Device drivers workshop

NetDev 0x1a

Paolo Abeni - Red Hat

Red Hat

The typical ‘new device driver’ submission

- Few/tens KLoC
- Grounded around H/W (or firmware) specific twists
- Patches are not written with reviewers in mind
- Submitter can be unfamiliar with the process

`New drivers` handling, the old way

Do:

- Basic API usage checking
- Basic Locking checking
- Reconfiguration (“pre-allocate before free”)
- Rx / Tx paths
- Linters & Nit-picking

`New drivers` handling, the old way [II]

Don’t

- Functional verification
- Line by line in depth review
- Guide step-by-step needed code cleanup and refactor

big surprise: merged code could contains obvious bugs

Welcome to the (not so new) AI overlord

Most of you is familiar with sashiko by now.

+ Line by line review, does not surrender after 10KLoC in a single patch
+ Catch non trivial locking dependencies across different patches
+ Is not fouled by typos, non trivial maths, type conversions

(Luckily?) It’s not perfect yet

- AI sloops/hallucination
- Can/does miss suboptimal implementation choices
- Non deterministic results, even across the same model
- “Taste” is not a metric

AI complains about low-probability corner cases created by too poor kernel frameworks (PCI error handling).

Why is this relevant to me?

- Alibaba EEA driver went through 43 revisions, Nebula matrix is at 21 and running.
- Even basic processing consumes maintainers time
- Increasing everybody’s frustration is not a good long term strategy

Setting expectations

- Your patch/series will require multiple revisions
- In early stages you will not get more feedback on top of sashiko reviews
- When/if stuck on the above, slow down and revisit your process

How can I help?

- AI reviews should be addressed alike traditional ones
  - Debate them or address them, do not ignore them
- Run your own sashiko instance _before_ the submission
- Let the tool help with the process, don’t reuse its output blindly.