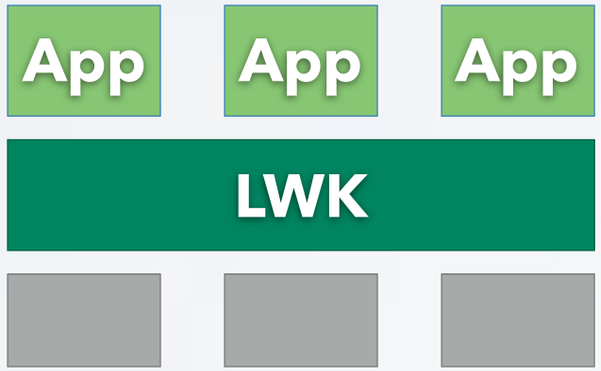
TECHNISCHE UNIVERSITÄT DRESDEN

TAILORING THE LINUX KERNEL AND MPI TO A FAST AND NOISE-FREE MICROKERNEL

CARSTEN WEINHOLD, TU DRESDEN

ADAM LACKORZYNSKI, JULIAN STECKLINA, MARKUS PARTHEYMÜLLER,
JAN BIERBAUM, MICHAEL JAHN, HERMANN HÄRTIG, STAFF@ZIH, STAFF@JSC

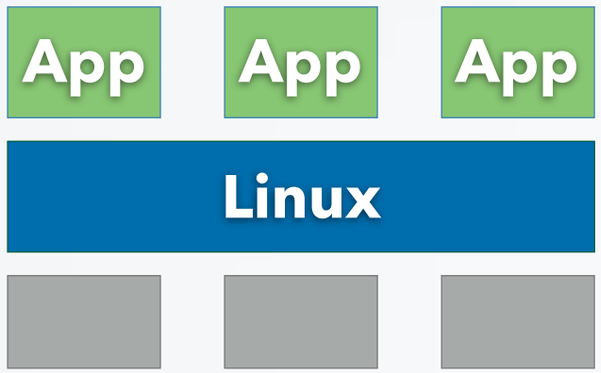
HPC OS ARCHITECTURE



LWK

⊕ No Noise
⊖ Compatibility
⊖ Features

CNK



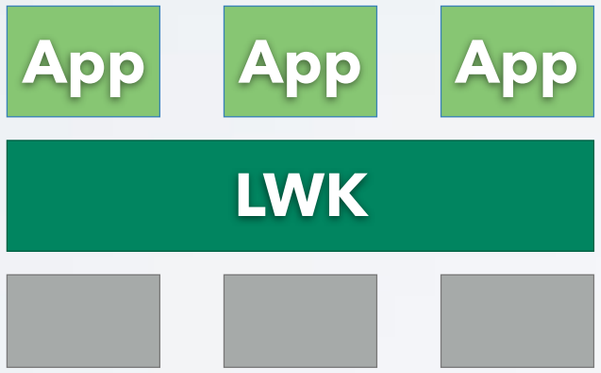
Linux (tweaked)

⊙ Low Noise
⊕ Compatibility
⊕ Features
⊖ Fast moving target

Cray

Argo

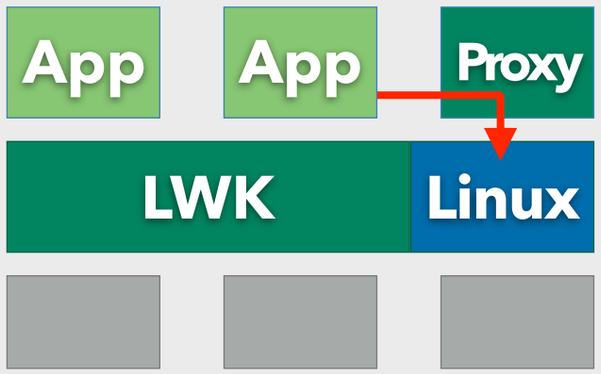
HPC OS ARCHITECTURE



LWK

⊕ No Noise
⊖ Compatibility
⊖ Features

CNK



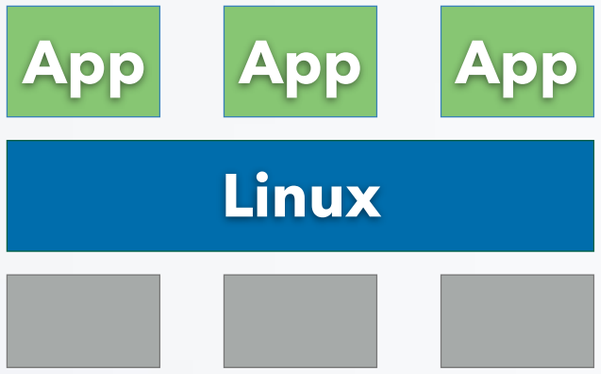
LWK + Linux

⊕ No Noise
⊕ Compatibility
⊕ Features
⊖ **Effort**

mOS

McKernel

Hobbes/Kitten



Linux (tweaked)

⊙ Low Noise
⊕ Compatibility
⊕ Features
⊖ Fast moving target

Cray

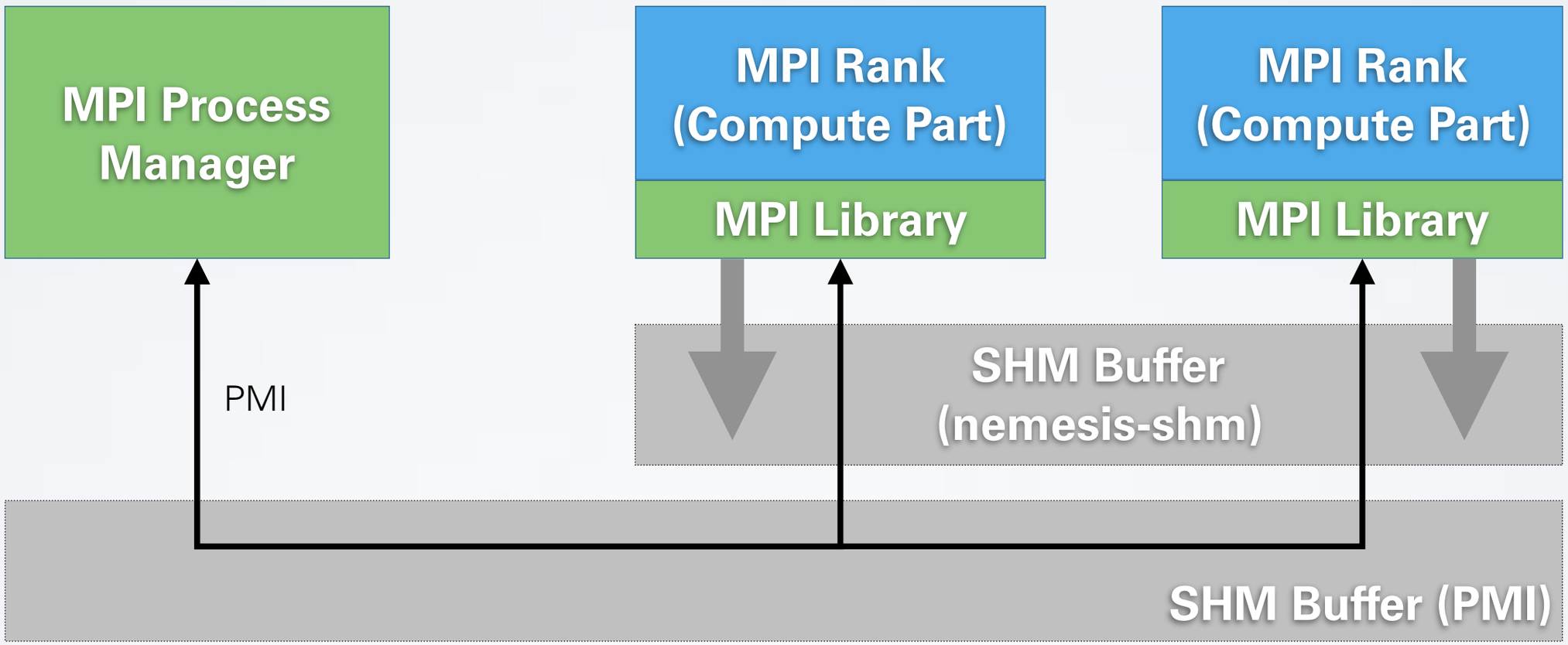
Argo

TECHNISCHE UNIVERSITÄT DRESDEN

VERSION 0

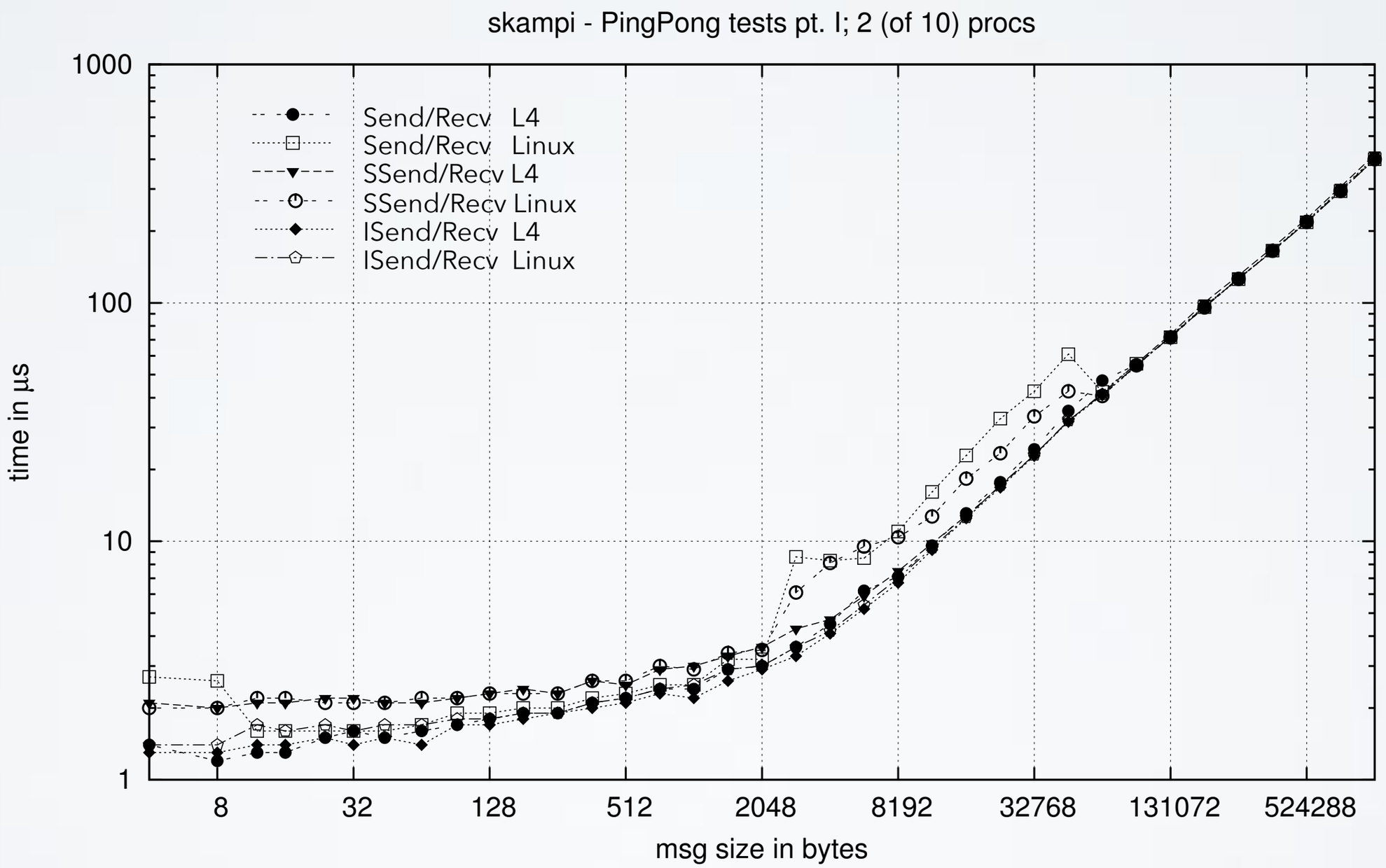
Pure L4

MPICH ON L4

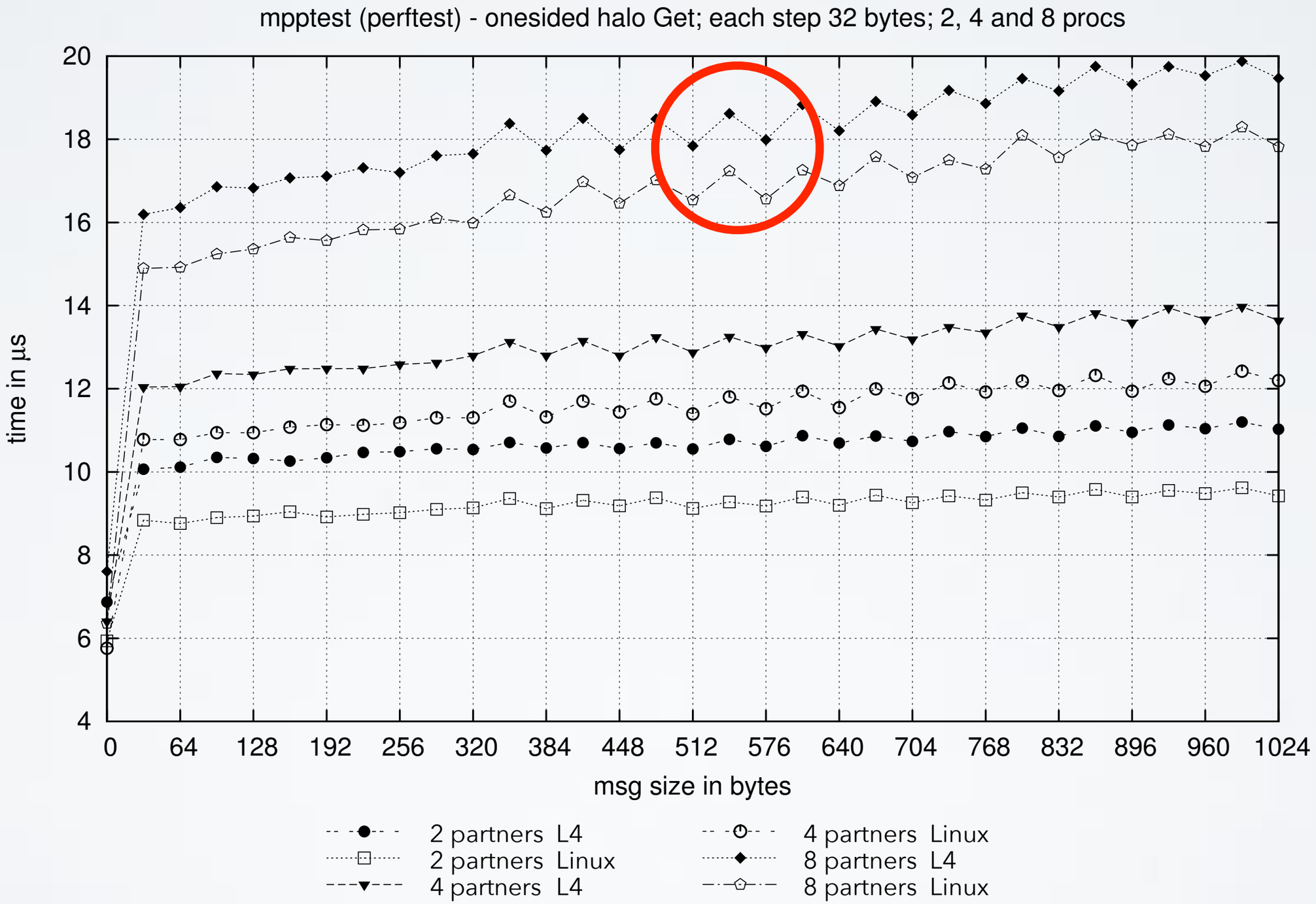

MPI Process Manager
PMI
MPI Rank (Compute Part)
MPI Library
MPI Rank (Compute Part)
MPI Library
SHM Buffer (nemesis-shm)
SHM Buffer (PMI)
L4 Microkernel / Hypervisor

TECHNISCHE UNIVERSITÄT DRESDEN

MPICH ON L4: PERFORMANCE



MPICH ON L4: PERFORMANCE



MPICH ON L4

```
MPICH-L4| 0>--- p1MD version  "0815" ---
MPICH-L4| MPI_Wtime
MPICH-L4| MPI_Wtime
MPICH-L4| 0>    [compiled:  "2013-08-12" ]
MPICH-L4| 0>
MPICH-L4| 0><i> Simulation started :: 01.01.1970  00:00:03.993  (UTC  +0)
MPICH-L4| 0>
MPICH-L4| 0>
MPICH-L4| 0>Command: rom/p1md rom/md_10k_nochkpt.nml
MPICH-L4| 0>
MPICH-L4| 0>
MPICH-L4| 0><i>   4 processes

                        <more output here>

MPICH-L4| 0><cpu> wall-clock timer function:  MPI_Wtime
MPICH-L4| 0>     Work count sum:     184102828
MPICH-L4| 1>
MPICH-L4| 2>
MPICH-L4| 3><cpu>        MD step performance:        2.936 steps/s
MPICH-L4| 0>
MPICH-L4| 1><cpu>        MD step performance:        2.938 steps/s
MPICH-L4| 2><cpu>        MD step performance:        2.939 steps/s
MPICH-L4| 3><cpu> MD interaction performance:        0.270 Mega interactions/s
MPICH-L4| 0><cpu>        MD step performance:        2.945 steps/s
MPICH-L4| 1><cpu> MD interaction performance:        0.271 Mega interactions/s
MPICH-L4| 2><cpu> MD interaction performance:        0.271 Mega interactions/s
MPICH-L4| 0><cpu> MD interaction performance:        0.271 Mega interactions/s
MPICH-L4| 0>
MPICH-L4| 0>
MPICH-L4| 0><i> Simulation finished :: 01.01.1970  00:12:58.468  (UTC  +0)
MPICH-L4| 0>
```

VERSION 0: MPICH ON L4

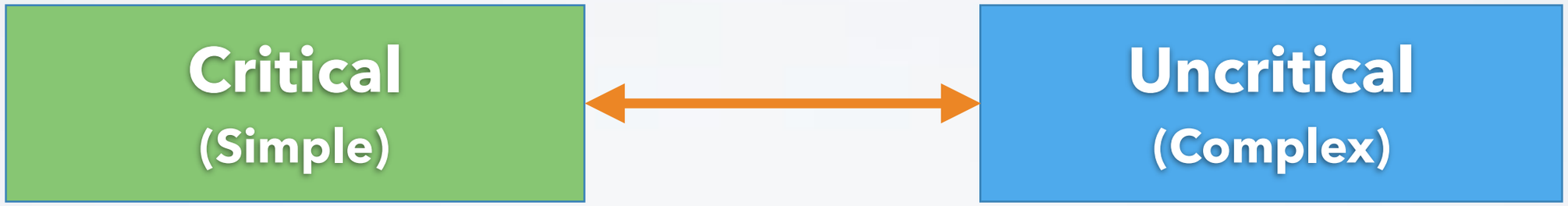
- It worked:
  - p1MD, MPICH test, perftest, …
  - HPL, CP2K, …
- Performance comparable to Linux(*)
- Major limitation: no network backend

TECHNISCHE UNIVERSITÄT DRESDEN

VERSION 1

Hybrid L4 + Linux

HYBRID SYSTEM

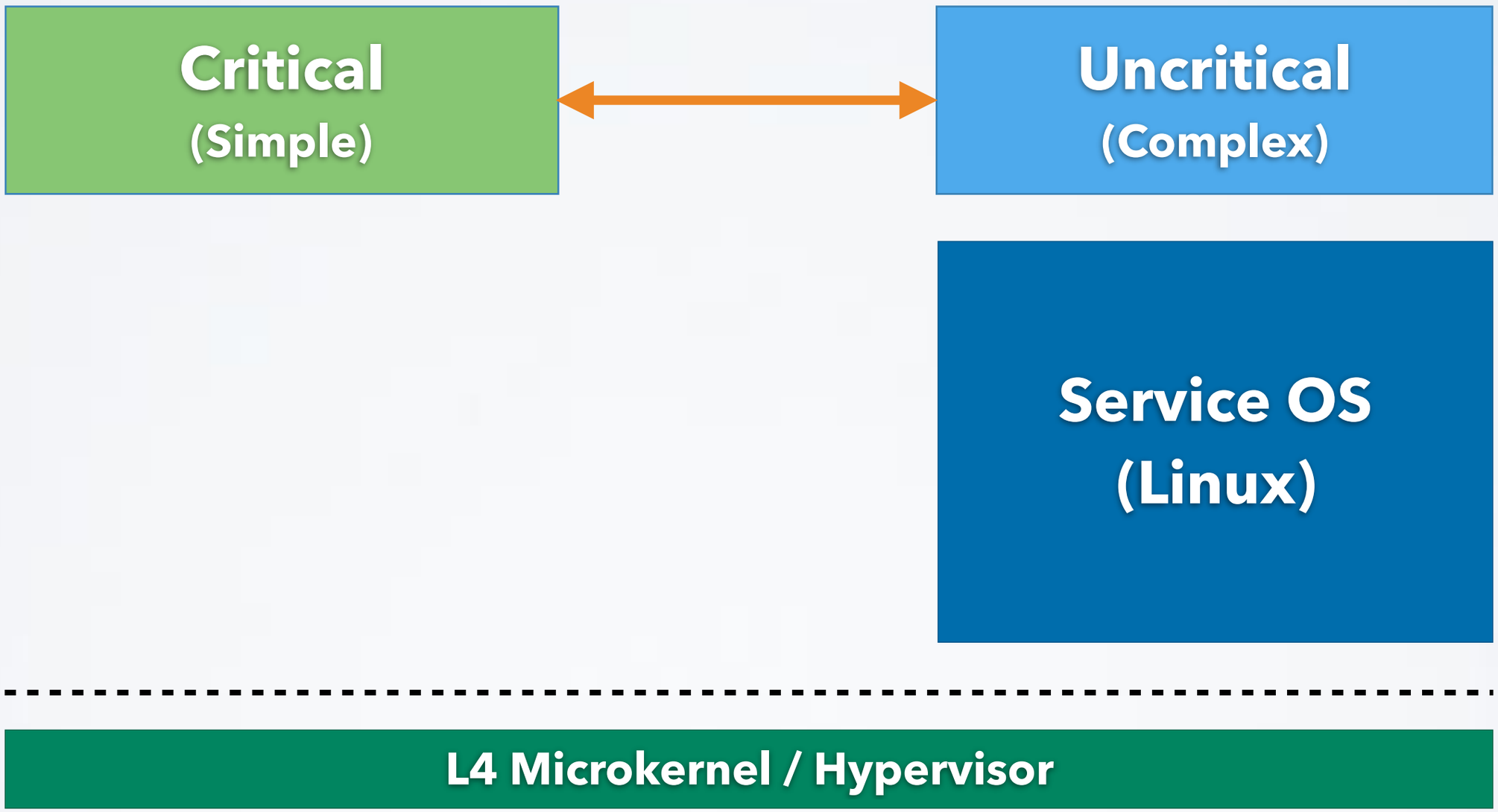


Real-time

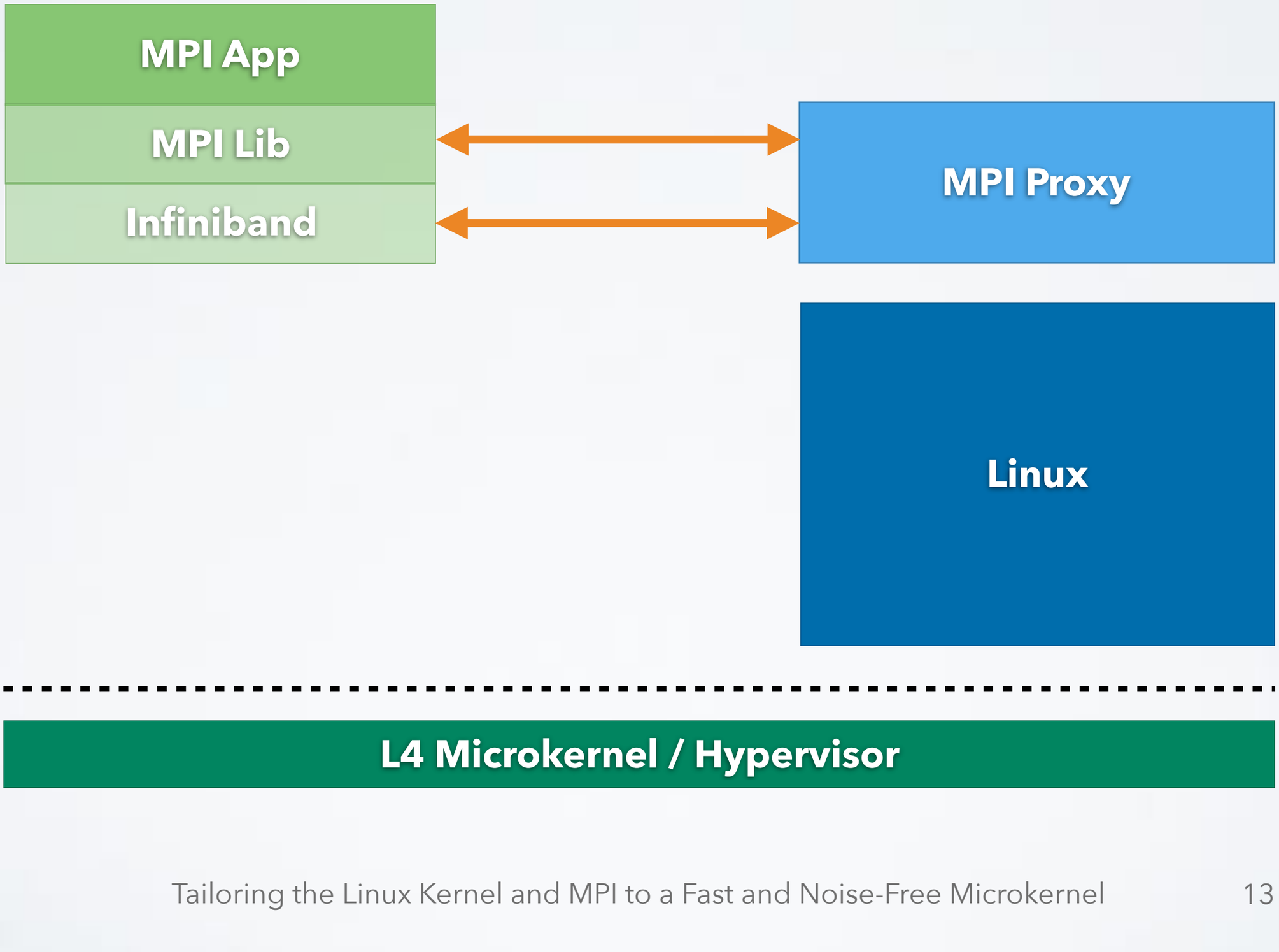
Security

Resilience

HYBRID SYSTEM



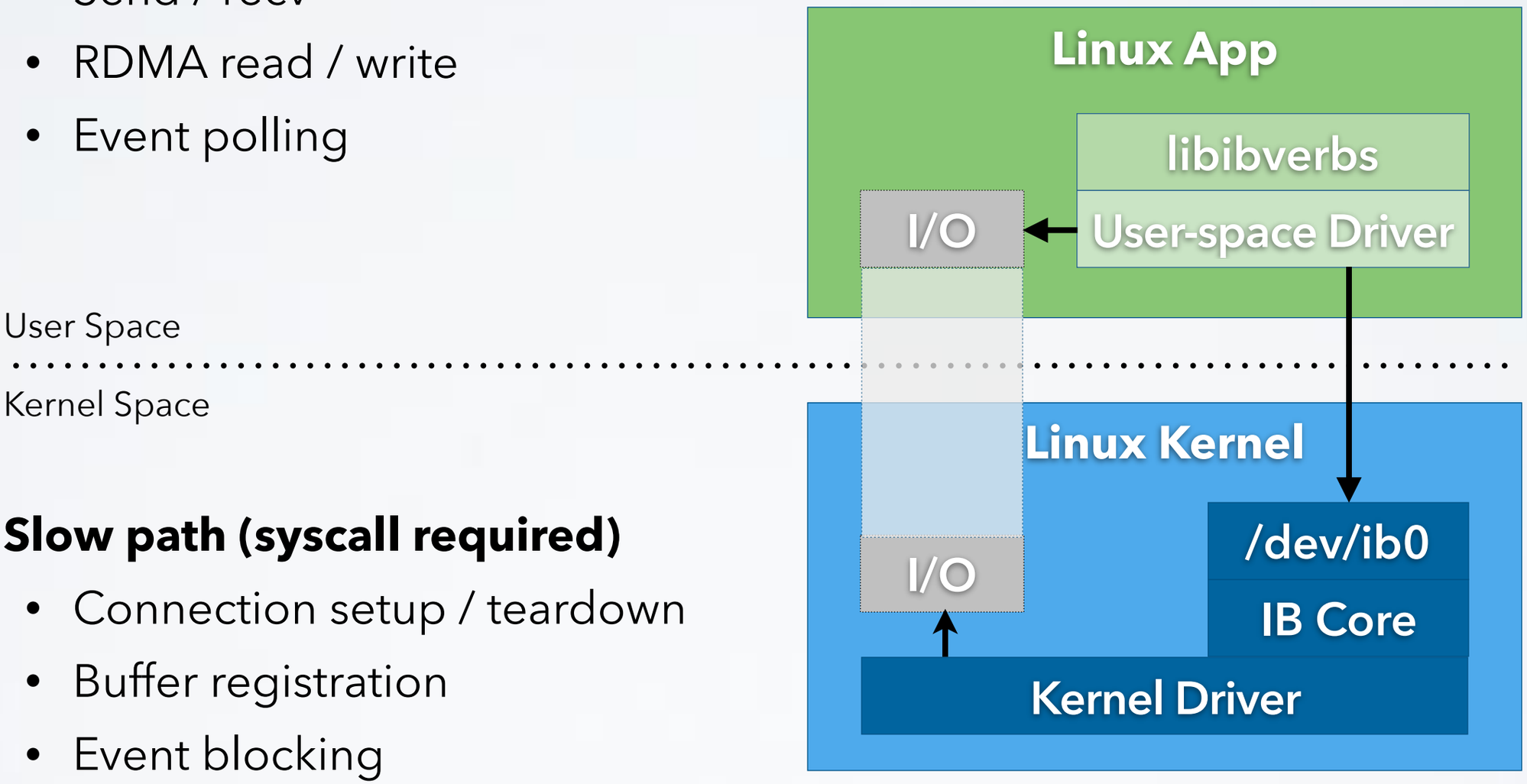
HYBRID SYSTEM



INFINIBAND DRIVER

Fast path (direct access to MMIO regs)

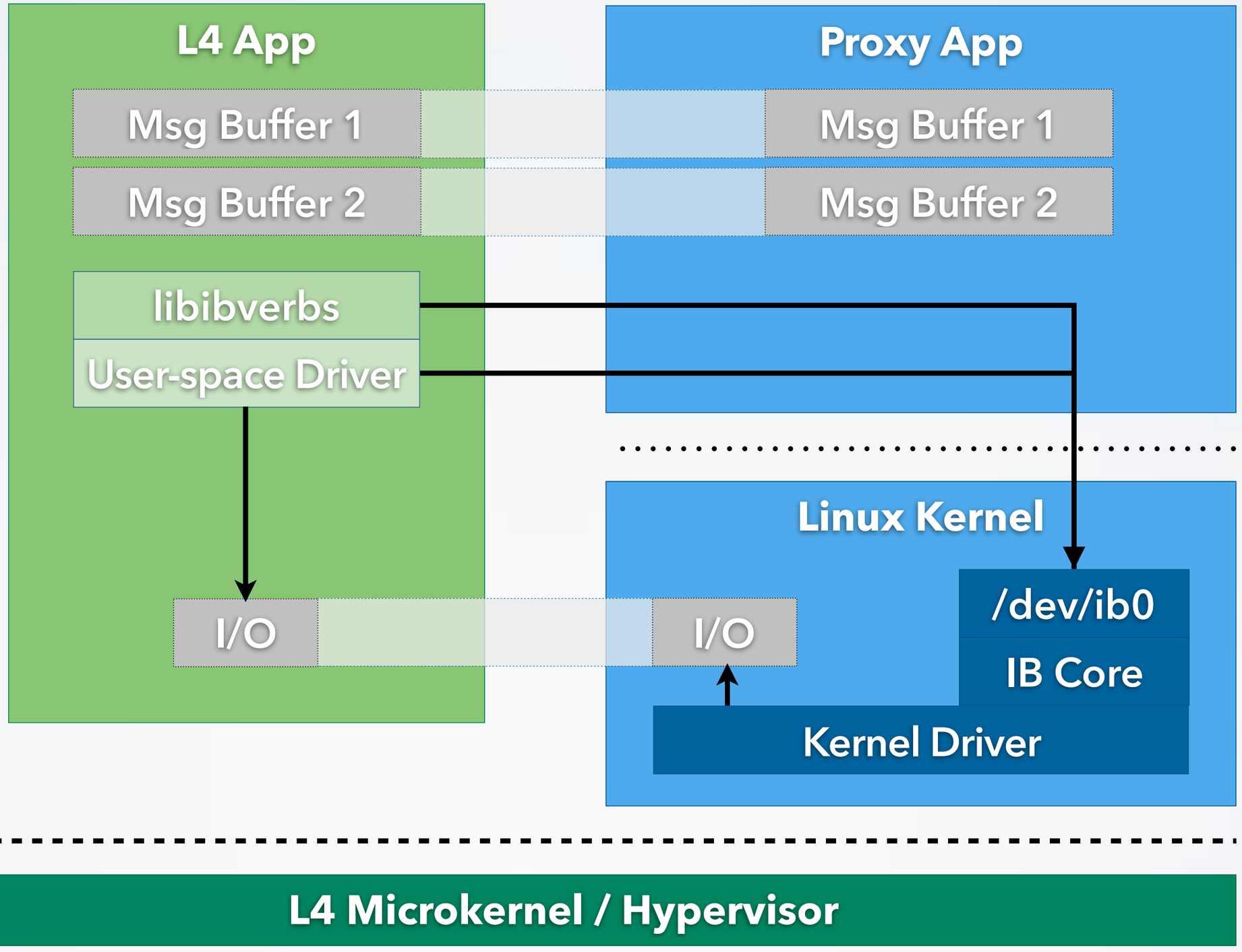
- Send / recv
- RDMA read / write
- Event polling



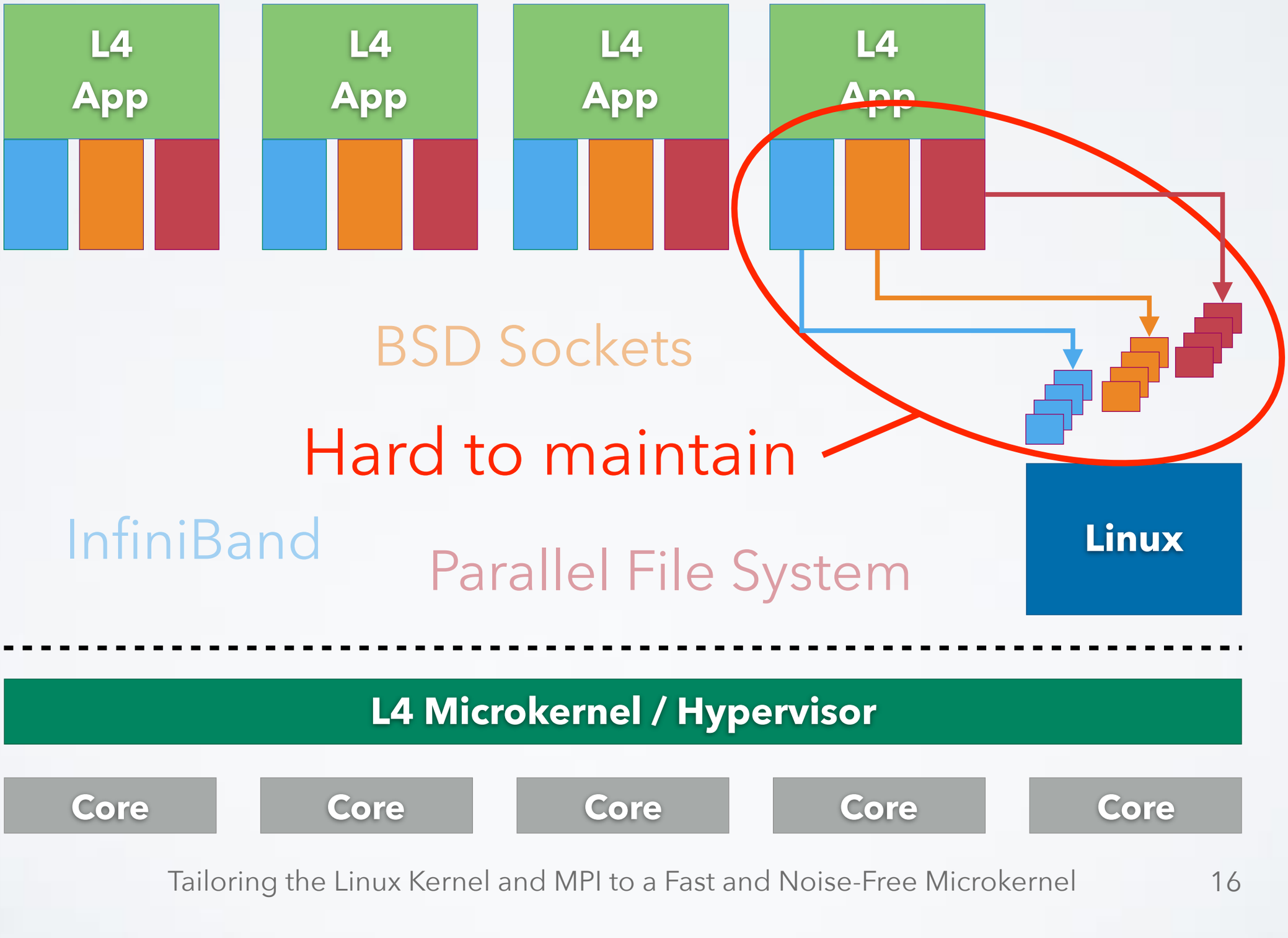
Slow path (syscall required)

- Connection setup / teardown
- Buffer registration
- Event blocking

INFINIBAND DRIVER



VERSION 1: SPLIT MVAPICH2



TECHNISCHE UNIVERSITÄT DRESDEN

VERSION 2

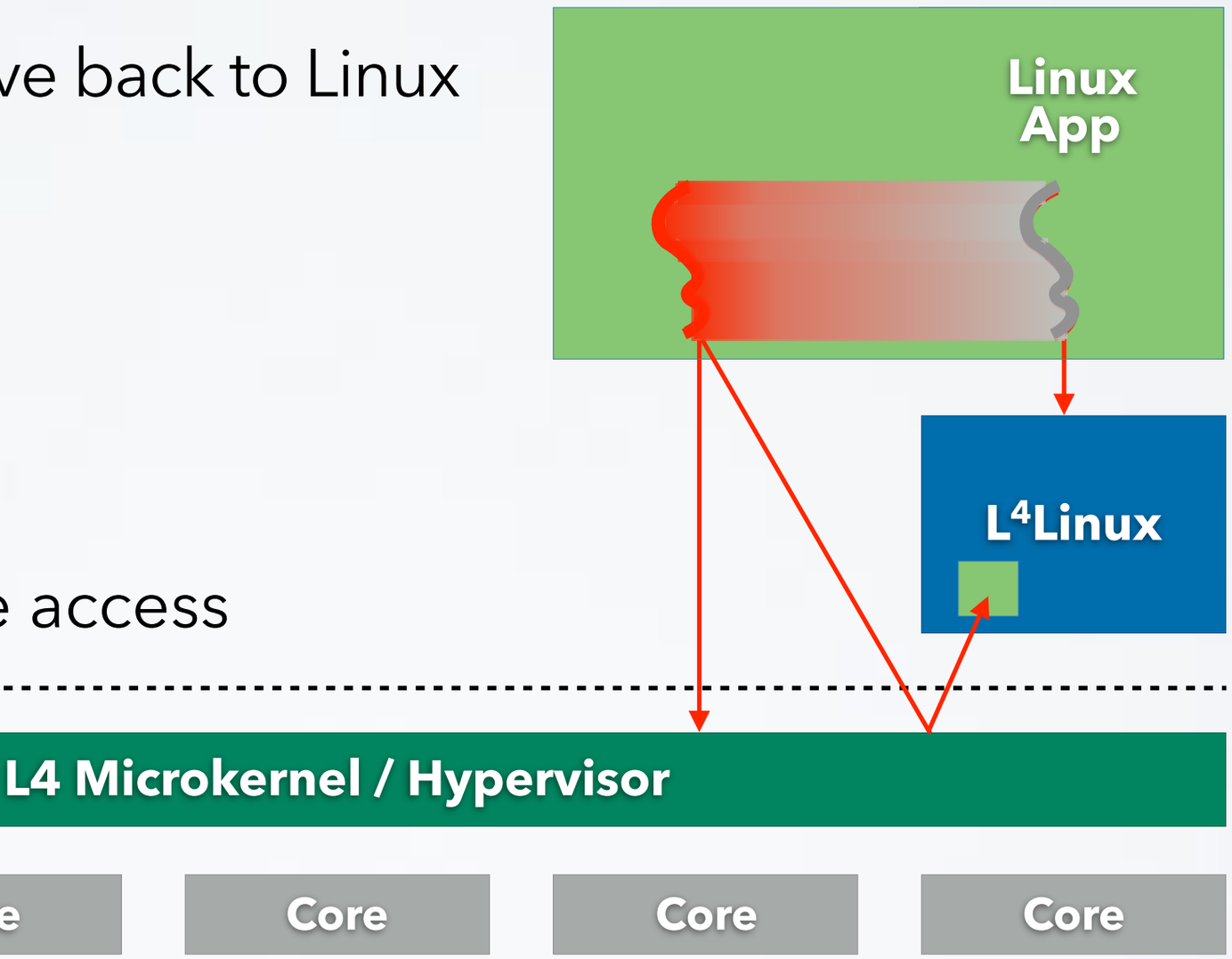
Decoupled Threads

L4 + L⁴LINUX

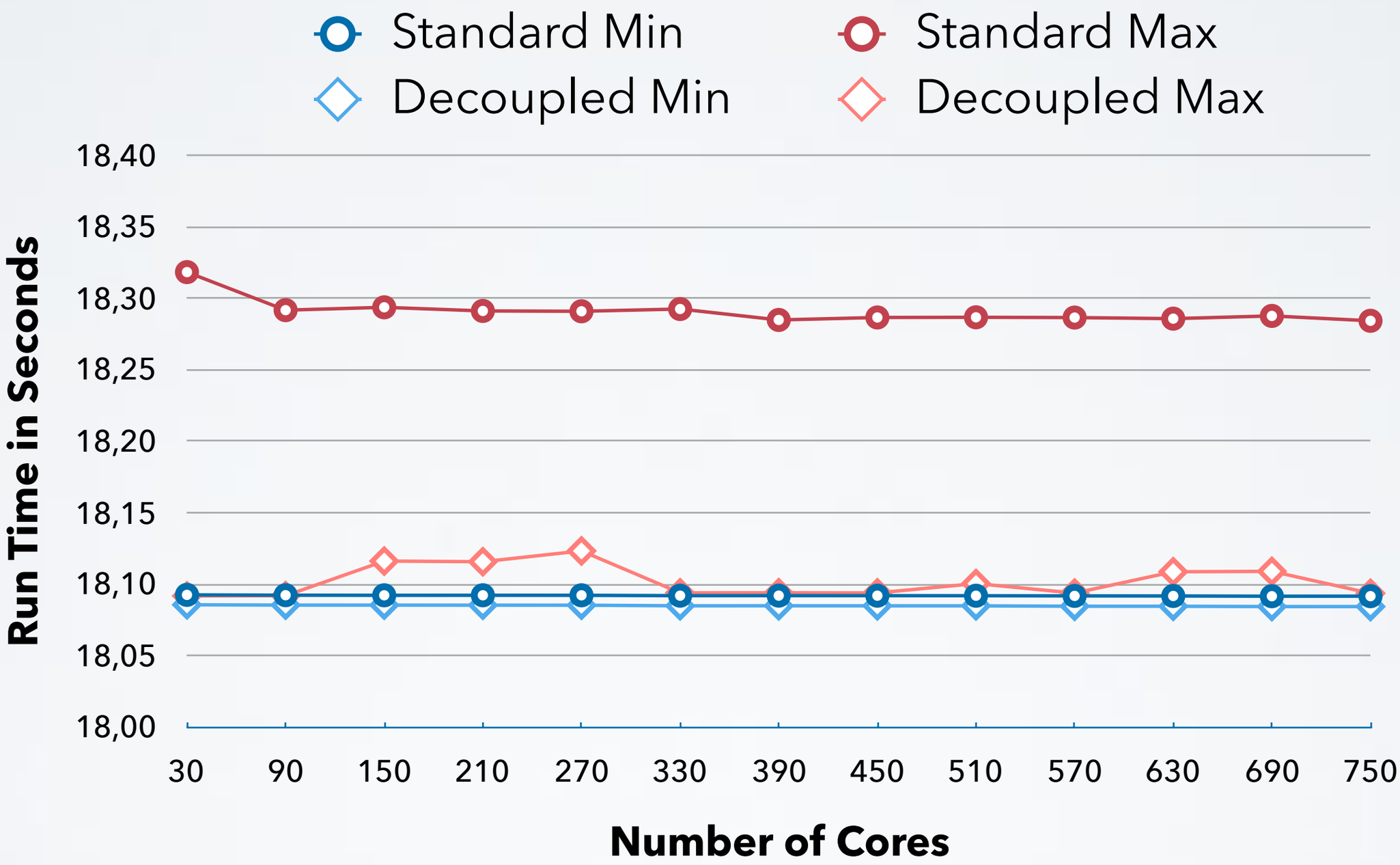
- **Unmodified** Linux programs (MPI, …)
- **L⁴Linux** on L4 microkernel
- **L4 microkernel** controls the node
- **Light-weight** and **low-noise** if needed
- **Linux process** = **L4 address space** + **thread**
- Linux syscalls / exceptions:
  generic forwarding to L⁴Linux kernel

DECOUPLED EXECUTION

- **Decoupling:** move Linux thread to new L4 thread on its own core
- **Linux syscall:** Move back to Linux
- **L4 syscalls:**
  - Scheduling
  - Threads
  - Memory
- **Direct I/O** device access

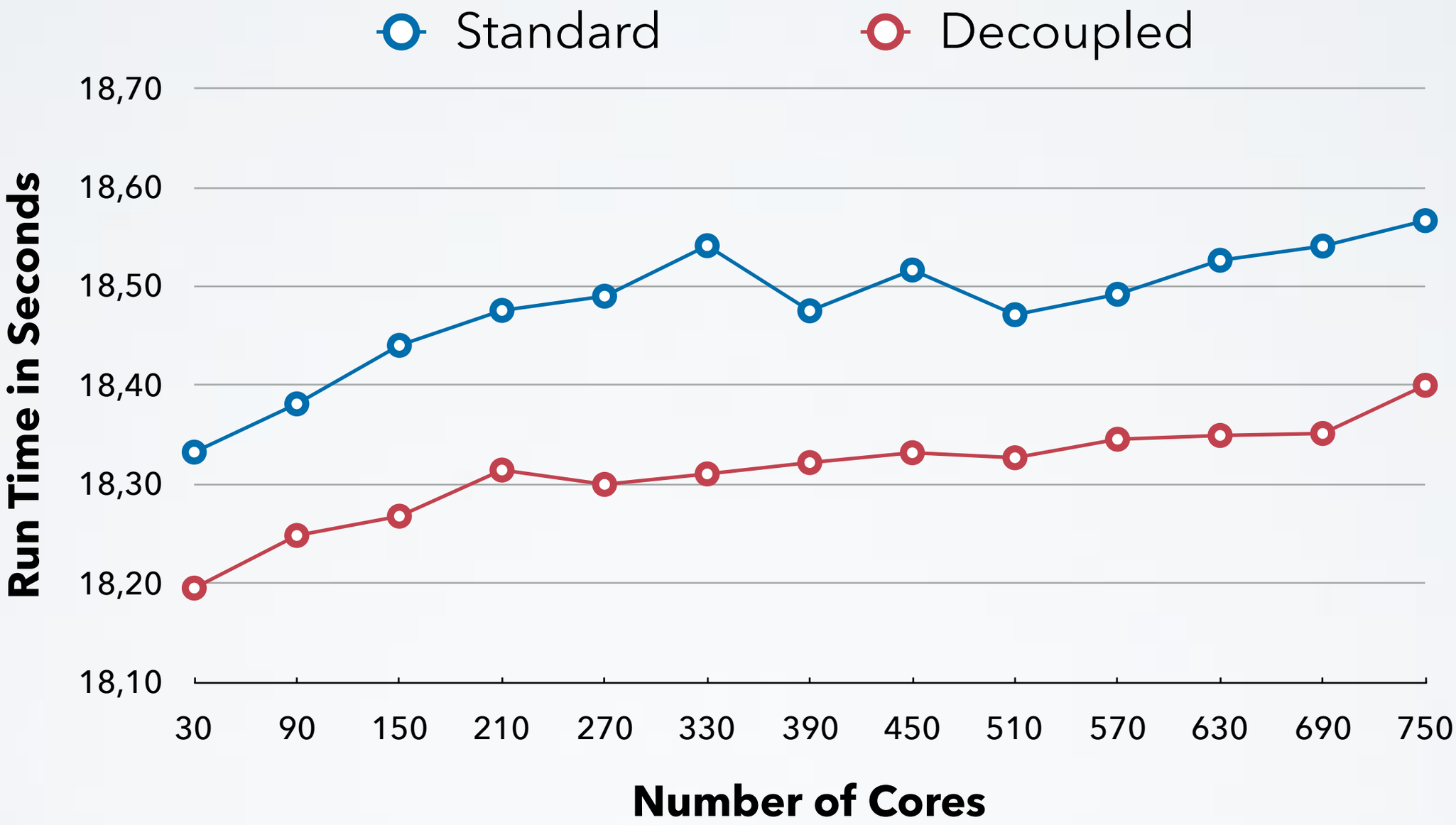


DECOUPLING: EP



Adam Lackorzynski, Carsten Weinhold, Hermann Härtig, "Decoupled: Low-Effort Noise-Free Execution on Commodity Systems", ROSS 2016, June 2016, Kyoto, Japan

DECOUPLING: BSP



Adam Lackorzynski, Carsten Weinhold, Hermann Härtig, "Decoupled: Low-Effort Noise-Free Execution on Commodity Systems", ROSS 2016, June 2016, Kyoto, Japan

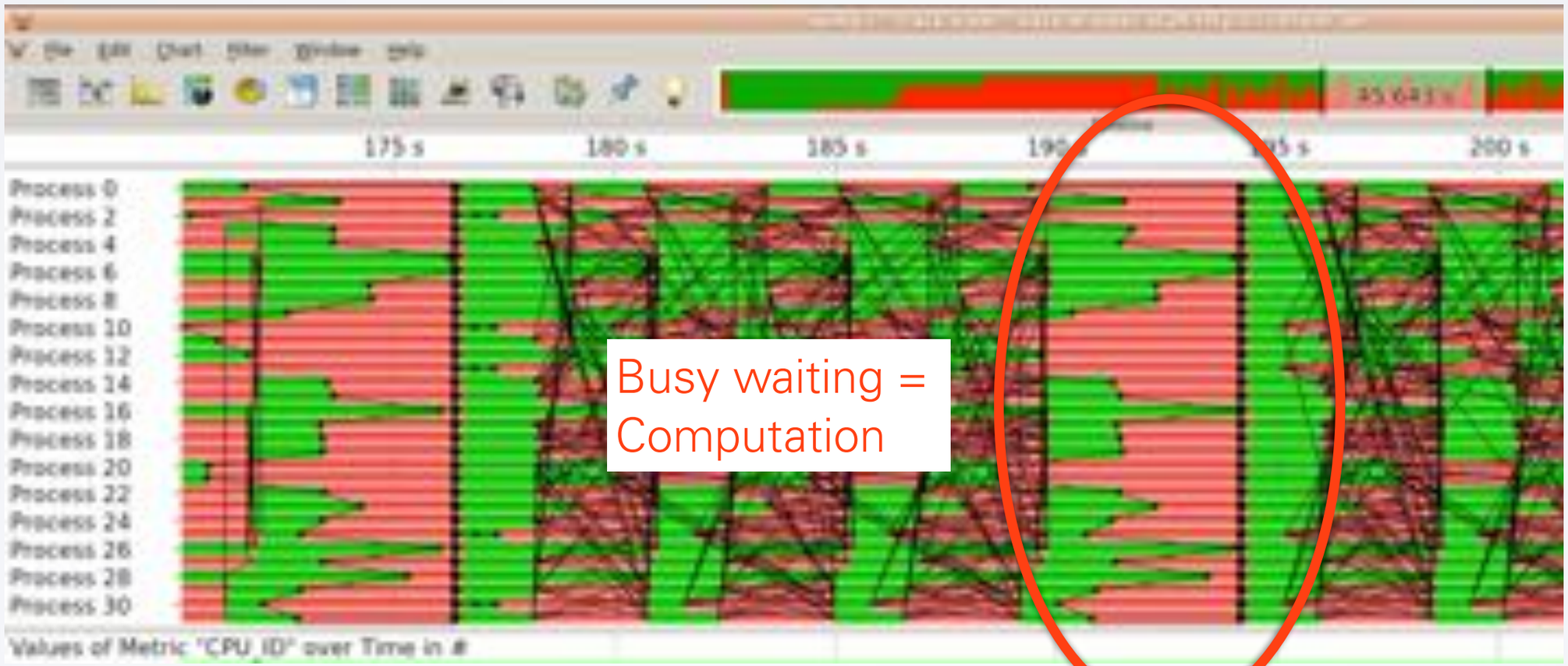
TECHNISCHE UNIVERSITÄT DRESDEN

VERSION 2.5

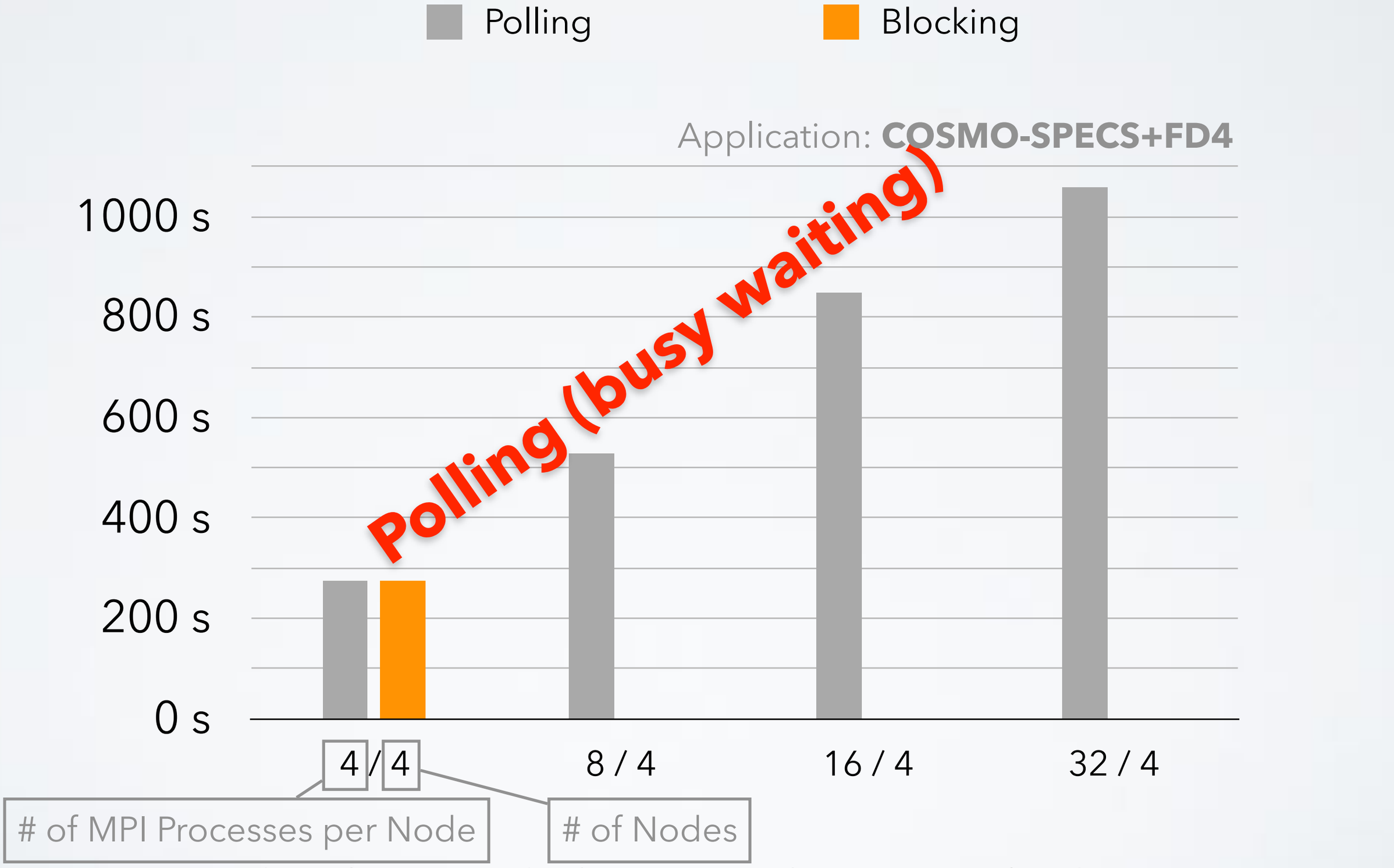
Decoupled Interrupts

WIP

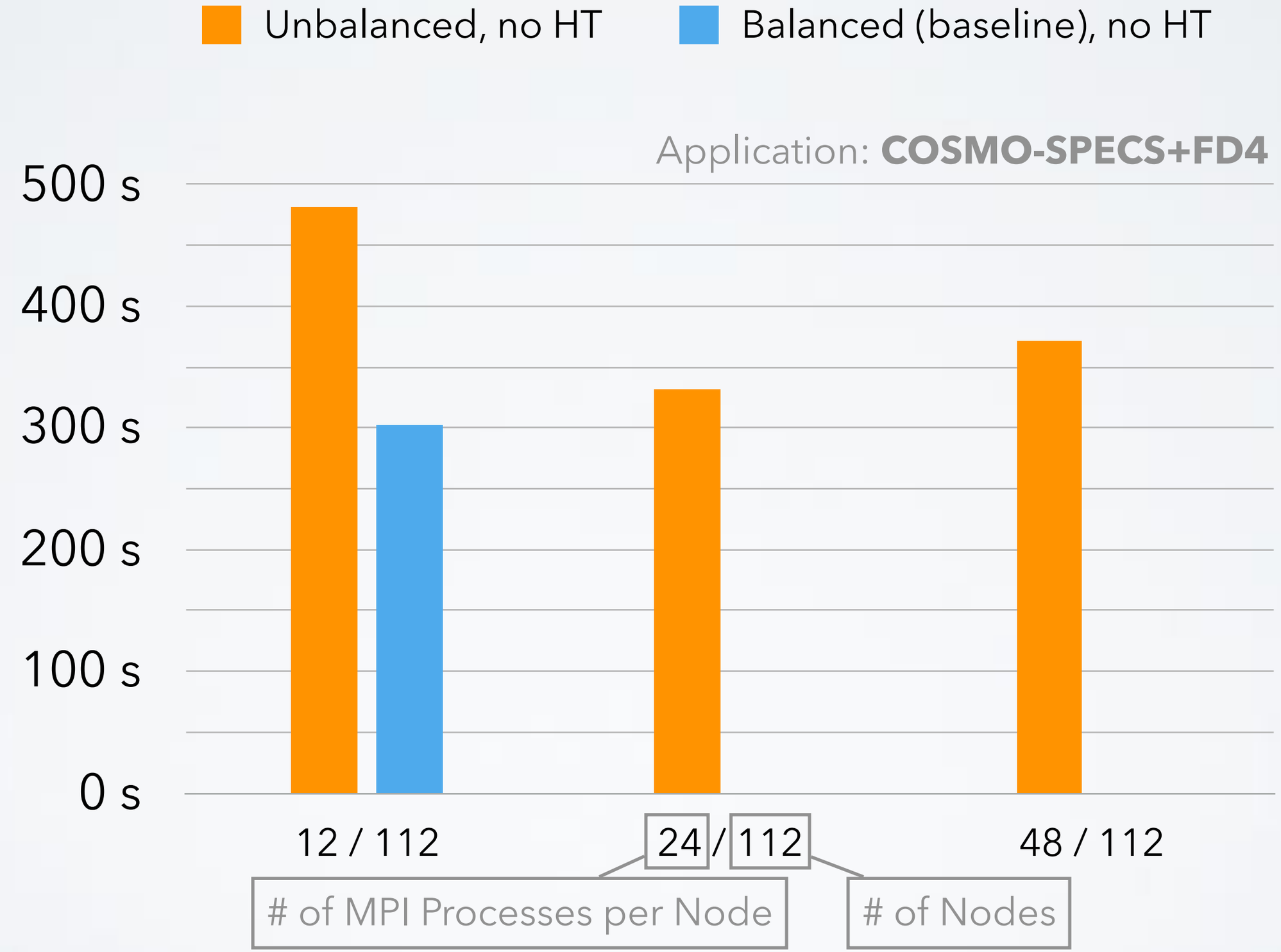
BUSY WAITING



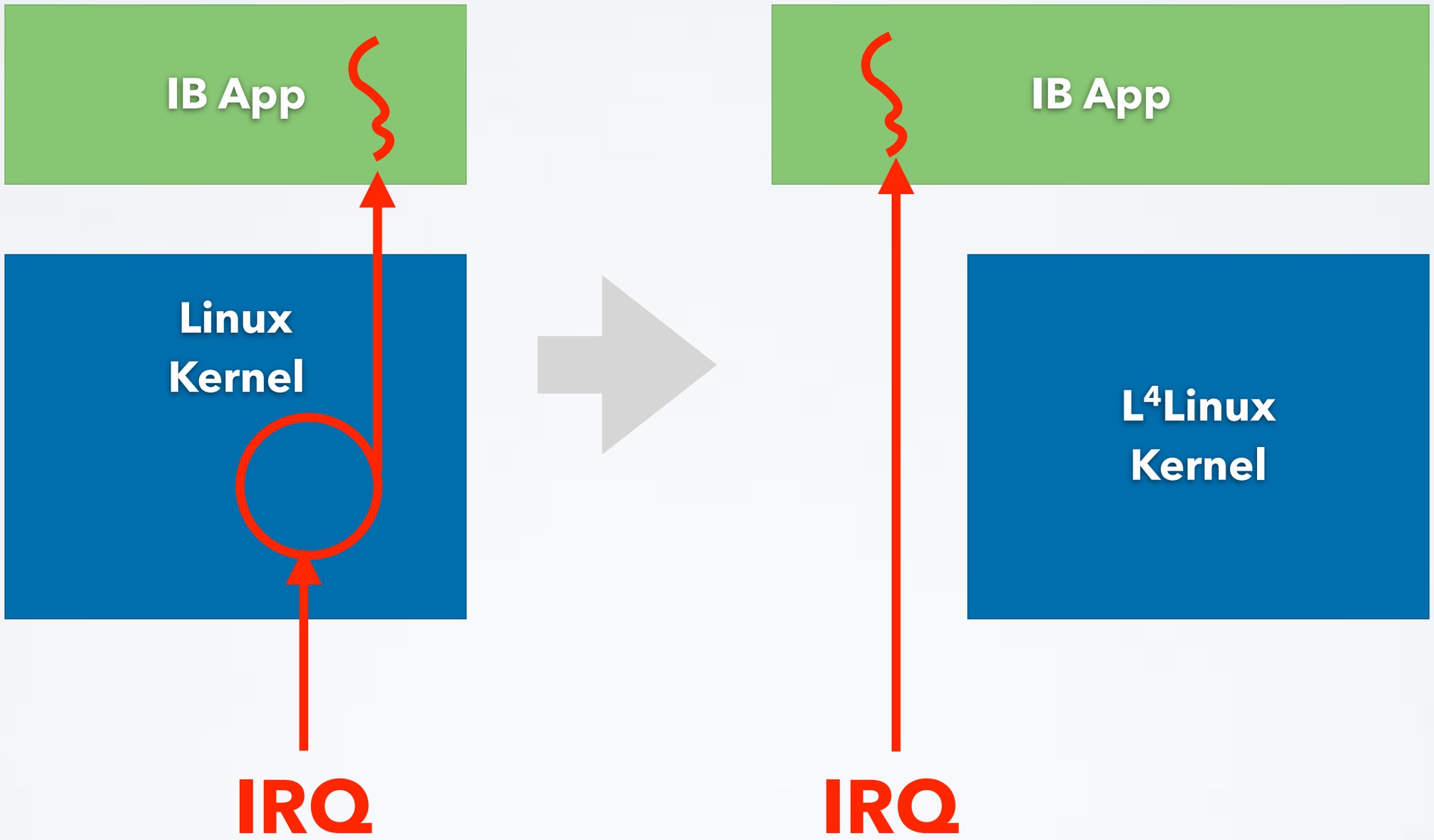
OVERSUBSCRIPTION



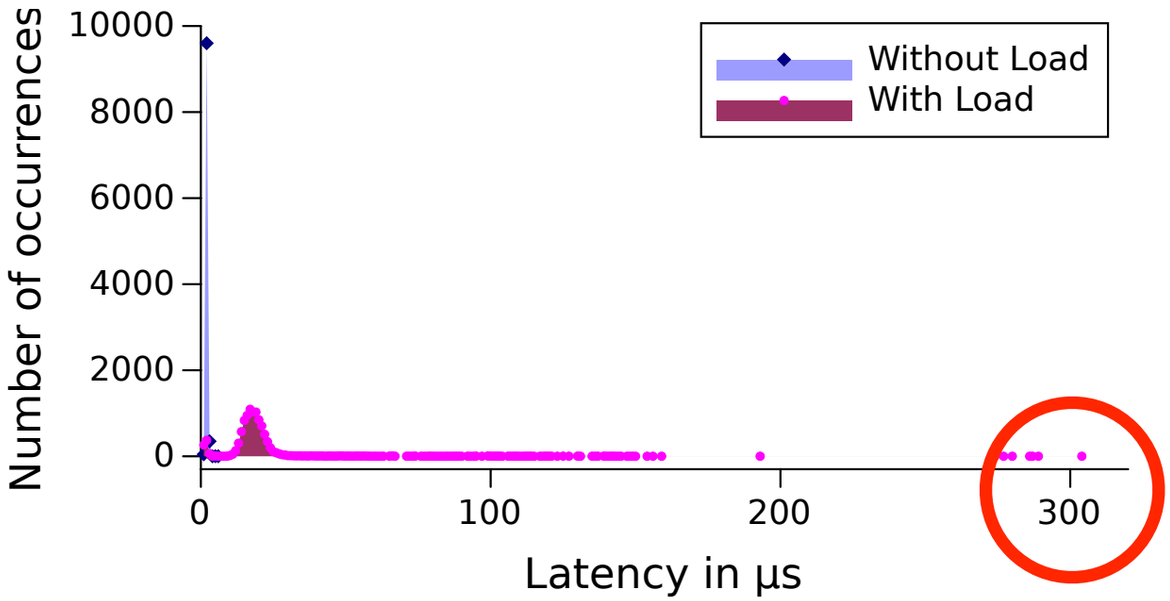
OVERSUBSCRIPTION



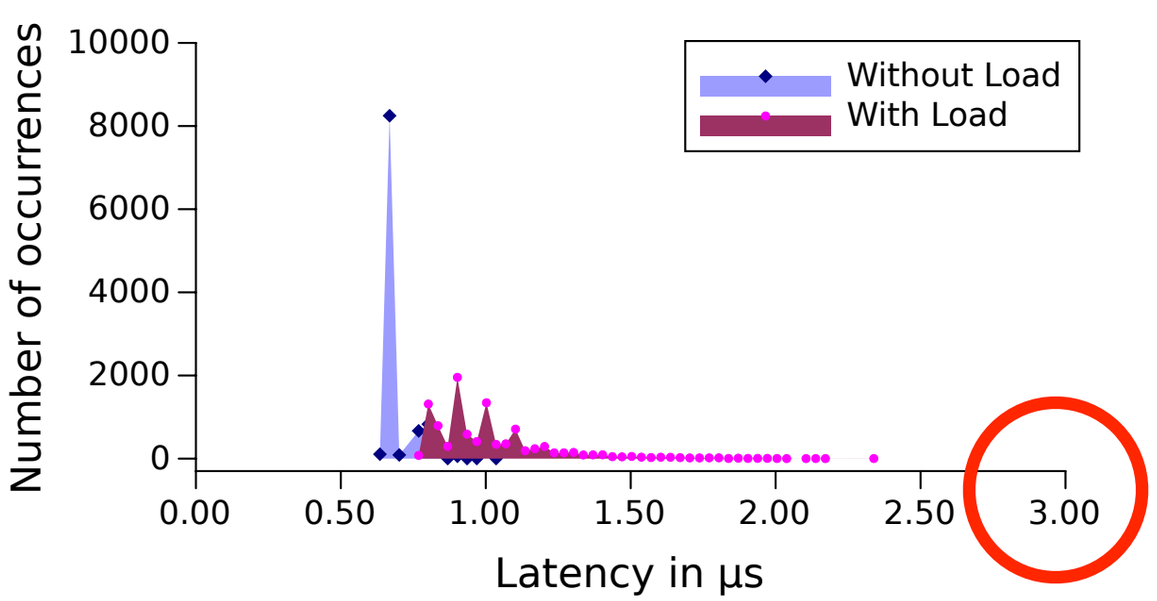
IRQ FASTPATH



WAKE FROM IRQ: LINUX VS L4



Linux



L4

Wake from interrupt on L4/Nova: 900 cycles, **0.3 µs**

(best case, on Intel Core i7-4770 CPU @ 3.40GHz)

Adam Lackorzynski, Carsten Weinhold, Hermann Härtig, „Predictable Low-Latency Interrupt Response with General-Purpose Systems“, OSPERT 2017, Duprovnik, Kroatia, June 2017

SUMMARY

- L4 Microkernel + L^{4}Linux + Decoupling
- Binary compatible + `sysfs` interface
- No modification to MVAPICH2
- Lessons learned:
  - Maximize reuse + minimize critical path
  - Methodology: Start from Linux, not from L4

Source: https://l4re.org

Project: https://ffmk.tudos.org

DFG Deutsche Forschungsgemeinschaft

SPPEXA German Priority Programme 1648 **Software for Exascale Computing**