Hardware Resource Control in L4 µ-kernels

François Goichon, Guillaume Salagnac, Stéphane Frénot
University of Lyon, INRIA

SEmba 2011

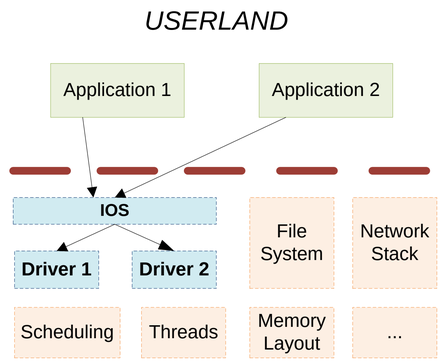
Motivation

In most operating systems, userland processes have unrestricted access to hardware drivers, by system calls in monolithic kernels or IPCs in µ-kernels such as L4. This unrestricted access can often allow **malicious software to force a denial of service on the driver** or strongly impact its quality of service.

To mitigate this safety issue without impacting much drivers code, **our approach is to extend L4 IPCs by adding a control layer to IPCs aimed at drivers.**
This would allow admission control to the driver, as well as **accounting and managing the driver's occupation by user processes**.

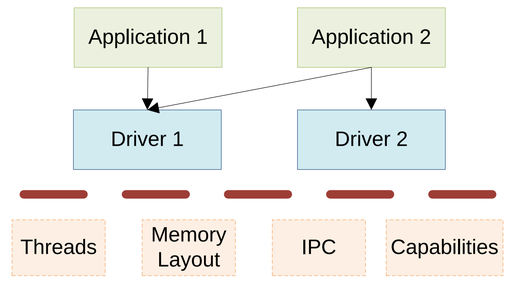
Context: Operating System Kernels

Monolithic Kernels



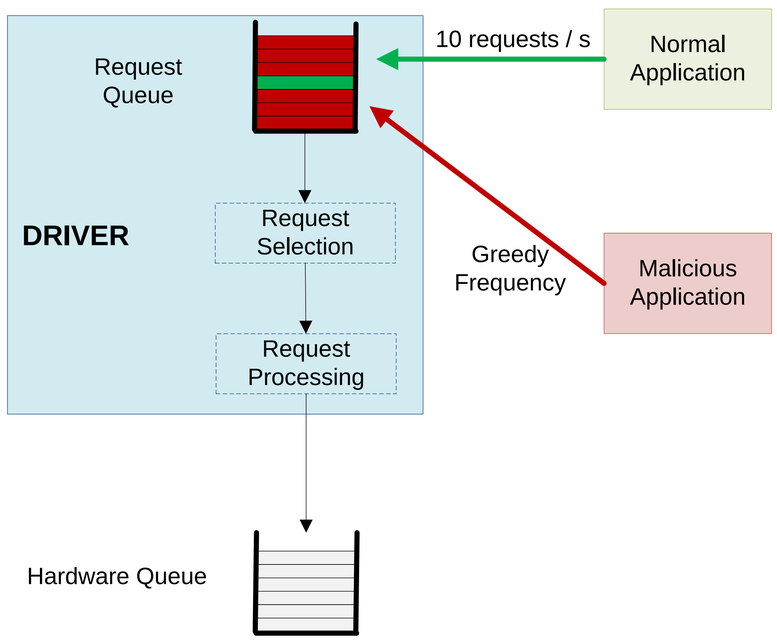
- All privileged code in kernel
- Communication via method calls
- Unified drivers interface (IOS)

L4 µ-kernel

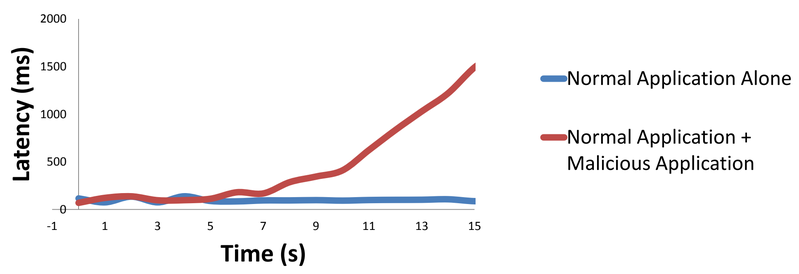


- Minimal kernel
- Communication via synchronous IPCs
- Userland privileges managed by capabilities

An Example of Resource Monopolization

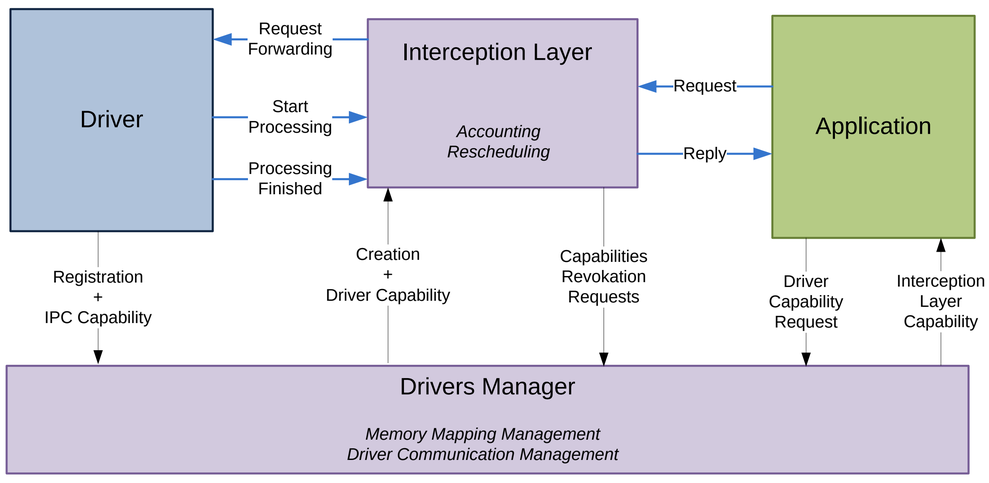


Delay between application deadlines and actual processing



➔ The quality of service provided by the driver to the normal application is severely impacted

Proposition: Extend L4 IPCs



- **Interception Layer:** Accounts actual hardware usage and reschedules requests if necessary

- **Drivers Manager:** Maintains drivers list, physical memory mappings and IPC capabilities

Expected Benefits

- **Safety:** Prevent malicious threads from monopolizing drivers

- **QoS Management:** Accounting and admission control would allow resource reservation and real-time guarantees

Open Questions

- How about resource control in higher layers?

- Which uniform resource reservation model?

- Robustness of the userland interception server to malicious users?