



Live Update Compatibility

Ensuring Live Update Success

Logan Odell • Google

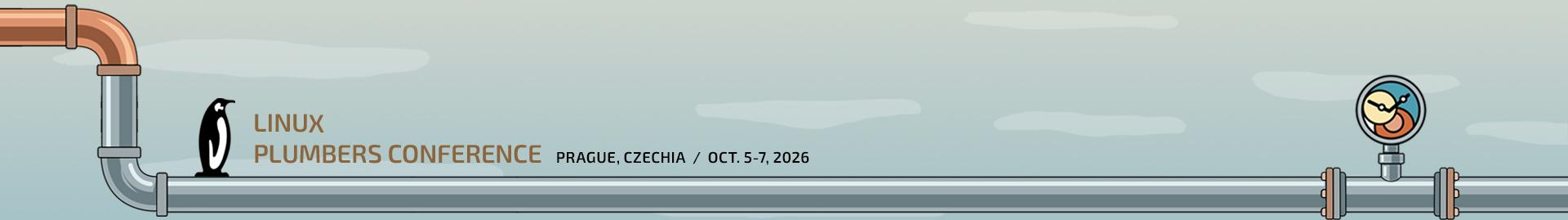


LINUX
PLUMBERS
CONFERENCE

PRAGUE, CZECHIA / OCT. 5-7, 2026

BoF Takeaways from Yesterday

- Subsystems should make efforts to avoid changing compatibility when not strictly required
 - e.g. Introduce a completely optional feature that can be safely ignored, whether present or not, by the new kernel
- Use compatibility strings when a breaking change is needed
- Deal with compatibility issues ad-hoc; no guarantees made (upstream) on compatibility; downstream can maintain their own compatibility matrix



From Last LPC

- General agreement that we can export compatibility data to new .liveupdate_versions section [1]
- What data to export?
 - Compatibility strings
 - General feature flags? Subsystem specific feature flags?

[1] <https://lore.kernel.org/lkml/20251211042624.175517-1-pratyush@kernel.org/>



LINUX
PLUMBERS CONFERENCE

PRAGUE, CZECHIA / OCT. 5-7, 2026



Feature Flags

- Define a common formatted [1] header that goes alongside the compatibility string
- Current proposed flags:
 - Supported - Tells the new kernel that the feature is supported; must be true if old kernel sets active and required
 - Active - Tells the new kernel that the feature being used by the old kernel. Any serialized data for the feature is valid.
 - Required - Tells the new kernel that it needs to support and act on this feature.

[1] <https://lore.kernel.org/all/20260903023452.721732-1-loganodell@google.com/>



LINUX

PLUMBERS CONFERENCE

PRAGUE, CZECHIA / OCT. 5-7, 2026



Expressing Subsystem Incompatibility

- Changing the subsystem's compatibility string
 - Generally done when the format of the proceeding data changes
- Adding an active + required feature
 - Tells the new kernel that it must support this feature or the Live Update will fail
- Supported & !Active & Required
 - Can be used to "slow roll" a feature out to create an intermediate compatible kernel

[1] <https://lore.kernel.org/all/20260903023452.721732-1-loganodell@google.com/>



LINUX

PLUMBERS CONFERENCE

PRAGUE, CZECHIA / OCT. 5-7, 2026



Open Questions

- Do we want to define a common way to express feature compatibility through a single interface or leave it up to subsystems to expose that information?
 - Ex: User wants to live update from kernel X to kernel Y. Subsystem Foo has feature Bar that is active and required in kernel X. Kernel Y does not support feature Bar. User can stop using feature Bar to make the live update compatible
- Are the supported/active/required bits enough or is there any other feature level data we want to expose?

