_ct_should_gc() to double-check step 1. nf_ct_should_gc() is thus called only for entries that already failed an expiry check. After this patch, once the confirmed bit check pas ---truncated---

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38691	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: pNFS: Fix uninited ptr deref in block/scsi layout The error occurs on the third attempt to encode extents. When function ext_tree_prepare_commit() reallocates a larger buffer to retry encoding extents, the "layoutupdate_pages" page array is initialized only after the retry loop. But ext_tree_free_commitdata() is called on every iteration and tries to put pages in the array, thus dereferencing uninitialized pointers. An additional problem is that there is no limit on the maximum possible buffer_size. When there are too many extents, the client may create a layoutcommit that is larger than the maximum possible RPC size accepted by the server. During testing, we observed two typical scenarios. First, one memory page for extents is enough when we work with small files, append data to the end of the file, or preallocate extents before writing. But when we fill a new large file without preallocating, the number of extents can be huge, and counting the number of written extents in ext_tree_encode_commit() does not help much. Since this number increases even more between unlocking and locking of ext_tree, the reallocated buffer may not be large enough again and again.
CVE-2025-39812	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: sctp: initialize more fields in sctp_v6_from_sk() syzbot found that sin6_scope_id was not properly initialized, leading to undefined behavior. Clear sin6_scope_id and sin6_flowinfo. BUG: KMSAN: uninit-value in __sctp_v6_cmp_addr+0x887/0x8c0 net/sctp/ipv6.c:649 __sctp_v6_cmp_addr+0x887/0x8c0 net/sctp/ipv6.c:649 sctp_inet6_cmp_addr+0x4f2/0x510 net/sctp/ipv6.c:983 sctp_bind_addr_conflict+0x22a/0x3b0 net/sctp/bind_addr.c:390 sctp_get_port_local+0x21eb/0x2440 net/sctp/socket.c:8452 sctp_get_port net/sctp/socket.c:8523 [inline] sctp_listen_start net/sctp/socket.c:8567 [inline] sctp_inet_listen+0x710/0xfd0 net/sctp/socket.c:8636 __sys_listen_socket net/socket.c:1912 [inline] __sys_listen net/socket.c:1927 [inline] __do_sys_listen net/socket.c:1932 [inline] __se_sys_listen net/socket.c:1930 [inline] __x64_sys_listen+0x343/0x4c0 net/socket.c:1930 x64_sys_call+0x271d/0x3e20 arch/x86/include/generated/asm/syscalls_64.h:51 do_syscall_x64 arch/x86/entry/syscall_64.c:63 [inline] do_syscall_64+0xd9/0x210 arch/x86/entry/syscall_64.c:94 entry_SYSCALL_64_after_hwframe+0x77/0x7f Local variable addr.i.i created at: sctp_get_port net/sctp/socket.c:8515 [inline] sctp_listen_start net/sctp/socket.c:8567 [inline] sctp_inet_listen+0x650/0xfd0 net/sctp/socket.c:8636 __sys_listen_socket net/socket.c:1912 [inline] __sys_listen net/socket.c:1927 [inline] __do_sys_listen net/socket.c:1932 [inline] __se_sys_listen net/socket.c:1930 [inline] __x64_sys_listen+0x343/0x4c0 net/socket.c:1930

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-38644	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: wifi: mac80211: reject TDLS operations when station is not associated syzbot triggered a WARN in ieee80211_tdls_oper() by sending NL80211_TDLS_ENABLE_LINK immediately after NL80211_CMD_CONNECT, before association completed and without prior TDLS setup. This left internal state like sdata->u.mgd.tdls_peer uninitialized, leading to a WARN_ON() in code paths that assumed it was valid. Reject the operation early if not in station mode or not associated.
CVE-2025-38532	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net: libwx: properly reset Rx ring descriptor When device reset is triggered by feature changes such as toggling Rx VLAN offload, wx->do_reset() is called to reinitialize Rx rings. The hardware descriptor ring may retain stale values from previous sessions. And only set the length to 0 in rx_desc[0] would result in building malformed SKBs. Fix it to ensure a clean slate after device reset. [549.186435] [C16] ------[cut here]------ [549.186457] [C16] kernel BUG at net/core/skbuff.c:2814! [549.186468] [C16] Oops: invalid opcode: 0000 [#1] SMP NOPTI [549.186472] [C16] CPU: 16 UID: 0 PID: 0 Comm: swapper/16 Kdump: loaded Not tainted 6.16.0-rc4+ #23 PREEMPT(voluntary) [549.186476] [C16] Hardware name: Micro-Star International Co., Ltd. MS-7E16/X670E GAMING PLUS WIFI (MS-7E16), BIOS 1.90 12/31/2024 [549.186478] [C16] RIP: 0010:__pskb_pull_tail+0x3ff/0x510 [549.186484] [C16] Code: 06 f0 ff 4f 34 74 7b 4d 8b 8c 24 c8 00 00 00 45 8b 84 24 c0 00 00 00 e9 c8 fd ff ff 48 c7 44 24 08 00 00 00 00 e9 5e fe ff ff <0f> 0b 31 c0 e9 23 90 5b ff 41 f7 c6 ff 0f 00 00 75 bf 49 8b 06 a8 [549.186487] [C16] RSP: 0018:ffffb391c0640d70 EFLAGS: 00010282 [549.186490] [C16] RAX: 00000000ffffffff2 RBX: ffff8fe7e4d40200 RCX: 00000000ffffffff2 [549.186492] [C16] RDX: ffff8fe7c3a4bf8e RSI: 0000000000000180 RDI: ffff8fe7c3a4bf40 [549.186494] [C16] RBP: ffffb391c0640da8 R08: ffff8fe7c3a4c0c0 R09: 000000000000000e [549.186496] [C16] R10: ffffb391c0640d88 R11: 000000000000000e R12: ffff8fe7e4d40200 [549.186497] [C16] R13: 00000000ffffffff2 R14: ffff8fe7fa01a000 R15: 00000000ffffffff2 [549.186499] [C16] FS: 0000000000000000(0000) GS:ffff8fef5ae40000(0000) knlGS:0000000000000000 [549.186502] [C16] CS: 0010 DS: 0000 ES: 0000 CR0: 0000000080050033 [549.186503] [C16] CR2: 00007f77d81d6000 CR3: 000000051a032000 CR4: 0000000000750ef0 [549.186505] [C16] PKRU: 55555554 [549.186507] [C16] Call Trace: [549.186510] [C16] [549.186513] [C16] ? srso_alias_return_thunk+0x5/0xfbef5 [549.186517] [C16] __skb_pad+0xc7/0xf0 [549.186523] [C16] wx_clean_rx_irq+0x355/0x3b0 [libwx] [549.186533] [C16] wx_poll+0x92/0x120 [libwx] [549.186540] [C16] __napi_poll+0x28/0x190 [549.186544] [C16] net_rx_action+0x301/0x3f0 [549.186548] [C16] ? srso_alias_return_thunk+0x5/0xfbef5 [549.186551] [C16] ? __raw_spin_lock_irqsave+0x1e/0x50 [549.186554] [C16] ? srso_alias_return_thunk+0x5/0xfbef5 [549.186557] [C16] ? wake_up_nohz_cpu+0x35/0x160 [549.186559] [C16] ? srso_alias_return_thunk+0x5/0xfbef5 [549.186563] [C16] handle_softirqs+0xf9/0x2c0 [549.186568] [C16] __irq_exit_rcu+0xc7/0x130 [549.186572] [C16] common_interrupt+0xb8/0xd0 [549.186576] [C16] [549.186577] [C16] [549.186579] [C16] asm_common_interrupt+0x22/0x40 [549.186582] [C16] RIP: 0010:cpuidle_enter_state+0xc2/0x420 [

CVE	Severity	CVSS	CVSS Vector	Component	Description
					549.186585] [C16] Code: 00 00 e8 11 0e 5e ff e8 ac f0 ff ff 49 89 c5 0f 1f 44 00 00 31 ff e8 0d ed 5c ff 45 84 ff 0f 85 40 02 00 00 fb 0f 1f 44 00 00 <45> 85 f6 0f 88 84 01 00 00 49 63 d6 48 8d 04 52 48 8d 04 82 49 8d [549.186587] [C16] RSP: 0018:ffffb391c0277e78 EFLAGS: 00000246 [549.186590] [C16] RAX: ffff8fef5ae40000 RBX: 0000000000000003 RCX: 0000000000000000 [549.186591] [C16] RDX: 0000007fde0faac5 RSI: ffffffff826e53f6 RDI: ffffffff826fa9b3 [549.186593] [C16] RBP: ffff8fe7c3a20800 R08: 0000000000000002 R09: 0000000000000000 [549.186595] [C16] R10: 0000000000000000 R11: 000000000000ffff R12: ffffffff82ed7a40 [549.186596] [C16] R13: 0000007fde0faac5 R14: 0000000000000003 R15: 0000000000000000 [549.186601] [C16] ? cpuidle_enter_state+0xb3/0x420 [549.186605] [C16] cpuidle_en ---truncated---
CVE-2025-38601	MEDIUM	5.5	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: wifi: ath11k: clear initialized flag for deinit-ed srng lists In a number of cases we see kernel panics on resume due to ath11k kernel page fault, which happens under the following circumstances: 1) First ath11k_hal_dump_srng_stats() call Last interrupt received for each group: ath11k_pci 0000:01:00.0: group_id 0 22511ms before ath11k_pci 0000:01:00.0: group_id 1 14440788ms before [..] ath11k_pci 0000:01:00.0: failed to receive control response completion, polling.. ath11k_pci 0000:01:00.0: Service connect timeout ath11k_pci 0000:01:00.0: failed to connect to HTT: -110 ath11k_pci 0000:01:00.0: failed to start core: -110 ath11k_pci 0000:01:00.0: firmware crashed: MHI_CB_EE_RDDM ath11k_pci 0000:01:00.0: already resetting count 2 ath11k_pci 0000:01:00.0: failed to wait wlan mode request (mode 4): -110 ath11k_pci 0000:01:00.0: qmi failed to send wlan mode off: -110 ath11k_pci 0000:01:00.0: failed to reconfigure driver on crash recovery [..] 2) At this point reconfiguration fails (we have 2 resets) and ath11k_core_reconfigure_on_crash() calls ath11k_hal_srng_deinit() which destroys srng lists. However, it does not reset per-list ->initialized flag. 3) Second ath11k_hal_dump_srng_stats() call sees stale ->initialized flag and attempts to dump srng stats: Last interrupt received for each group: ath11k_pci 0000:01:00.0: group_id 0 66785ms before ath11k_pci 0000:01:00.0: group_id 1 14485062ms before ath11k_pci 0000:01:00.0: group_id 2 14485062ms before ath11k_pci 0000:01:00.0: group_id 3 14485062ms before ath11k_pci 0000:01:00.0: group_id 4 14780845ms before ath11k_pci 0000:01:00.0: group_id 5 14780845ms before ath11k_pci 0000:01:00.0: group_id 6 14485062ms before ath11k_pci 0000:01:00.0: group_id 7 66814ms before ath11k_pci 0000:01:00.0: group_id 8 68997ms before ath11k_pci 0000:01:00.0: group_id 9 67588ms before ath11k_pci 0000:01:00.0: group_id 10 69511ms before BUG: unable to handle page fault for address: ffffa007404eb010 #PF: supervisor read access in kernel mode #PF: error_code(0x0000) - not-present page PGD 100000067 P4D 100000067 PUD 10022d067 PMD 100b01067 PTE 0 Oops: 0000 [#1] PREEMPT SMP NOPTI RIP: 0010:ath11k_hal_dump_srng_stats+0x2b4/0x3b0 [ath11k] Call Trace: ? __die_body+0xae/0xb0 ? page_fault_oops+0x381/0x3e0 ? exc_page_fault+0x69/0xa0 ? asm_exc_page_fault+0x22/0x30 ? ath11k_hal_dump_srng_stats+0x2b4/0x3b0 [ath11k (HASH:6cea 4)] ath11k_qmi_driver_event_work+0xbd/0x1050 [ath11k (HASH:6cea 4)] worker_thread+0x389/0x930 kthread+0x149/0x170 Clear per-list ->initialized flag in ath11k_hal_srng_deinit().

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2024-35843	MEDIUM	6.8	CVSS:3.1/AV:L/AC:L/PR:N/UI:N/S:U/C:L/I:N/A:H	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: iommu/vt-d: Use device rbtree in iopf reporting path The existing I/O page fault handler currently locates the PCI device by calling pci_get_domain_bus_and_slot(). This function searches the list of all PCI devices until the desired device is found. To improve lookup efficiency, replace it with device_rbtree_find() to search the device within the probed device rbtree. The I/O page fault is initiated by the device, which does not have any synchronization mechanism with the software to ensure that the device stays in the probed device tree. Theoretically, a device could be released by the IOMMU subsystem after device_rbtree_find() and before iopf_get_dev_fault_param(), which would cause a use-after-free problem. Add a mutex to synchronize the I/O page fault reporting path and the IOMMU release device path. This lock doesn't introduce any performance overhead, as the conflict between I/O page fault reporting and device releasing is very rare.
CVE-2025-12781	MEDIUM	5.3	CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:N/I:L/A:N	python3 3.12.11	When passing data to the b64decode(), standard_b64decode(), and urlsafe_b64decode() functions in the "base64" module the characters "+/" will always be accepted, regardless of the value of "altchars" parameter, typically used to establish an "alternative base64 alphabet" such as the URL safe alphabet. This behavior matches what is recommended in earlier base64 RFCs, but newer RFCs now recommend either dropping characters outside the specified base64 alphabet or raising an error. The old behavior has the possibility of causing data integrity issues. This behavior can only be insecure if your application uses an alternate base64 alphabet (without "+/"). If your application does not use the "altchars" parameter or the urlsafe_b64decode() function, then your application does not use an alternative base64 alphabet. The attached patches DOES NOT make the base64-decode behavior raise an error, as this would be a change in behavior and break existing programs. Instead, the patch deprecates the behavior which will be replaced with the newly recommended behavior in a future version of Python. Users are recommended to mitigate by verifying user-controlled inputs match the base64 alphabet they are expecting or verify that their application would not be affected if the b64decode() functions accepted "+" or "/" outside of altchars.
CVE-2025-12084	MEDIUM	5.3	CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:N/I:N/A:L	python3 3.12.11	When building nested elements using xml.dom.minidom methods such as appendChild() that have a dependency on _clear_id_cache() the algorithm is quadratic. Availability can be impacted when building excessively nested documents.
CVE-2025-13836	MEDIUM	7.5	CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:N/I:N/A:H	python3 3.12.11	When reading an HTTP response from a server, if no read amount is specified, the default behavior will be to use Content-Length. This allows a malicious server to cause the client to read large amounts of data into memory, potentially causing OOM or other DoS.
CVE-2024-57256	MEDIUM	6.8	CVSS:3.1/AV:P/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H	u-boot-rittal 2018.09	An integer overflow in ext4fs_read_symlink in Das U-Boot before 2025.01-rc1 occurs for zalloc (adding one to an le32 variable) via a crafted ext4 filesystem with an inode size of 0xffffffff, resulting in a malloc of zero and resultant memory overwrite.
CVE-2024-57254	MEDIUM	6.8	CVSS:3.1/AV:P/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H	u-boot-rittal 2018.09	An integer overflow in sqfs_inode_size in Das U-Boot before 2025.01-rc1 occurs in the symlink size calculation via a crafted squashfs filesystem.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2024-57255	MEDIUM	6.8	CVSS:3.1/AV:P/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H	u-boot-rittal 2018.09	An integer overflow in sqfs_resolve_symlink in Das U-Boot before 2025.01-rc1 occurs via a crafted squashfs filesystem with an inode size of 0xffffffff, resulting in a malloc of zero and resultant memory overwrite.
CVE-2024-57259	MEDIUM	6.8	CVSS:3.1/AV:P/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H	u-boot-rittal 2018.09	sqfs_search_dir in Das U-Boot before 2025.01-rc1 exhibits an off-by-one error and resultant heap memory corruption for squashfs directory listing because the path separator is not considered in a size calculation.
CVE-2019-11690	MEDIUM	5.9	CVSS:3.0/AV:N/AC:H/PR:N/UI:N/S:U/C:H/I:N/A:N	u-boot-rittal 2018.09	gen_rand_uuid in lib/uuid.c in Das U-Boot v2014.04 through v2019.04 lacks an srand call, which allows attackers to determine UUID values in scenarios where CONFIG_RANDOM_UUID is enabled, and Das U-Boot is relied upon for UUID values of a GUID Partition Table of a boot device.
CVE-2026-22184	MEDIUM	7.8	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	zlib 1.3.1	zlib versions up to and including 1.3.1.2 include a global buffer overflow in the untgz utility located under contrib/untgz. The vulnerability is limited to the standalone demonstration utility and does not affect the core zlib compression library. The flaw occurs when a user executes the untgz command with an excessively long archive name supplied via the command line, leading to an out-of-bounds write in a fixed-size global buffer.
CVE-2025-46394	LOW	3.3	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:L/A:N	busybox 1.36.1	In tar in BusyBox through 1.37.0, a TAR archive can have filenames hidden from a listing through the use of terminal escape sequences.
CVE-2025-71148	LOW	3.3	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:L/I:N/A:N	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: net/handshake: restore destructor on submit failure handshake_req_submit() replaces sk->sk_destruct but never restores it when submission fails before the request is hashed. handshake_sk_destruct() then returns early and the original destructor never runs, leaking the socket. Restore sk_destruct on the error path.
CVE-2026-22978	LOW	3.3	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:L/I:N/A:N	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: wifi: avoid kernel-infoleak from struct iw_point struct iw_point has a 32bit hole on 64bit arches. struct iw_point { void __user *pointer; /* Pointer to the data (in user space) */ __u16 length; /* number of fields or size in bytes */ __u16 flags; /* Optional params */ }; Make sure to zero the structure to avoid disclosing 32bits of kernel data to user space.
CVE-2024-57898	LOW	3.3	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:L/A:N	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: wifi: cfg80211: clear link ID from bitmap during link delete after clean up Currently, during link deletion, the link ID is first removed from the valid_links bitmap before performing any clean-up operations. However, some functions require the link ID to remain in the valid_links bitmap. One such example is cfg80211_cac_event(). The flow is - nl80211_remove_link() cfg80211_remove_link() ieee80211_del_intf_link() ieee80211_vif_set_links() ieee80211_vif_update_links() ieee80211_link_stop() cfg80211_cac_event() cfg80211_cac_event() requires link ID to be present but it is cleared already in cfg80211_remove_link(). Ultimately, WARN_ON() is hit. Therefore, clear the link ID from the bitmap only after completing the link clean-up.
CVE-2025-39964	LOW	3.3	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:L	linux-rittal 6.6.96+git	In the Linux kernel, the following vulnerability has been resolved: crypto: af_alg - Disallow concurrent writes in af_alg_sendmsg Issuing two writes to the same af_alg socket is bogus as the data will be interleaved in an unpredictable fashion. Furthermore, concurrent writes may create inconsistencies in the internal socket state. Disallow this by adding a new ctx->write field that indiciates exclusive ownership for writing.

CVE	Severity	CVSS	CVSS Vector	Component	Description
CVE-2025-13837	LOW	5.5	CVSS:3.1/AV:L/AC:L/PR:N/UI:R/S:U/C:N/I:N/A:H	python3 3.12.11	When loading a plist file, the plistlib module reads data in size specified by the file itself, meaning a malicious file can cause OOM and DoS issues
CVE-2024-57257	LOW	2.4	CVSS:3.1/AV:P/AC:L/PR:N/UI:N/S:U/C:N/I:N/A:L	u-boot-rittal 2018.09	A stack consumption issue in sqfs_size in Das U-Boot before 2025.01-rc1 occurs via a crafted squashfs filesystem with deep symlink nesting.
CVE-2025-38073	UNASSIGNED	N/A	N/A	linux-rittal 6.6.96+git	Rejected reason: This CVE ID has been rejected or withdrawn by its CVE Numbering Authority.
CVE-2025-37942	UNASSIGNED	N/A	N/A	linux-rittal 6.6.96+git	Rejected reason: This CVE ID has been rejected or withdrawn by its CVE Numbering Authority.
CVE-2023-52979	UNASSIGNED	N/A	N/A	linux-rittal 6.6.96+git	Rejected reason: This CVE ID has been rejected or withdrawn by its CVE Numbering Authority.
CVE-2025-71149	UNASSIGNED	N/A	N/A	linux-rittal 6.6.96+git	Rejected reason: This CVE ID has been rejected or withdrawn by its CVE Numbering Authority.

Recommended Actions

Rittal recommends updating affected devices to firmware version **v10.1.2** or later to address the vulnerabilities listed in this bulletin.

Contact

For questions regarding this bulletin, please contact:
Rittal PSIRT – psirt@rittal.com

Disclaimer

This security bulletin is provided "as is" without warranty of any kind. Rittal does not accept any liability for damages resulting from the use of this information. The information in this bulletin is subject to change without notice. Users are advised to verify the applicability of this information to their specific environment. This document may be updated as new information becomes available.