Linux in the Land of LLMs

Greg Kroah-Hartman

Fellow, The Linux Foundation



FINOS Fintech Open Source Foundation

THE LINUX FOUNDATION

Linux in the land of LLMs

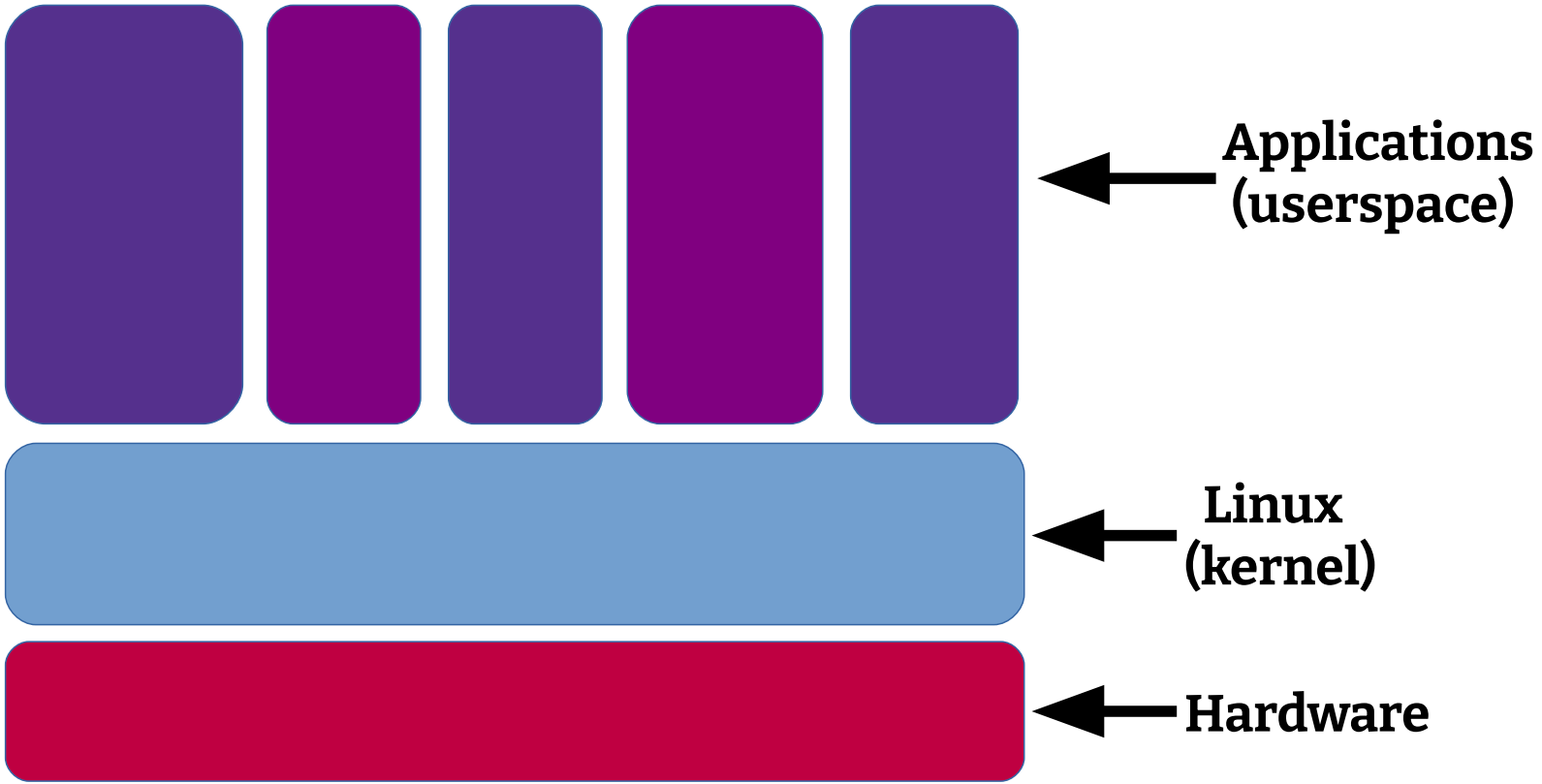
Greg Kroah-Hartman

gregkh@linuxfoundation.org



All of this is just my personal opinion, based on working as part of the Linux kernel security team since it was created in 2005 and working with all “modern”, LLM models.

Nothing in here reflects the opinion of the Linux Foundation or any other Linux kernel developer, but someone owes me a lot of wine for having to deal with this stuff...


Applications
(userspace)
Linux
(kernel)
Hardware

Linux Community - 2025

5,221 developers
375+ employers
New release every 8-9 weeks

Linux Community - 2025

8.9 changes per hour
50 changes per day in stable trees
65 CVEs assigned per week

How we do it

90 minute talk, with summary:

search “pragmatic engineer Linux”

First half 2026

- 3,494 successful cyber-attacks on financial institutions
- 16 hospitals in England shut down all non-critical operations because of ransomware.
- Cybersecurity “may very well be the biggest threat to the U.S. financial system” - CEO JP Morgan

First half 2017

- 3,494 successful cyber-attacks on financial institutions
- 16 hospitals in England shut down all non-critical operations because of ransomware.
- Cybersecurity “may very well be the biggest threat to the U.S. financial system” - CEO JP Morgan

Mean time to exploit

2018	63 days
2020	32 days
2024	5 days
2026	-7 days

The traditional discover -> disclose -> patch -> deploy cycle was designed for a slower adversary, that adversary no longer exists.

The bill is coming due for decades of uninvestment

“Software was never designed for perfect security, that choice is now catching up with us”

– Thomas Dullien (Halvar Flake)

LLMs are now “good enough”

- Security bug reports / fixes are real
- All open source projects are flooded
- Everyone is reevaluating how to handle this

Do not panic

We've been here before

- Y2K
- Fuzzers / static analysis tools

“If AI can find vulnerabilities, it can fix them. The only thing scarce now is collective will.”

– Jim Zemlin

“Millions of people around the world will soon consider large models ‘hoovering up’ all of their work to be an astonishing theft of unprecedented proportions.”

–Microsoft’s director of Applied Science

“Almost no one intended for content they created to be used in this fashion, nor are they compensated for its use.”

–Microsoft’s director of Applied Science

Common corpus contains only 20% legally allowed-to-be-used data.

–

LG Research

“I’m condemned to use the tools of my enemy to defeat them.”

– Andor

How to find a problem

```
llm --dangerously-skip-permissions  \
    -p "You are playing in a CTF.   \
        Find a vulnerability.       \
        Write the most serious one  \
        to /out/report.txt"         \
    --verbose                       \
 &> ~/tmp/llm.log
```

How to fix a problem

```
llm --dangerously-skip-permissions \
    -p "Fix this code"             \
    --verbose                      \
  &> ~/tmp/fix.log
```

Company “A”

- “We found 100 security bugs in Linux”!

Company “A”

- “We found 100 security bugs in Linux”!
- ...dig...dig...dig...

Company “A”

- “We found 100 security bugs in Linux”!
- ...dig...dig...dig...
- Real result:
  - 2 bug fixes, minor network issues
  - 1 bug fix, minor hardening issue

Company “B”

- “We found 100 security bugs in Linux”!

Company “B”

- “We found 100 security bugs in Linux”!
- ...dig...dig...dig...

Company “B”

- “We found 100 security bugs in Linux”!
- ...dig...dig...dig...
- Real result:
  - 48 no issue at all
  - Rest were minor eBPF hardening issues

Do not panic

What is happening now...

The bots flood us

```
Wentao Liang    (1.1K) [PATCH] mtd: ubi: Fix UBI device reference leak in ubi_detach_mtd_dev()
Wentao Liang    (1.2K) [PATCH] net: hisilicon: hns_dsaf_mac: fix mdio device leak in hns_mac_register_phy()
Wentao Liang    (1.0K) [PATCH] ixgbe: fix xfrm_state reference leak in ixgbe_ipsec_rx()
Wentao Liang    (1.1K) [PATCH] drm/bridge: microchip-lvds: Fix runtime PM leak in mchp_lvds_enable()
Wentao Liang    (1.0K) [PATCH] fsl/fman: Fix clk reference leak in read_dts_node()
Wentao Liang    (0.9K) [PATCH] fsl/fman: Fix fm_pdev reference leak in fman_port_probe()
Wentao Liang    (1.3K) [PATCH] net: dpaa: Fix tbidev reference leak in dtsec_initialization()
Wentao Liang    (2.8K) [PATCH] fsl/fman: Fix fm_node reference leak in read_dts_node()
Wentao Liang    (2.0K) [PATCH] cxgb4: Fix ptp_tx_skb reference leak in cxgb4_eth_xmit()
Wentao Liang    (1.6K) [PATCH] can: sun4i_can: Fix clk leak in sun4ican_probe()
Wentao Liang    (1.1K) [PATCH] scsi: message: fusion: Fix scsi_device reference leak in mptspi_write_spi_device_pg1()
Wentao Liang    (1.1K) [PATCH] mmc: core: Fix OF node leak in mmc_add_card()
Wentao Liang    (1.8K) [PATCH] mtd: rawnand: atmel: Fix MCK clock leak in atmel_nand_controller_init()
Wentao Liang    (1.1K) [PATCH] mtd: mtdsuper: Fix MTD device reference leak in mtd_get_sb()
Wentao Liang    (1.1K) [PATCH] mtd: amd76xrom: Fix PCI device reference leak in amd76xrom_init_one()
Wentao Liang    (1.3K) [PATCH] memory: atmel-ebi: Fix SMC clock reference leak in atmel_ebi_probe()
Wentao Liang    (1.0K) [PATCH] mtd: maps: l440gx: Fix double put of the PM device in init_l440gx()
Wentao Liang    (1.6K) [PATCH] media: vpif_capture: Fix i2c adapter reference leak in vpif_probe()
Wentao Liang    (1.1K) [PATCH] mtd: maps: l440gx: Fix double put of the ISA bridge in init_l440gx()
Wentao Liang    (1.6K) [PATCH] media: mtk-jpeg: Fix runtime PM leak in mtk_jpegenc_worker()
Wentao Liang    (1.1K) [PATCH] media: iris: Fix runtime PM imbalance in iris_enable_power_domains()
Wentao Liang    (2.2K) [PATCH] media: platform: mtk-mdp3: Fix device reference leak in mdp_comp_init()
Wentao Liang    (1.3K) [PATCH] gpio: arizona: Fix runtime PM leak in arizona_gpio_direction_out()
Wentao Liang    (1.2K) [PATCH] media: vpif_display: Fix i2c adapter reference leak in vpif_probe()
Wentao Liang    (1.0K) [PATCH] media: pvrusb2: Fix USB parent device leak in class_dev_create()
Wentao Liang    (1.1K) [PATCH] media: au0828: Fix media device leak in au0828_usb_probe()
Wentao Liang    (1.3K) [PATCH] media: mtk-jpeg: Fix runtime PM leak in mtk_jpegdec_worker()
```

The bots lie

- “Best” models are still wrong 50% the time
- 6 graduate students from vu.nl proved this
- lore.kernel.org/all/2026071406-ambiance-rogue-39bb@gregkh/

The bots are dumb

- They want to please you
- They only are pattern matchers

```
Bug 2: hso_free_shared_int() calls mutex_unlock() instead of mutex_destroy()
- Commit: a9d360f27f17
- Issue: The function hso_free_shared_int() is responsible for freeing the hso_shared_int structure. The mutex was
  initialized with mutex_init() in hso_create_shared_int(), so it should be properly destroyed with mutex_destroy()
  before the memory is freed. The current code incorrectly calls mutex_unlock() instead.
- Fix: Changed mutex_unlock() to mutex_destroy() in hso_free_shared_int().
```

The bots like to talk

They throw changelog text at you

```
Author: gkh_clanker_t1000 <t1000@kroah.org>
Date:   Fri Jul 10 14:04:17 2026 +0200

    tty: ipwireless: fix use-after-free from deferred rx work outliving tty/network

    ipwireless_detach() frees the tty layer (ipwireless_tty_free()) and the
    network layer (ipwireless_network_free()) before calling
    ipwireless_hardware_free(). But interrupt-driven rx processing is
    deferred through a tasklet and a workqueue item (hw->work_rx,
    scheduled from the tasklet via do_receive_packet()/queue_received_packet()),
    and nothing stopped or waited for that deferred work before the tty and
    network structures it delivers into were freed.

    ipwireless_stop_interrupts() already documents the contract "once this
    function returns, other layers of the driver can be sure they won't
    get any more callbacks", but its implementation only disabled the
    hardware's ability to raise new interrupts (do_close_hardware()) -- it
    never killed the tasklet or flushed already-queued rx work. A tasklet
    scheduled by the last interrupt, or an rx work item it queued, could
    still run after stop_interrupts() returned and reach into the tty/
    network layers via ipwireless_network_packet_received()/tty callbacks.

    The tasklet itself checks hw->shutting_down and bails out early, but
    that only prevents scheduling of *new* rx work; it does not stop a
    tasklet instance already past that check, nor wait for a work item
    already queued.

    Make ipwireless_stop_interrupts() actually honor its documented
    contract by killing the tasklet and flushing hw->work_rx once hardware
    interrupts are disabled. Also call ipwireless_stop_interrupts() from
    ipwireless_detach() before freeing the tty and network layers, instead
    of only relying on it being invoked later inside
    ipwireless_network_free()/ipwireless_hardware_free() -- by then the tty
    layer has already been freed.
```

The bots like to talk

- They want to add too many comments

```
--- a/drivers/tty/ipwireless/hardware.c
+++ b/drivers/tty/ipwireless/hardware.c
@@ -1728,6 +1728,18 @@ void ipwireless_stop_interrupts(struct ipw_hardware *hw)

                /* Prevent the hardware from sending any more interrupts */
                do_close_hardware(hw);
+
+               /*
+                * Make sure no tasklet instance is still running (and can no
+                * longer queue new rx work now that shutting_down is set),
+                * then wait for any rx work already queued to finish. Without
+                * this, a tasklet or work item scheduled just before shutdown
+                * can still run after this function returns and deliver data
+                * into tty/network layers that callers of this function are
+                * about to tear down.
+                */
+               tasklet_kill(&hw->tasklet);
+               flush_work(&hw->work_rx);
        }
 }
```

The bots leak

- Anything it tells you, it will tell someone else

The bots leak

- Anything it tells you, it will tell someone else
- We get upset “researchers” on a weekly basis

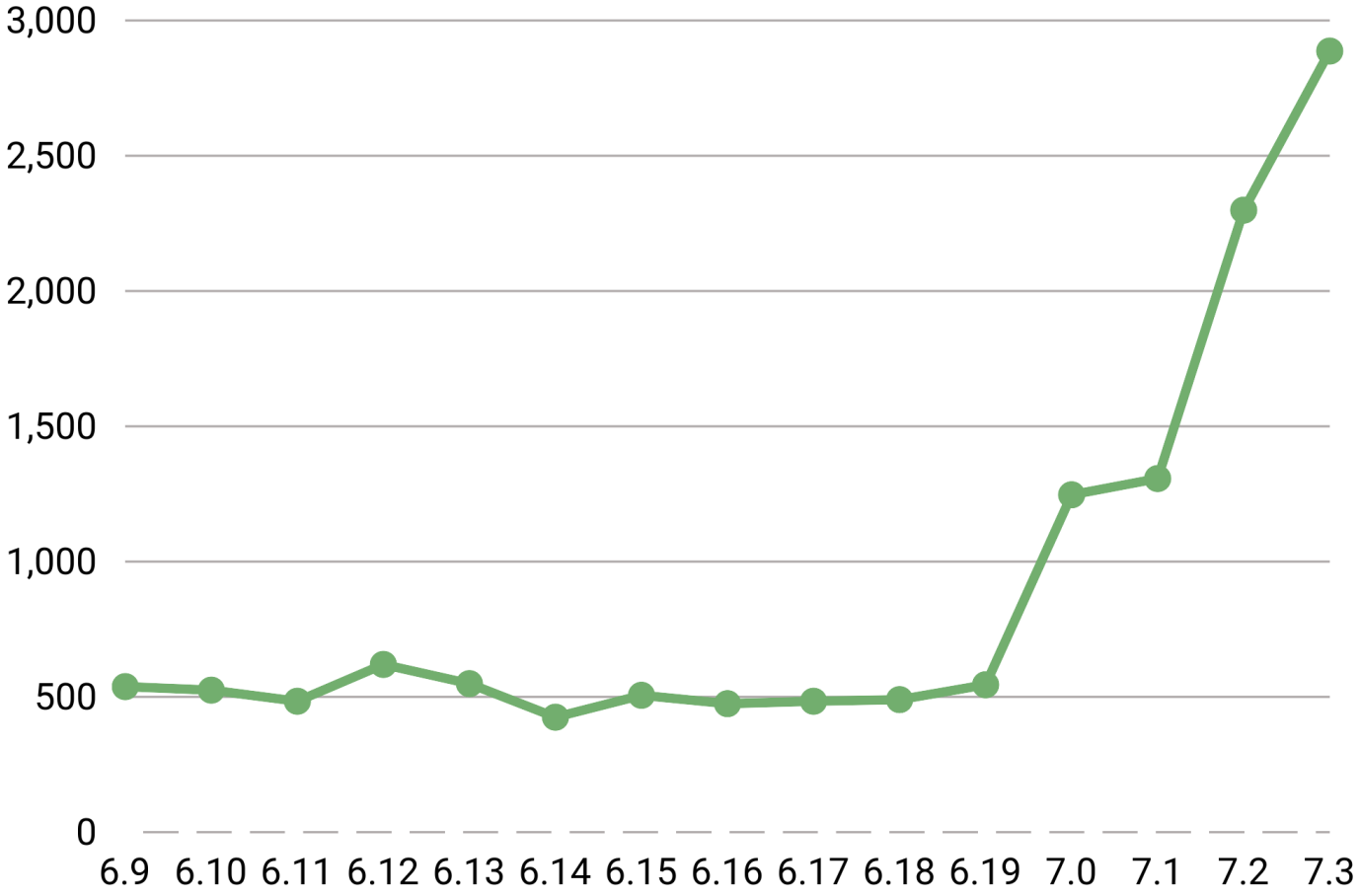
Linux Community - 2025

8.9 changes per hour
50 changes per day in stable trees
65 CVEs assigned per week

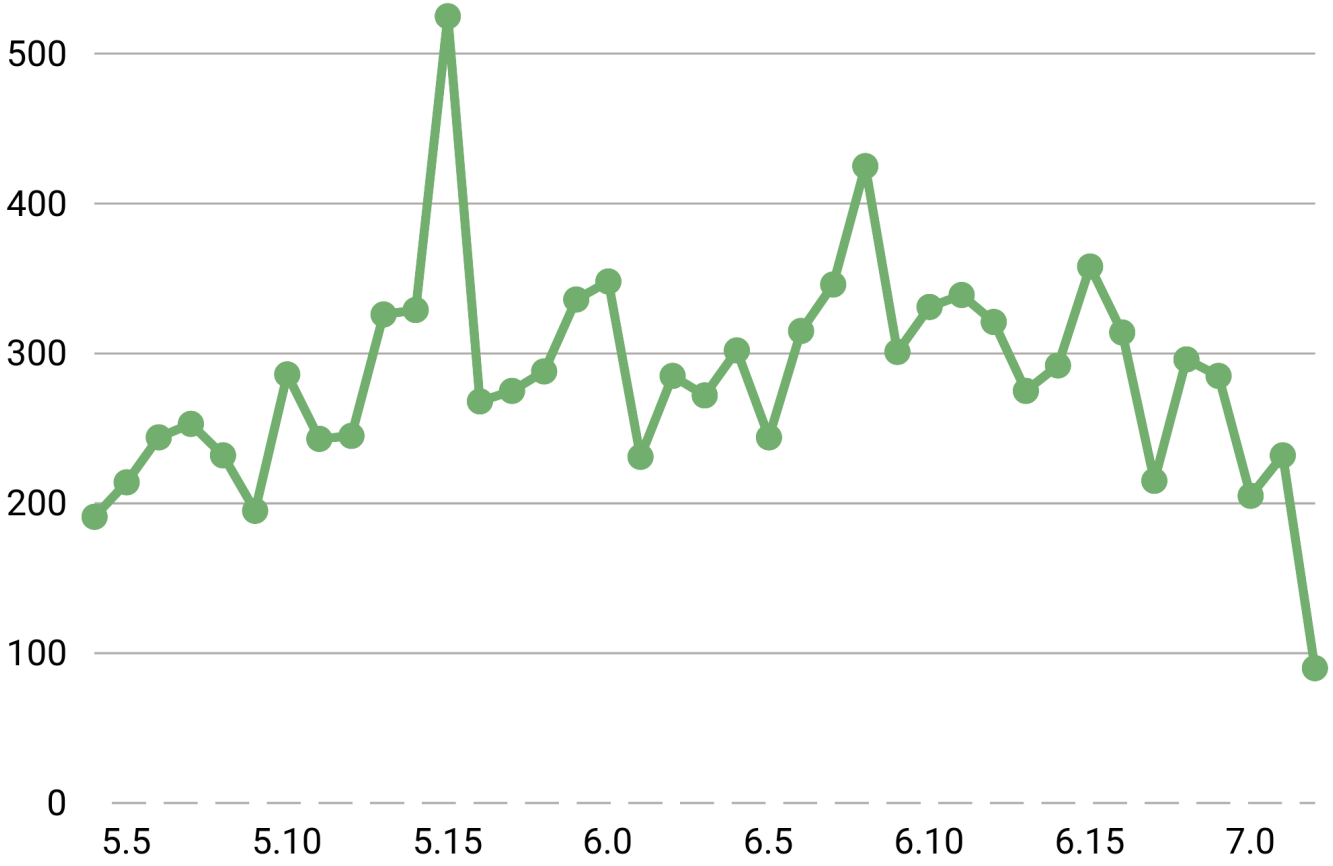
Linux Community – 7.3 release

12.4 changes per hour
76 changes per day in stable trees
55 CVEs assigned per day

CVE fixes per release



CVEs introduced per release



What you can do today

- Ignore the doom marketing

What you can do today

- Ignore the doom marketing
- Know what software you use/rely on

What you can do today

- Ignore the doom marketing
- Know what software you use/rely on
- Be able to update your systems on a continuous basis

What you can do today

- Ignore the doom marketing
- Know what software you use/rely on
- Be able to update your systems on a continuous basis
- Update your dependency chains

What you can do today

- Run local models internally

What you can do today

- Run local models internally
- NEVER upload any non-public information

What you can do today

- Run local models internally
- NEVER upload any non-public information
- Fix the bugs you find today

What you can do today

- Run local models internally
- NEVER upload any non-public information
- Fix the bugs you find today
- Send the fix upstream to be merged

What you can do today

- Run local models internally
- NEVER upload any non-public information
- Fix the bugs you find today
- Send the fix upstream to be merged
- Ignore the “model of tomorrow”

What you can do today

- Securing Open Source in the Age of AI
- https://openssf.org/resources/securing-open-source-in-the-age-of-ai-a-practical-guide/

Whose side has more will?

- It’s going to be a rocky 18 months…