



# Live Update File Handler Dependency

Samiullah Khawaja - Google



LINUX  
PLUMBERS  
CONFERENCE

PRAGUE, CZECHIA / OCT. 5-7, 2026

# How it works today

- Preservation Order Dependency
  - Might be required for various reasons including:
    - Enforcing immutability.
    - Performance reasons to avoid expensive checks during shutdown.
  - LUO doesn't require or forbid an order.
    - File handlers may enforce ordering based on requirements.
- File handlers should relax dependencies where possible.
  - As reviewers, we should keep them rare.

Note that a dependency enforced today can be relaxed later. One relaxed today can never be enforced later.



LINUX

PLUMBERS CONFERENCE

PRAGUE, CZECHIA / OCT. 5-7, 2026



# Preserve a bag of FDs

- A new ioctl to preserve an Array of FDs.
  - Userspace doesn't need to worry about FDs and orders.
  - Gives kernel the intent and tokens for the preservation of FDs.
  - Allows it to manage the dependency and preservation order internally.
  - `LIVEUPDATE_SESSION_PRESERVE_FDS`
    - Array of FDs and Tokens
- Userspace can still do preservation of FDs individually
  - Compatibility is not broken
  - No new behaviour is enforced



LINUX

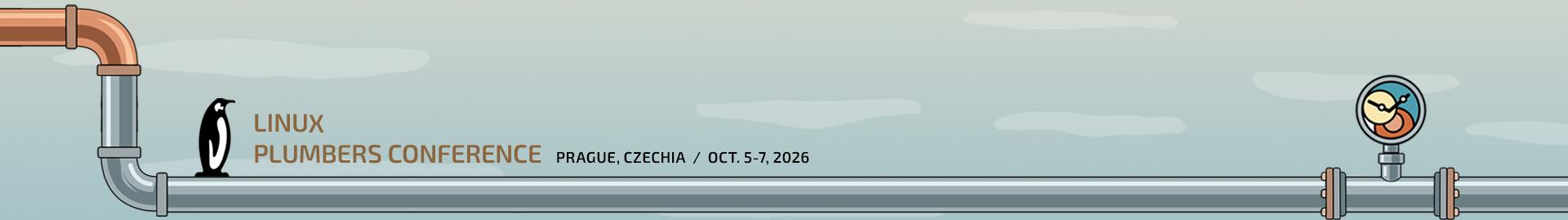
PLUMBERS CONFERENCE

PRAGUE, CZECHIA / OCT. 5-7, 2026



# How to handle it in kernel? - Recursive

- Just trigger preservation of FD\_B from FD\_A
  - Internal API `liveupdate_preserve_fd()`
- Simplest approach but doesn't always work
  - Can introduce nested locking issues
  - FD\_A holds locks while preserving FD\_B
- Rework file handlers to trigger preservation of all dependencies without any locks?
  - Complicates file handler implementation.



# How to handle it in kernel? - Retry

- Preserve FD\_A
  - It uses `liveupdate_get_token_outgoing()` to get token.
  - Fails since FD\_B is not preserved.
- Preserve FD\_B
  - Should succeed
- Try preserving FD\_A again
  - Should succeed now.
- Works, but can be slow
  - Dedicated error code for dependency? ENOENT?
  - Multiple tries and preservation rollbacks
  - Can be  $O(n^2)$  in worst case.



# How to handle it in kernel? - Levels

- Define levels to represent class of file handlers
  - File handlers represent a type of resource or functionality.
  - The dependency and association between them is static
    - A MEMORY PROVIDER is a dependency of (used by) a MAPPER.
    - Levels are optional
  - MEMORY, MAPPER and DEVICE (maybe more)
- These can be used by LUO to order preservation
  - Easy to configure, levels are static and should not change.
    - A memory provider will not become a device FD.
  - Can also handle circular dependency down the road.



# Circular Dependency

- Just try to relax them... really
- But just for the sake of example:
  - iommufd depends on dmabuf (e.g. export BAR)
  - dmabuf depends on vfio-pci (exports BAR)
  - vfio-pci depends on iommufd
  - vfio => iommufd => dmabuf => vfio
- New Levels
  - DEVICE\_MEMORY: dmabuf
  - DEVICE\_DMA: vfio
- File handler can preserve FDs at multiple levels
  - Can be an extension that is added later



LINUX

PLUMBERS CONFERENCE

PRAGUE, CZECHIA / OCT. 5-7, 2026

