barkhausen institut

Leibniz Universität Hannover

KERNKONZEPT

TECHNISCHE UNIVERSITÄT DRESDEN

MettEagle: Costs And Benefits Of Implementing Containers On Microkernels

Till Miemietz, Viktor Reusch, Matthias Hille, Lars Wrenger, Jana Eisoldt, Jan Klötzke, Max Kurze, Adam Lackorzynski, Michael Roitzsch, Hermann Härtig

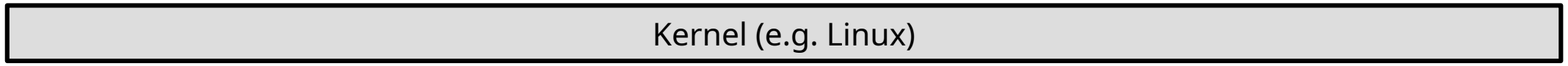
Containers From A Systems Perspective



- Lightweight virtualization of userspace instances on a shared kernel
  - Prominent use case: Shared systems hosting untrusted clients

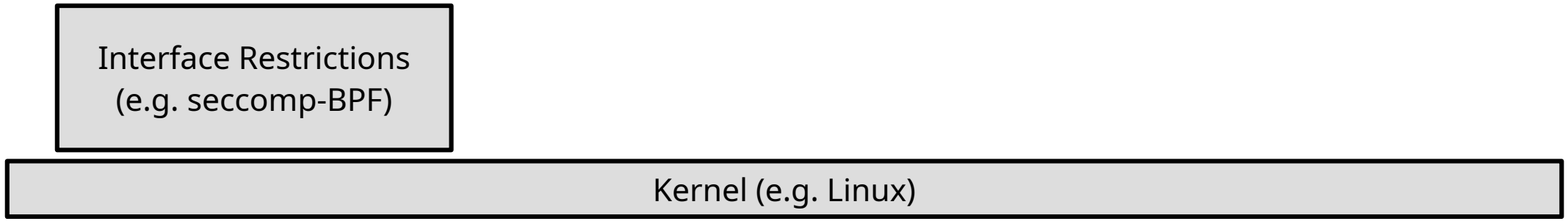
Containers From A Systems Perspective

- Lightweight virtualization of userspace instances on a shared kernel
  - Prominent use case: Shared systems hosting untrusted clients



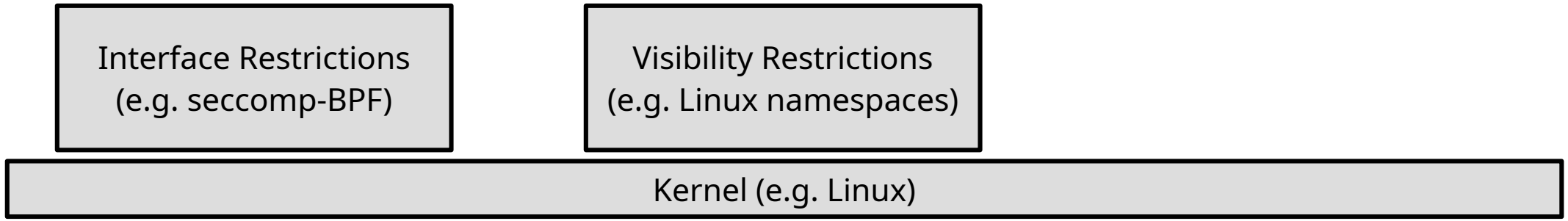
Containers From A Systems Perspective

- Lightweight virtualization of userspace instances on a shared kernel
  - Prominent use case: Shared systems hosting untrusted clients



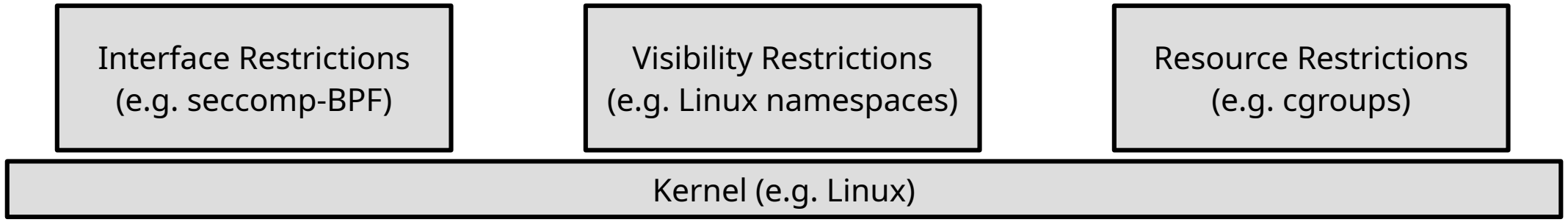
Containers From A Systems Perspective

- Lightweight virtualization of userspace instances on a shared kernel
  - Prominent use case: Shared systems hosting untrusted clients



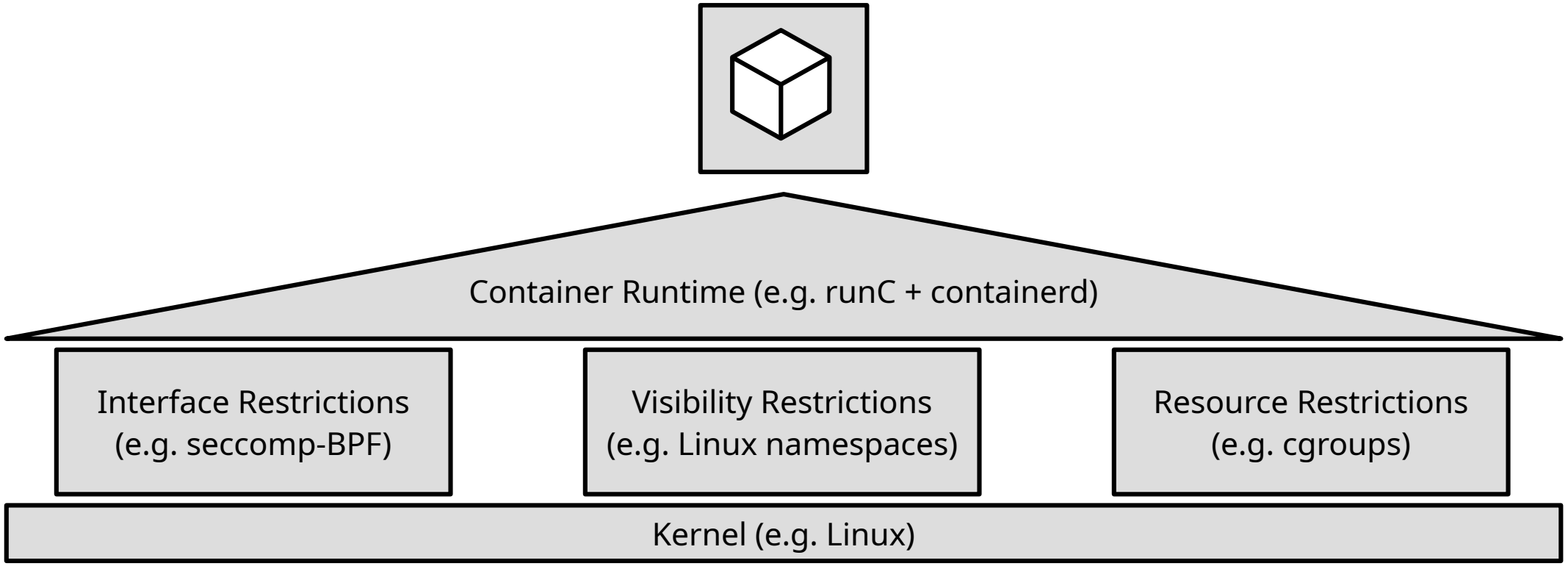
Containers From A Systems Perspective

- Lightweight virtualization of userspace instances on a shared kernel
  - Prominent use case: Shared systems hosting untrusted clients



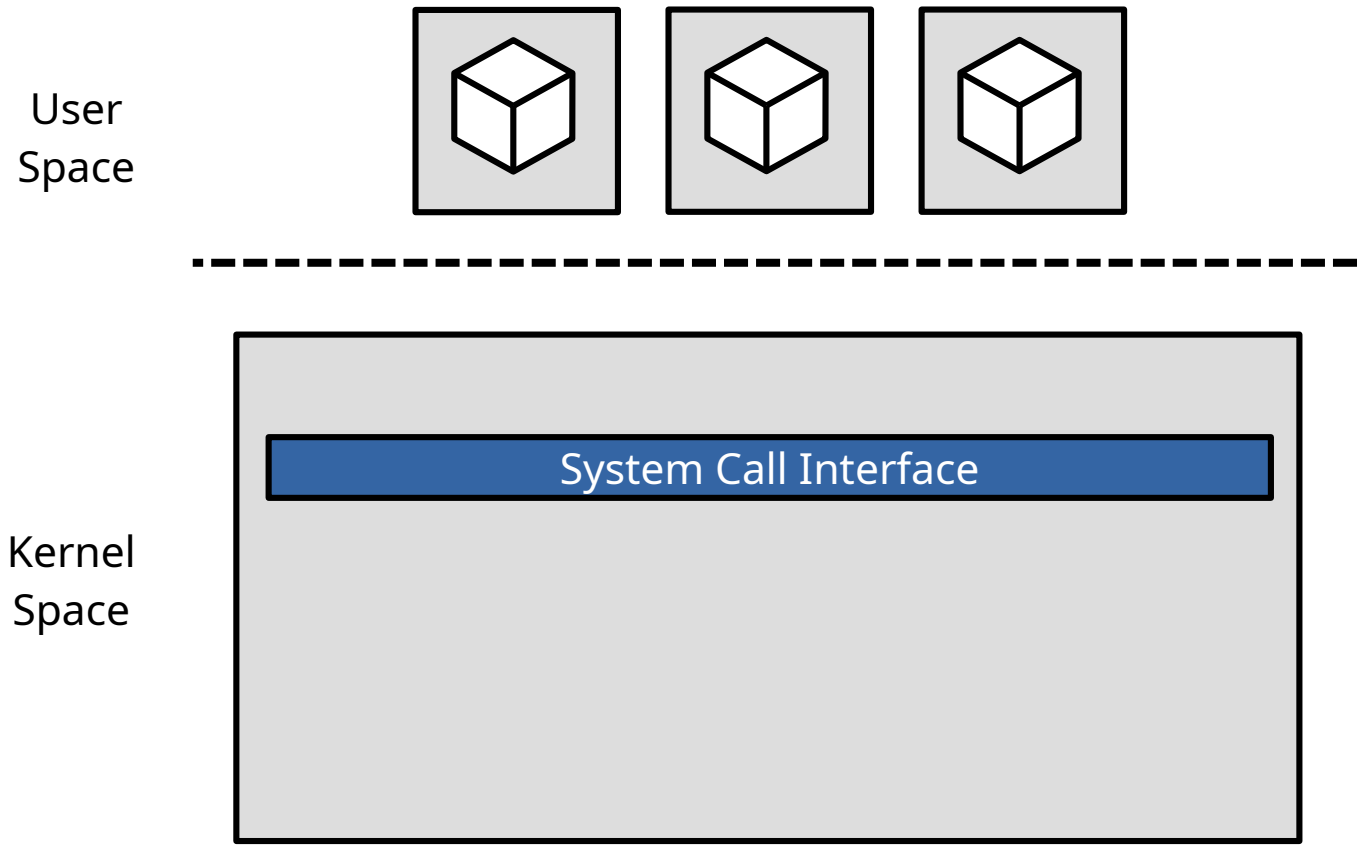
Containers From A Systems Perspective

- Lightweight virtualization of userspace instances on a shared kernel
  - Prominent use case: Shared systems hosting untrusted clients



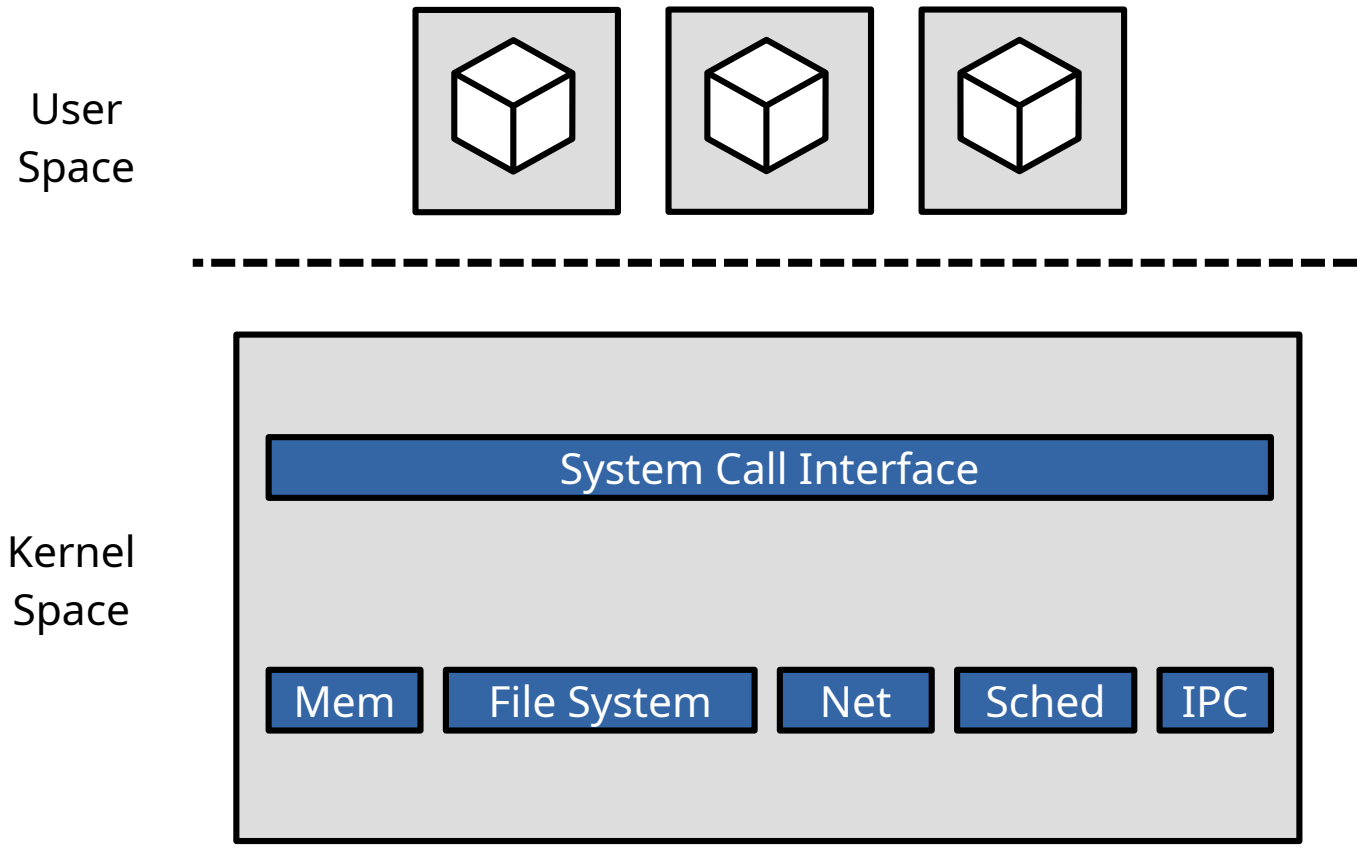
Containers On Monoliths – A Security Perspective

- Containers try to retroactively fix an OS design with weak isolation



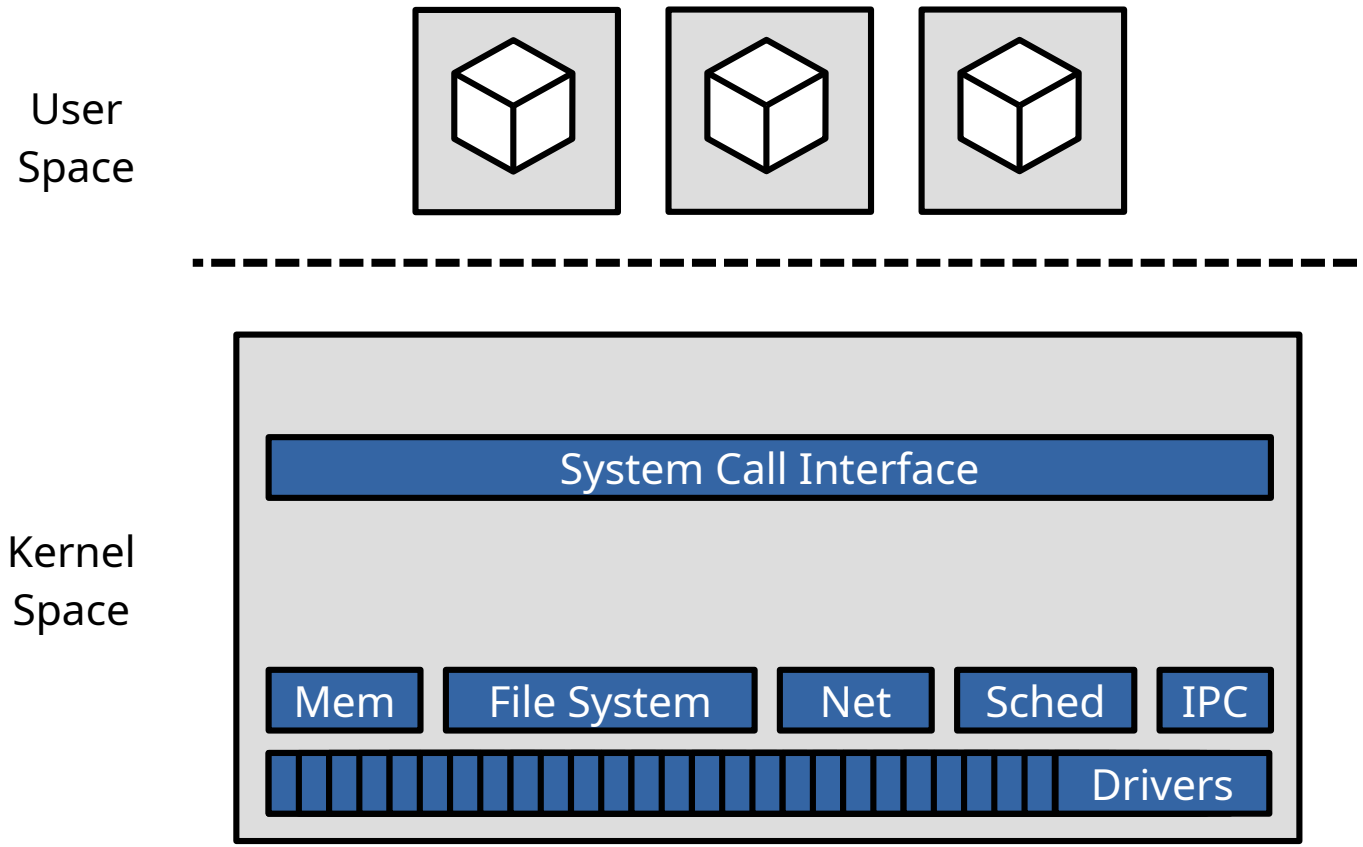
Containers On Monoliths – A Security Perspective

- Containers try to retroactively fix an OS design with weak isolation



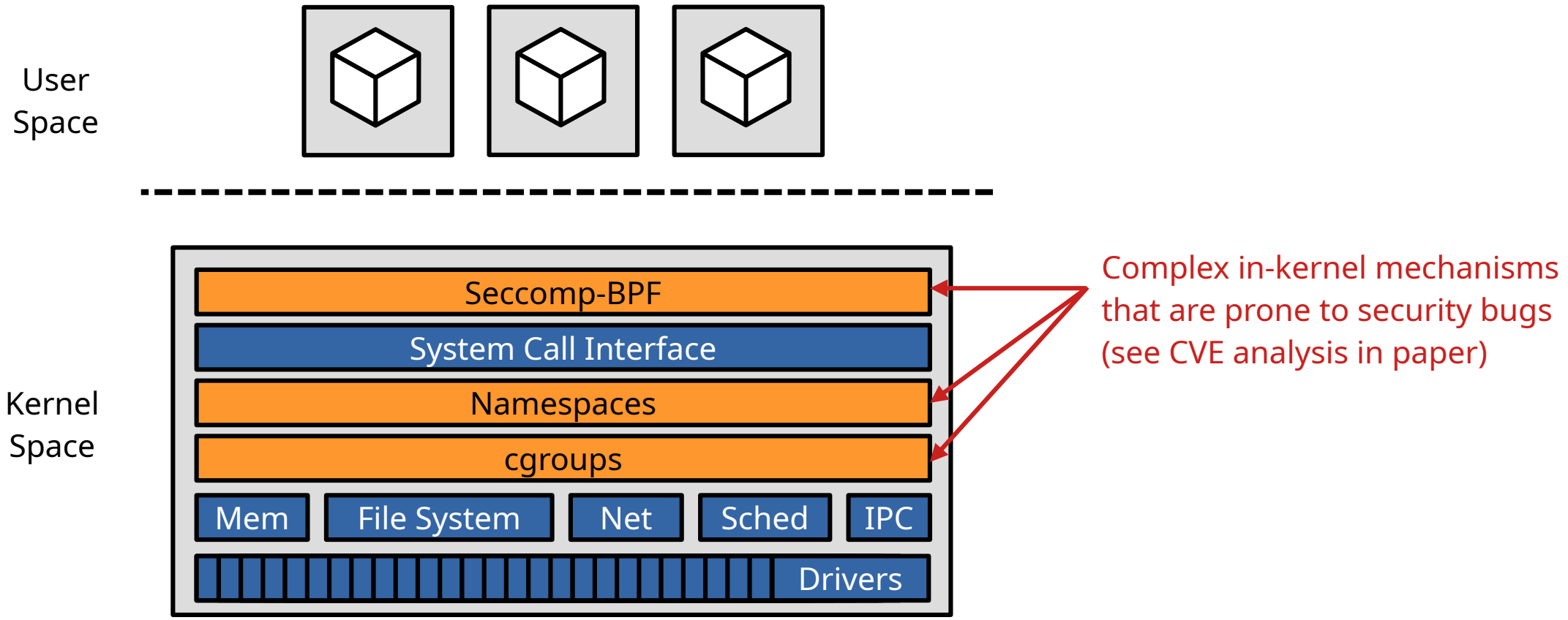
Containers On Monoliths – A Security Perspective

- Containers try to retroactively fix an OS design with weak isolation



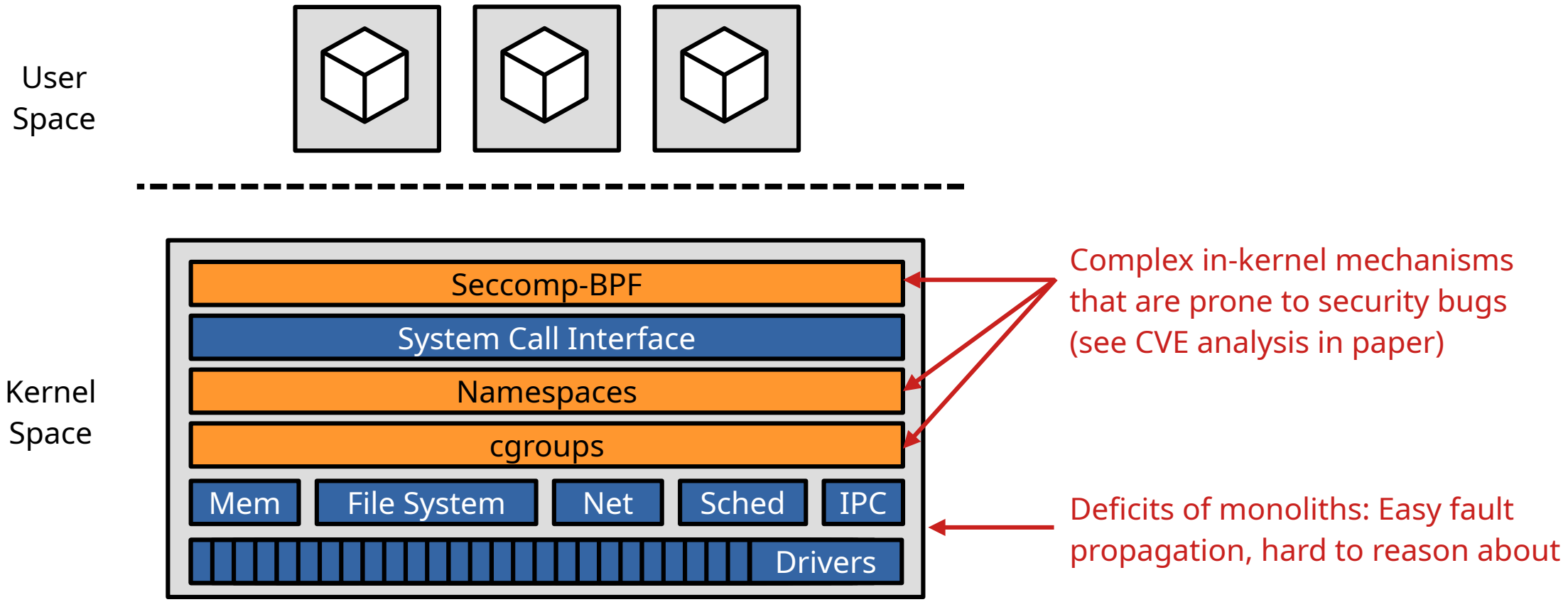
Containers On Monoliths – A Security Perspective

- Containers try to retroactively fix an OS design with weak isolation



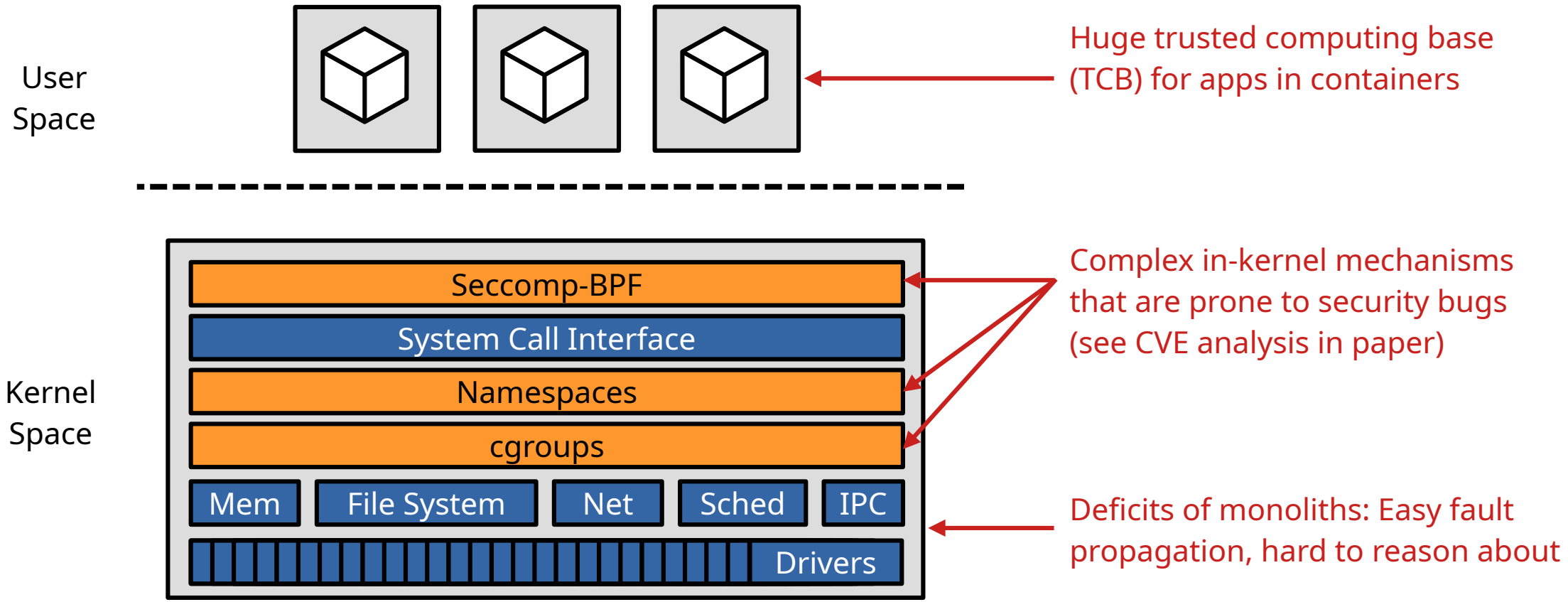
Containers On Monoliths – A Security Perspective

- Containers try to retroactively fix an OS design with weak isolation



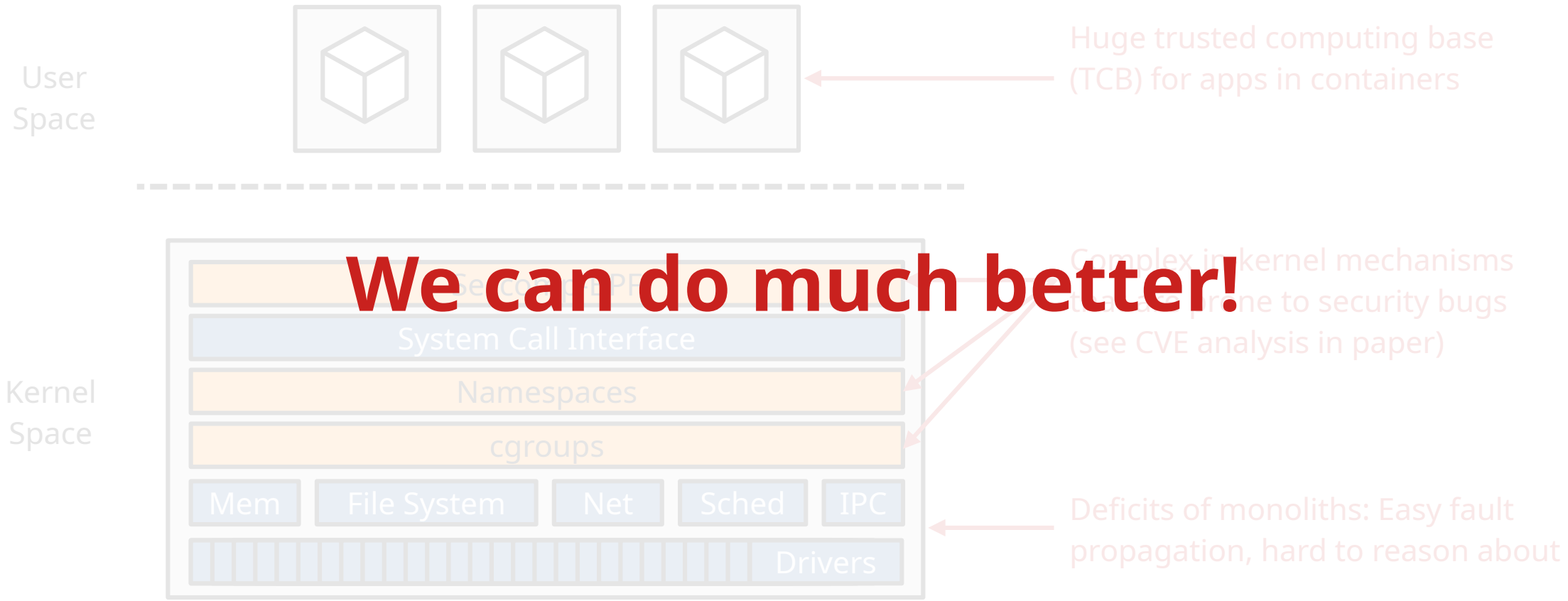
Containers On Monoliths – A Security Perspective

- Containers try to retroactively fix an OS design with weak isolation



Containers On Monoliths – A Security Perspective

- Containers try to retroactively fix an OS design with weak isolation



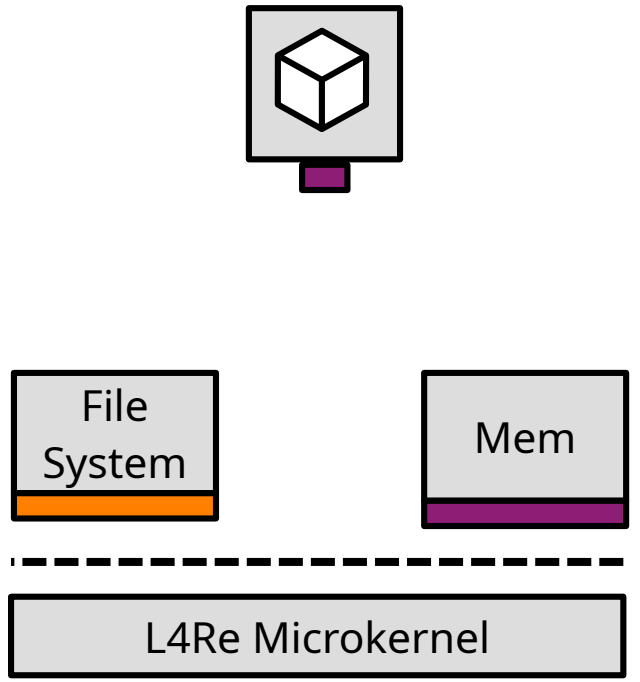
A Different Approach To Container Isolation



- Build containers on L4Re, a microkernel OS enforcing PoLA by design
  - Capability-based access control: Explicit configuration of access rights

A Different Approach To Container Isolation

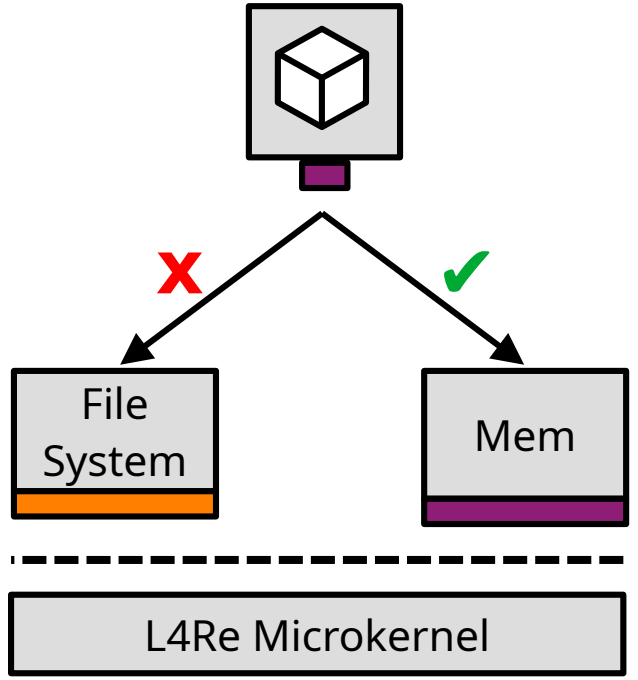
- Build containers on L4Re, a microkernel OS enforcing PoLA by design
  - Capability-based access control: Explicit configuration of access rights



A Different Approach To Container Isolation

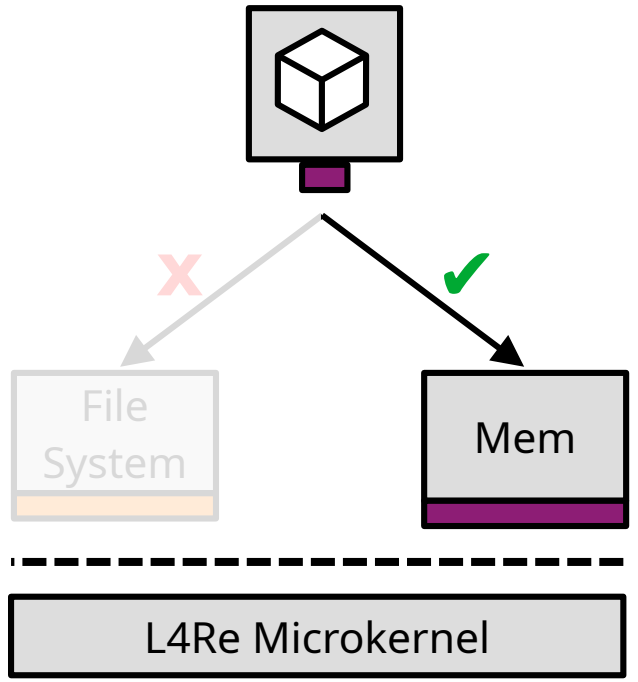
- Build containers on L4Re, a microkernel OS enforcing PoLA by design
  - Capability-based access control: Explicit configuration of access rights

- Interface restrictions: Not needed with capabilities



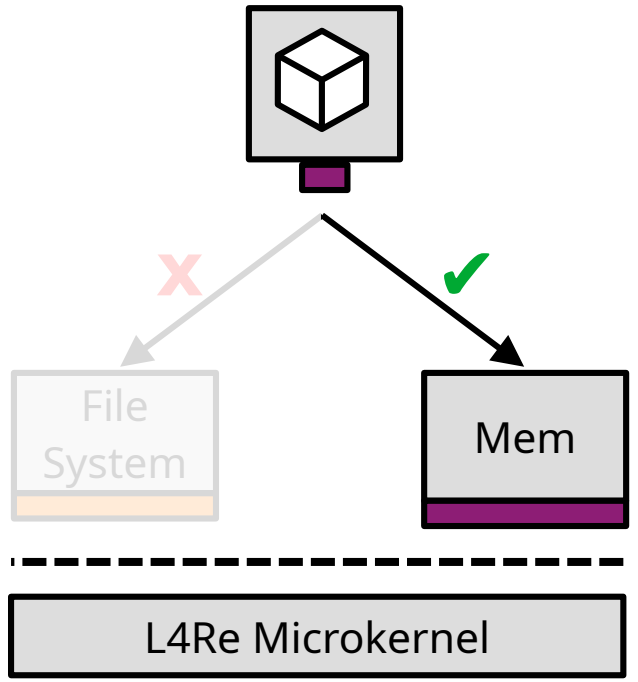
A Different Approach To Container Isolation

- Build containers on L4Re, a microkernel OS enforcing PoLA by design
  - Capability-based access control: Explicit configuration of access rights
- Interface restrictions: Not needed with capabilities
- Visibility restrictions: Implicit through capability setup
  - No global namespaces exist



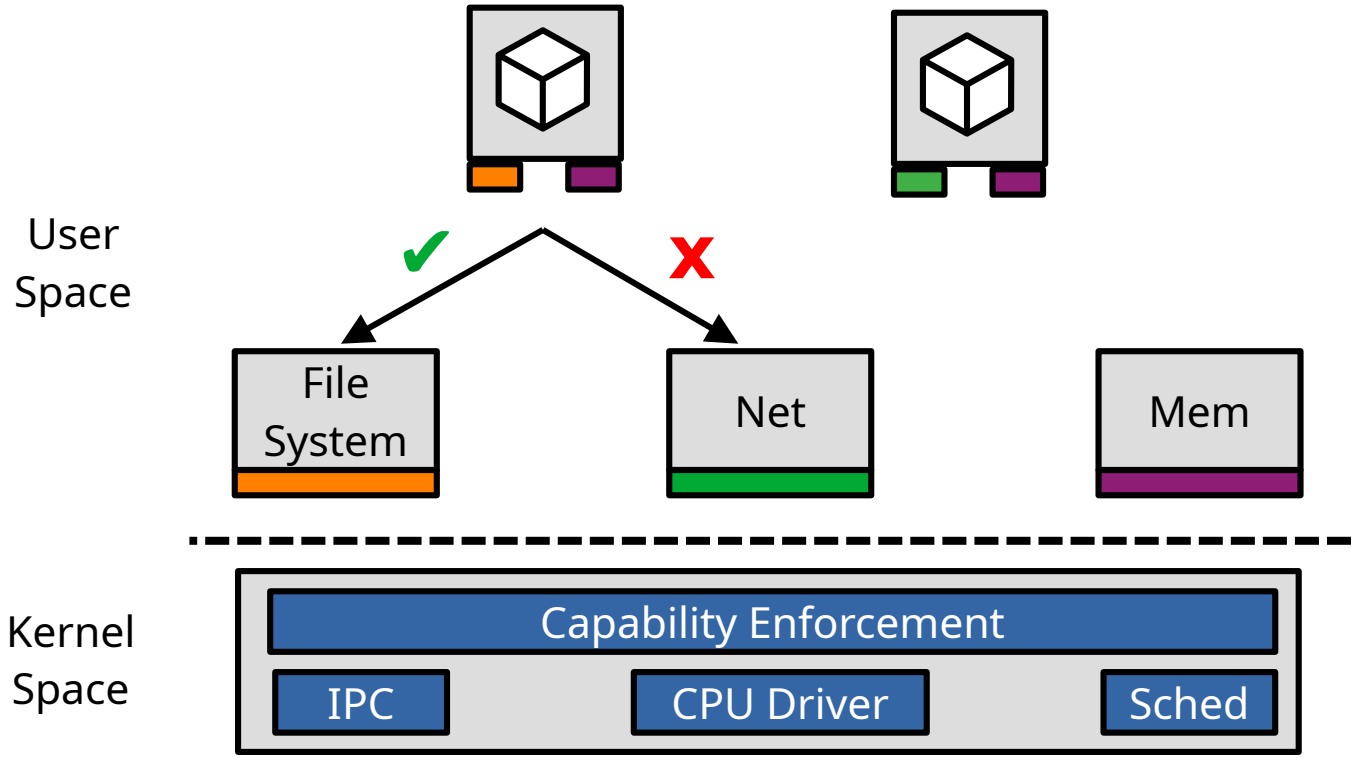
A Different Approach To Container Isolation

- Build containers on L4Re, a microkernel OS enforcing PoLA by design
  - Capability-based access control: Explicit configuration of access rights
- Interface restrictions: Not needed with capabilities
- Visibility restrictions: Implicit through capability setup
  - No global namespaces exist
- Resource restrictions: Implemented by each service
  - Resource context tied to service session



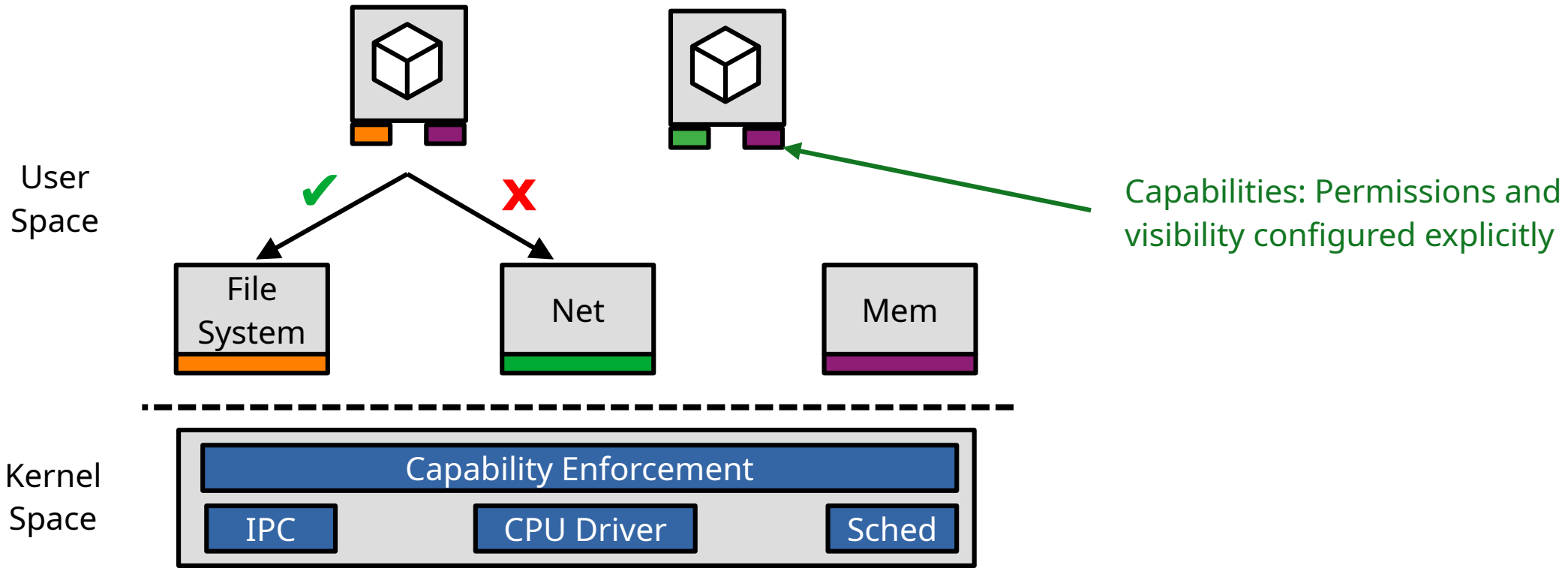
What Do We Gain In Terms Of Security?

- Containers on L4Re are conceptually much simpler than on Linux
  - See paper for detailed discussion of security benefits



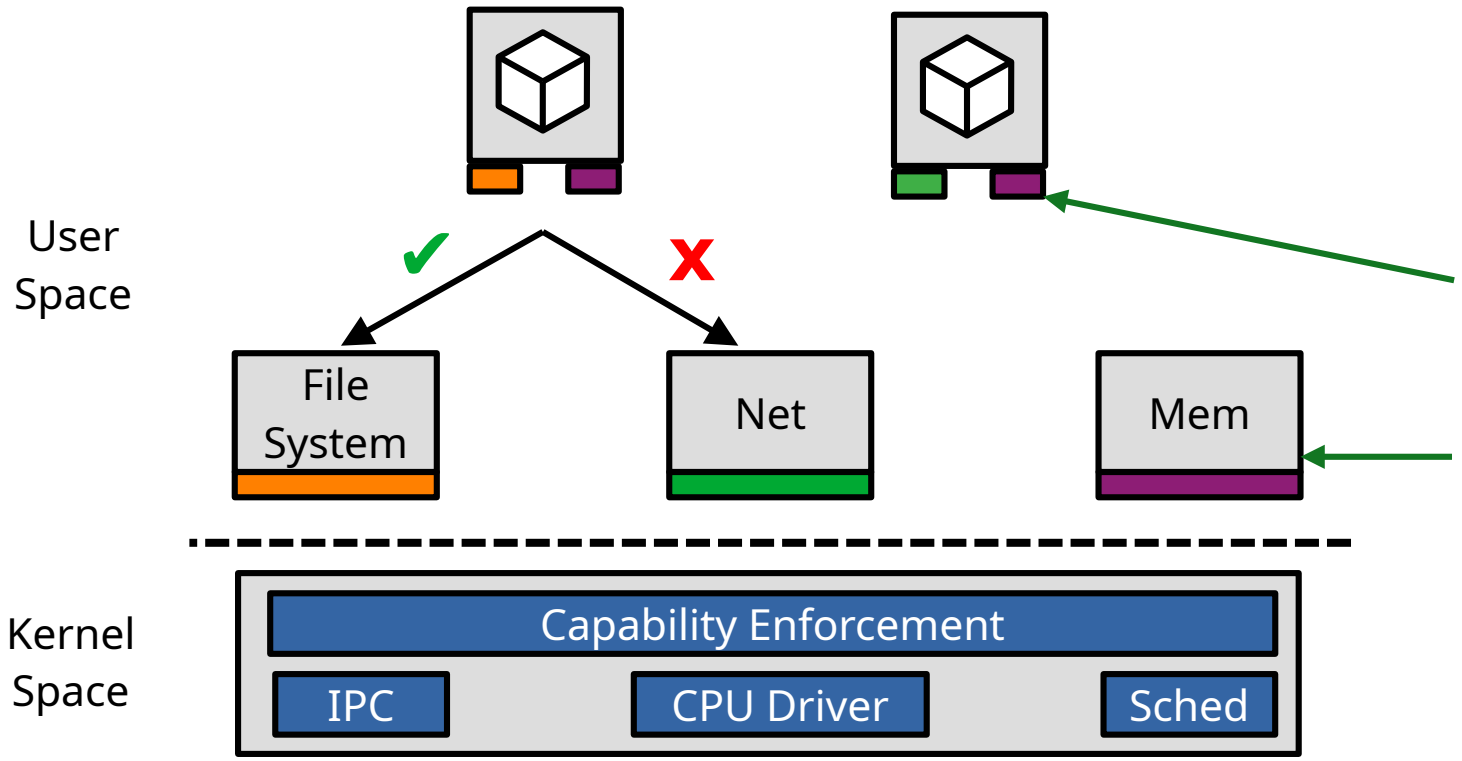
What Do We Gain In Terms Of Security?

- Containers on L4Re are conceptually much simpler than on Linux
  - See paper for detailed discussion of security benefits



What Do We Gain In Terms Of Security?

- Containers on L4Re are conceptually much simpler than on Linux
  - See paper for detailed discussion of security benefits

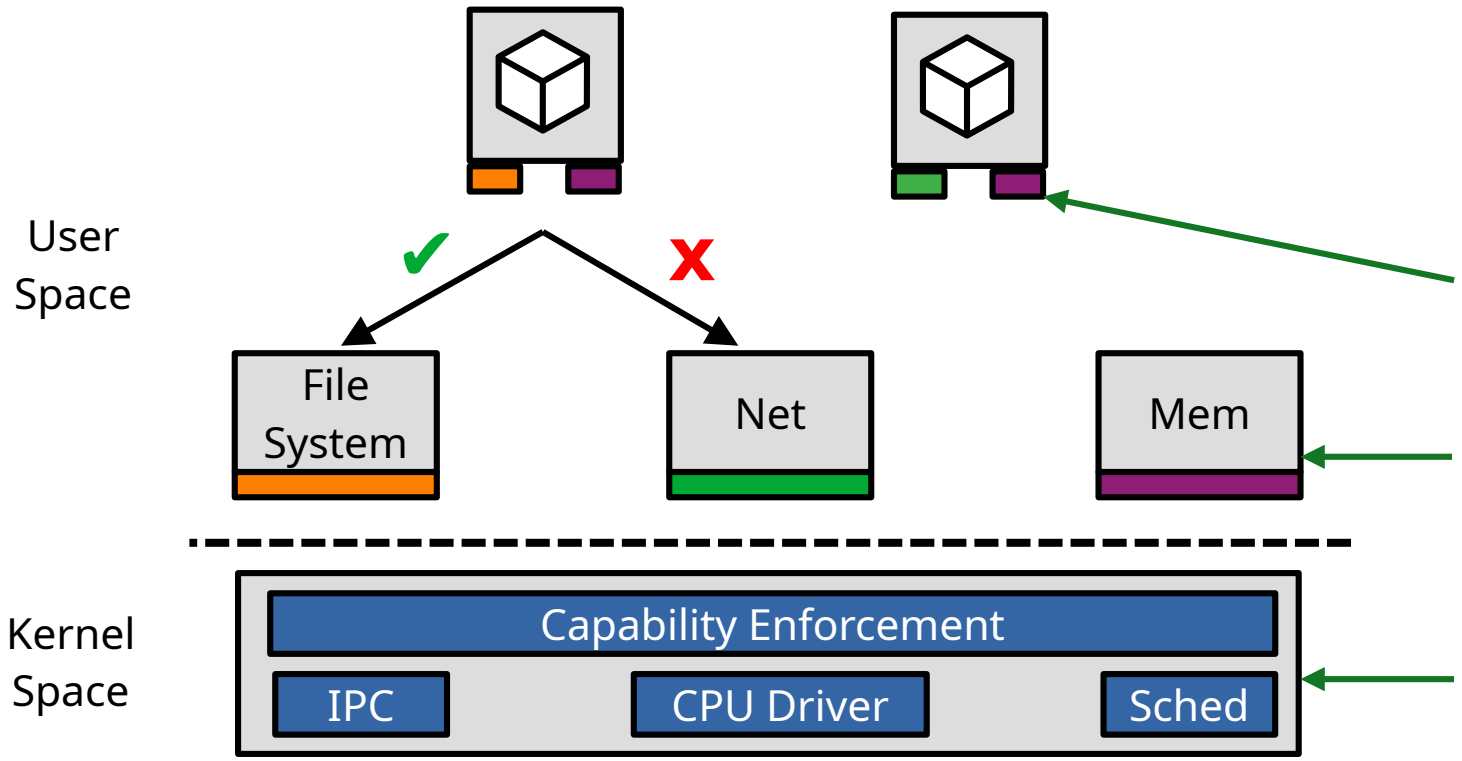


Capabilities: Permissions and visibility configured explicitly

Userspace system services: Fault containment, modularity

What Do We Gain In Terms Of Security?

- Containers on L4Re are conceptually much simpler than on Linux
  - See paper for detailed discussion of security benefits



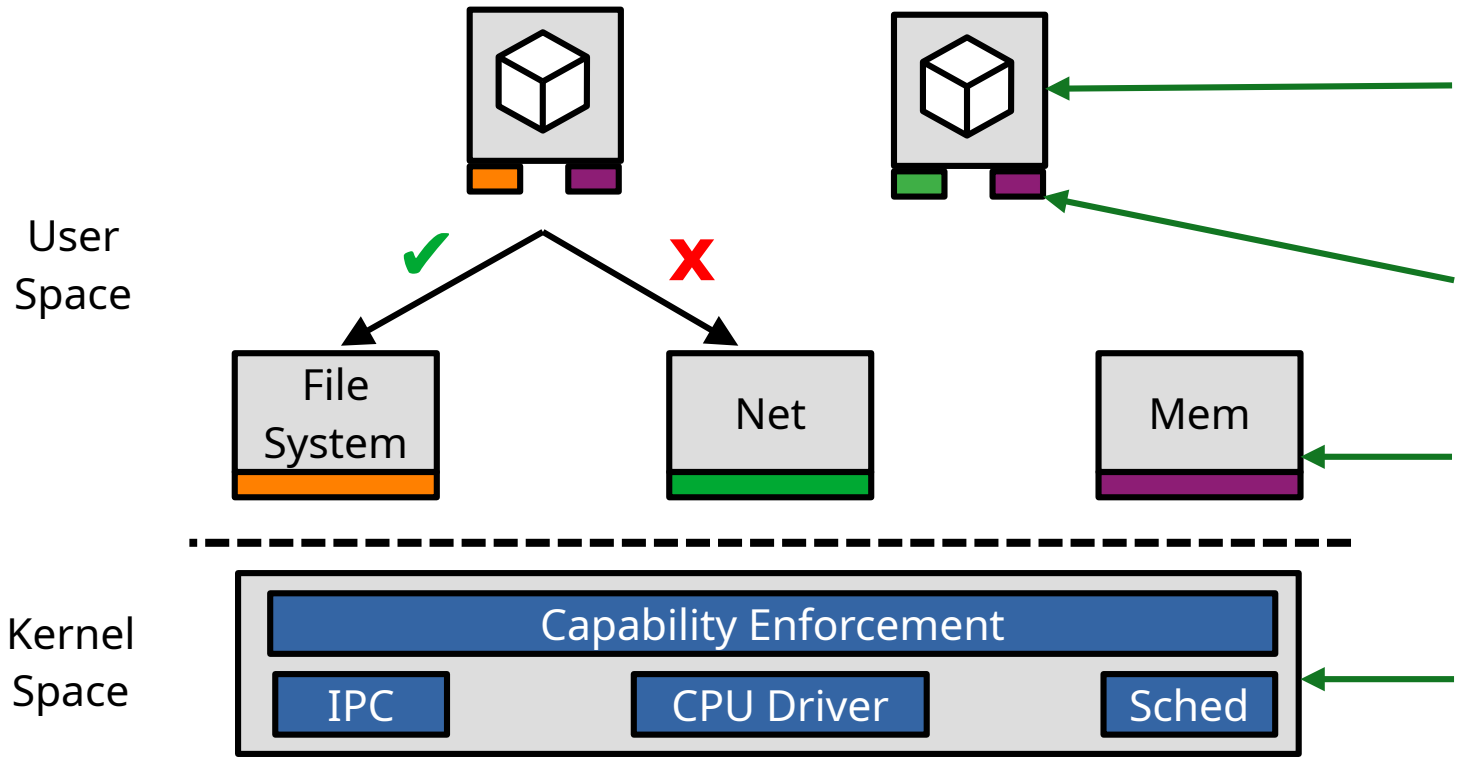
Capabilities: Permissions and visibility configured explicitly

Userspace system services: Fault containment, modularity

Certified microkernel reduces attack surface of whole system.

What Do We Gain In Terms Of Security?

- Containers on L4Re are conceptually much simpler than on Linux
  - See paper for detailed discussion of security benefits



Containers' TCB for security and availability only comprises components used.

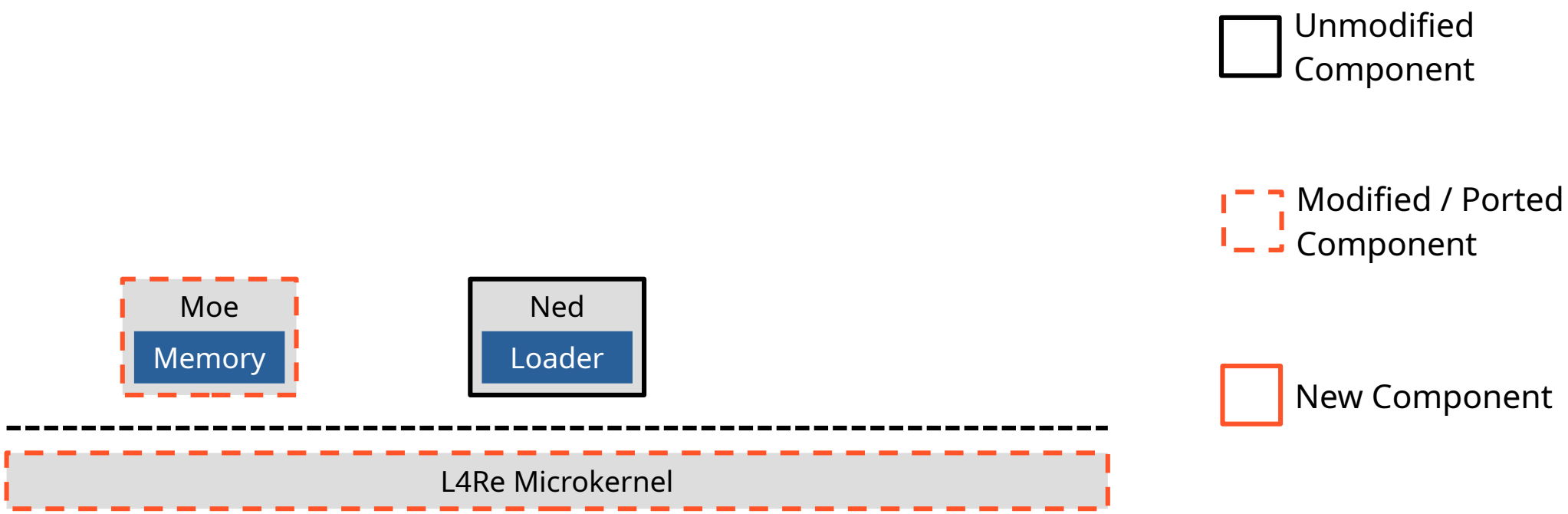
Capabilities: Permissions and visibility configured explicitly

Userspace system services: Fault containment, modularity

Certified microkernel reduces attack surface of whole system.

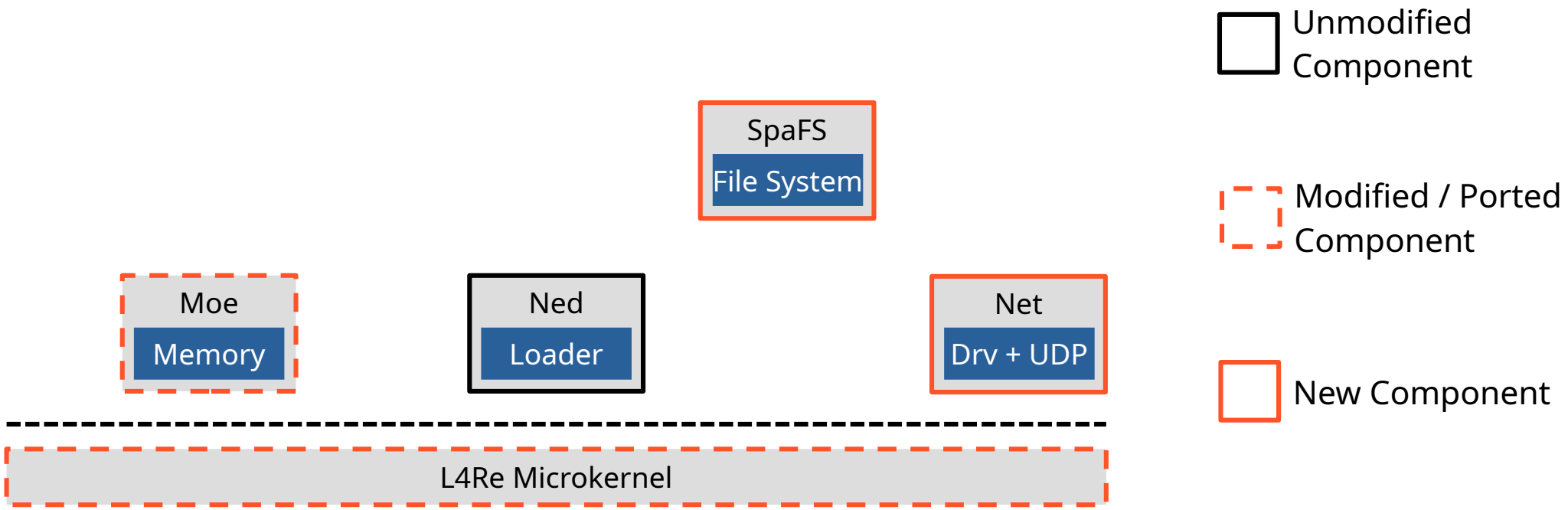
Implementing MettEagle

- Container users expect ecosystem and scalability
  - Only small modifications of L4Re base system required



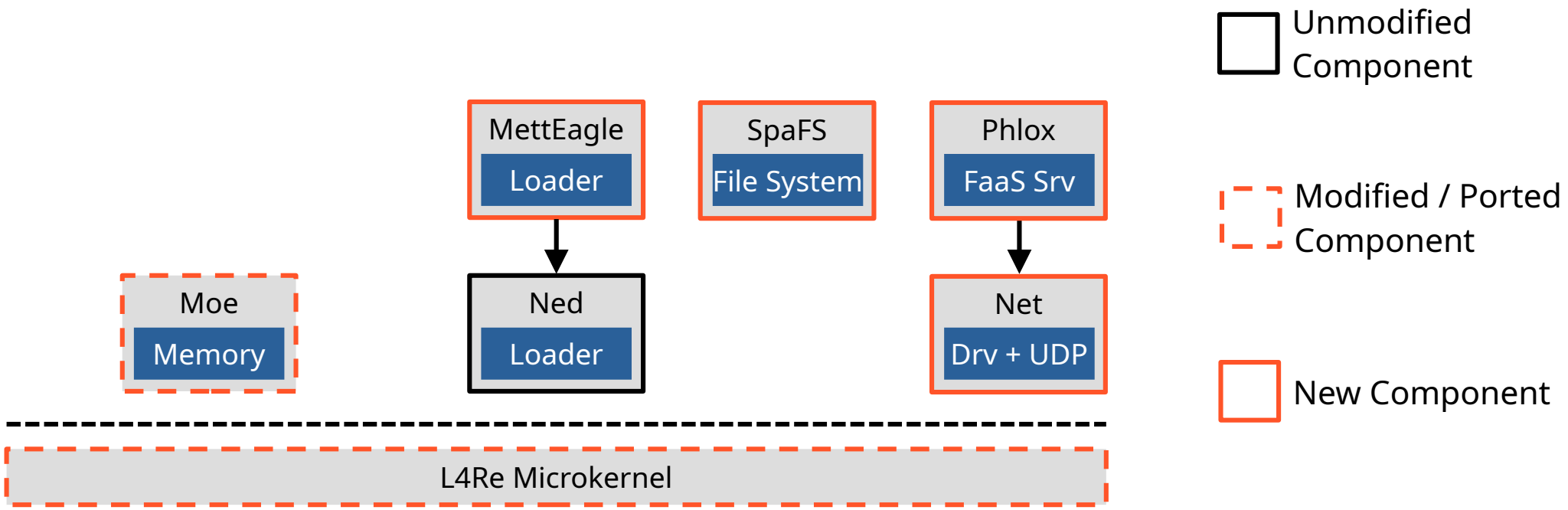
Implementing MettEagle

- Container users expect ecosystem and scalability
  - Only small modifications of L4Re base system required



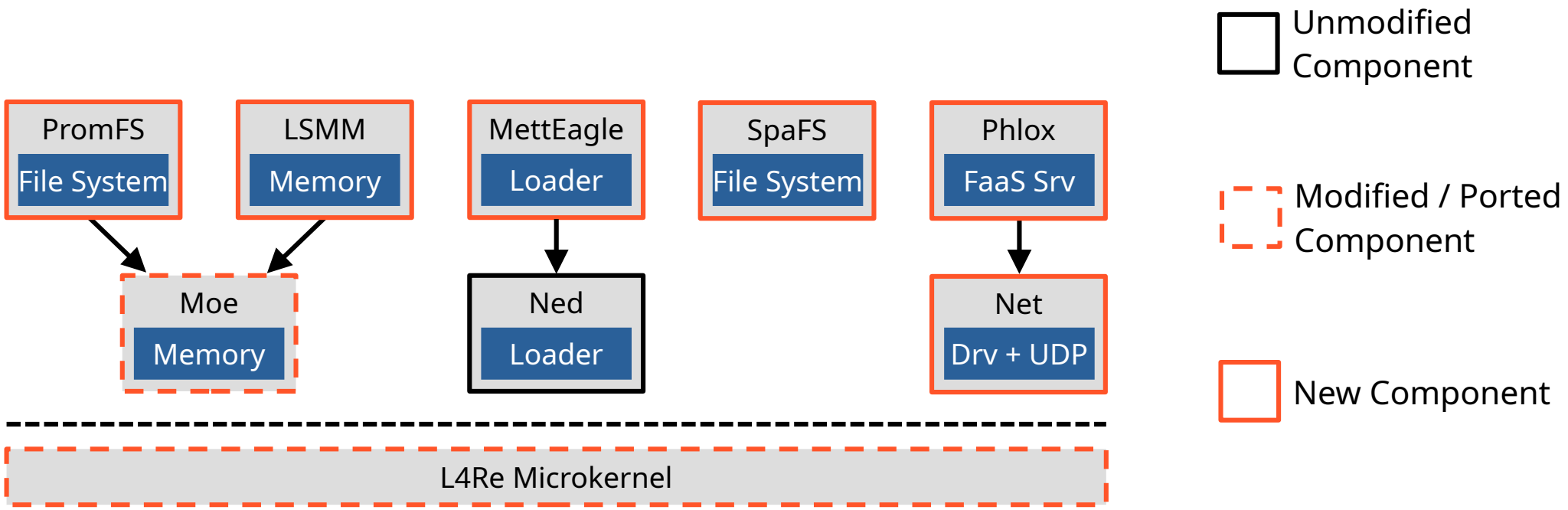
Implementing MettEagle

- Container users expect ecosystem and scalability
  - Only small modifications of L4Re base system required



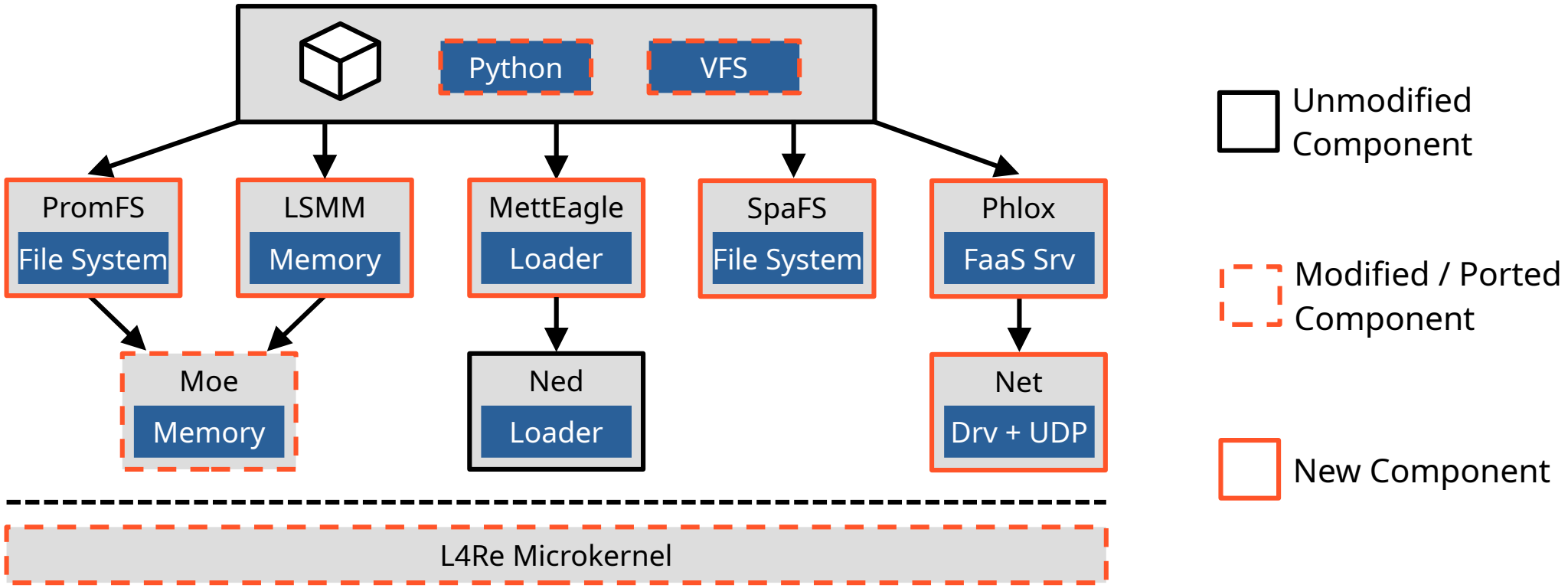
Implementing MettEagle

- Container users expect ecosystem and scalability
  - Only small modifications of L4Re base system required



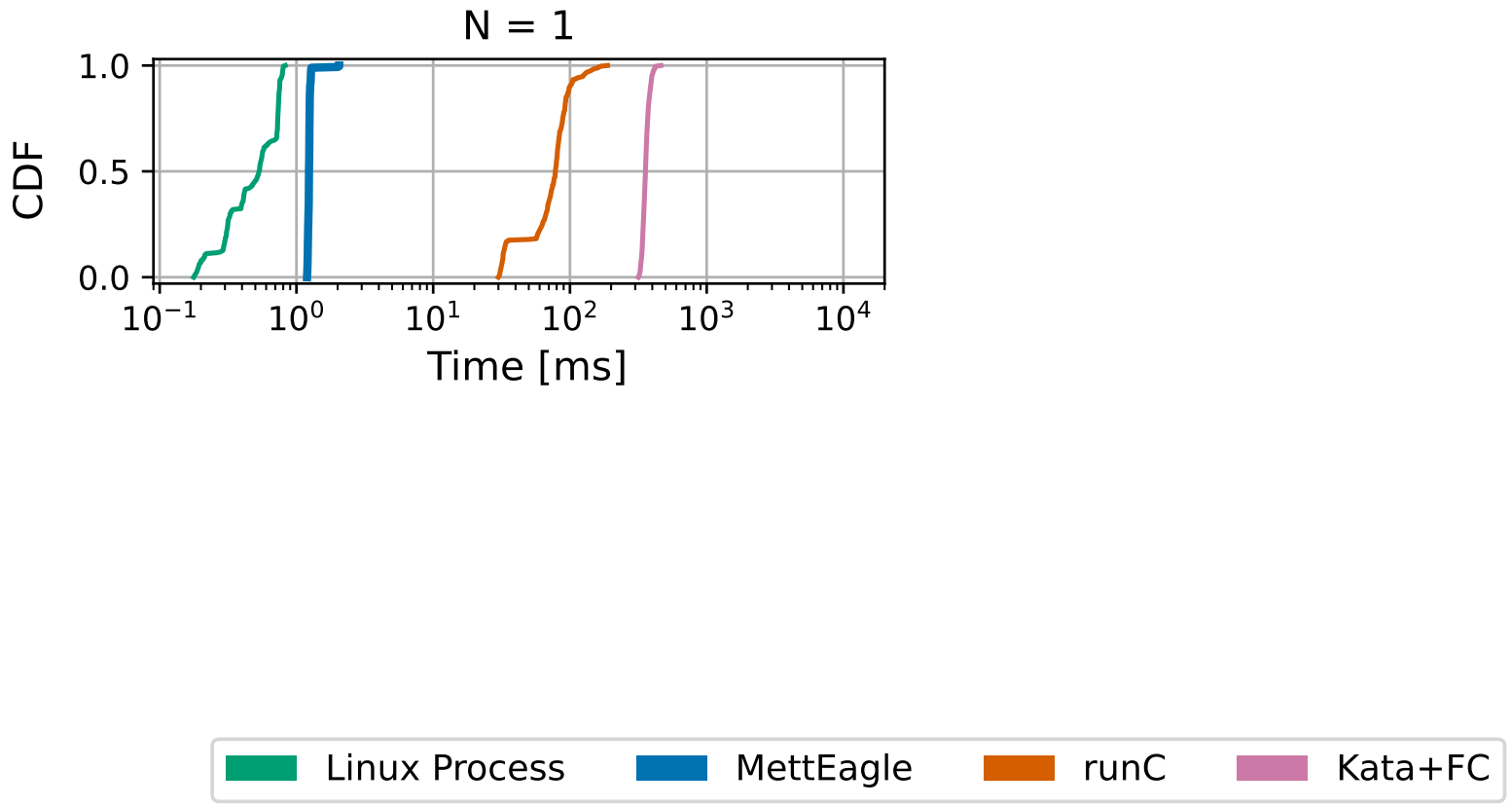
Implementing MettEagle

- Container users expect ecosystem and scalability
  - Only small modifications of L4Re base system required



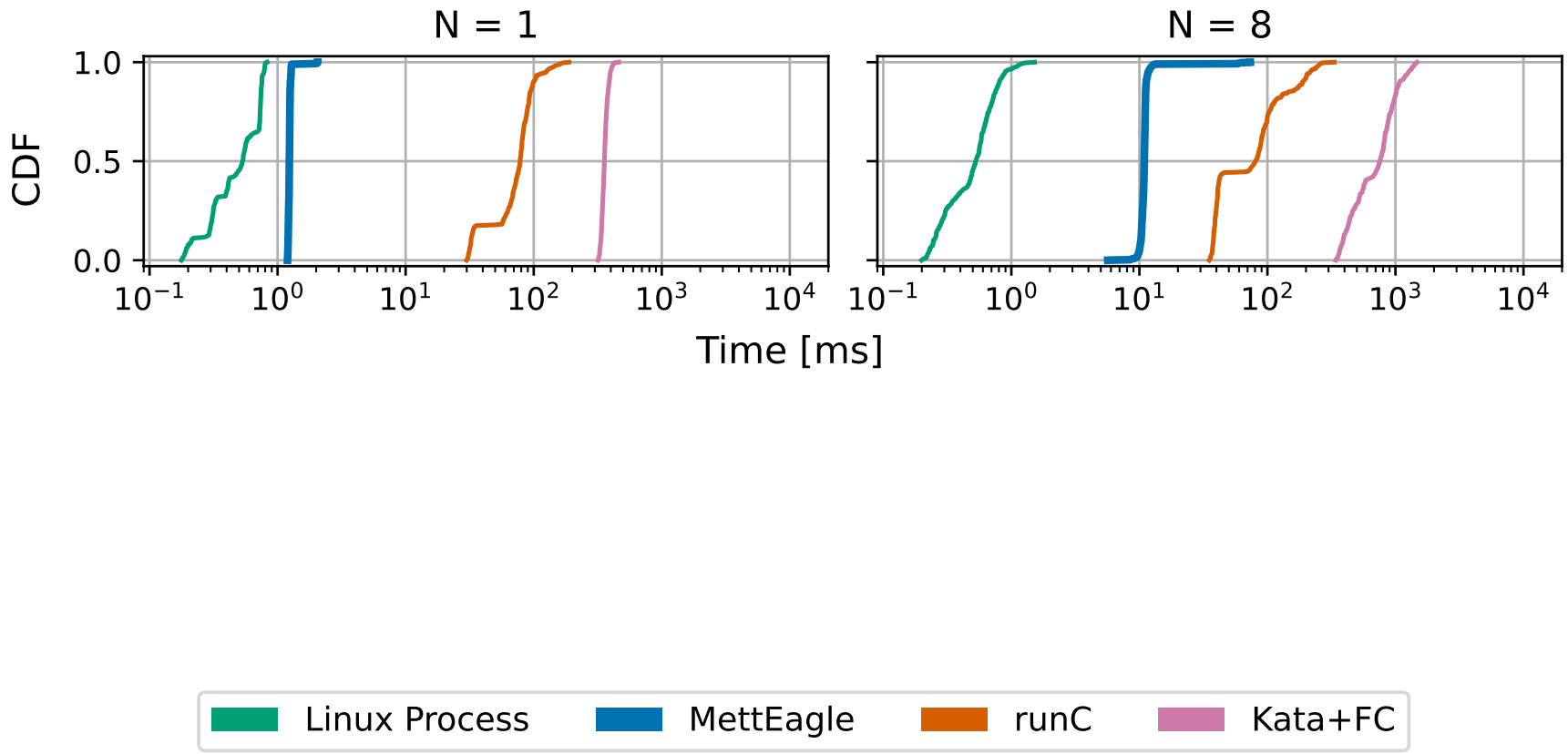
Performance Evaluation – Parallel Container Cold Start

- Benchmarks on dual-socket server machine (2 x 32 Cores, 512 GB RAM)
  - Fine-grained locking in microkernel key to performance



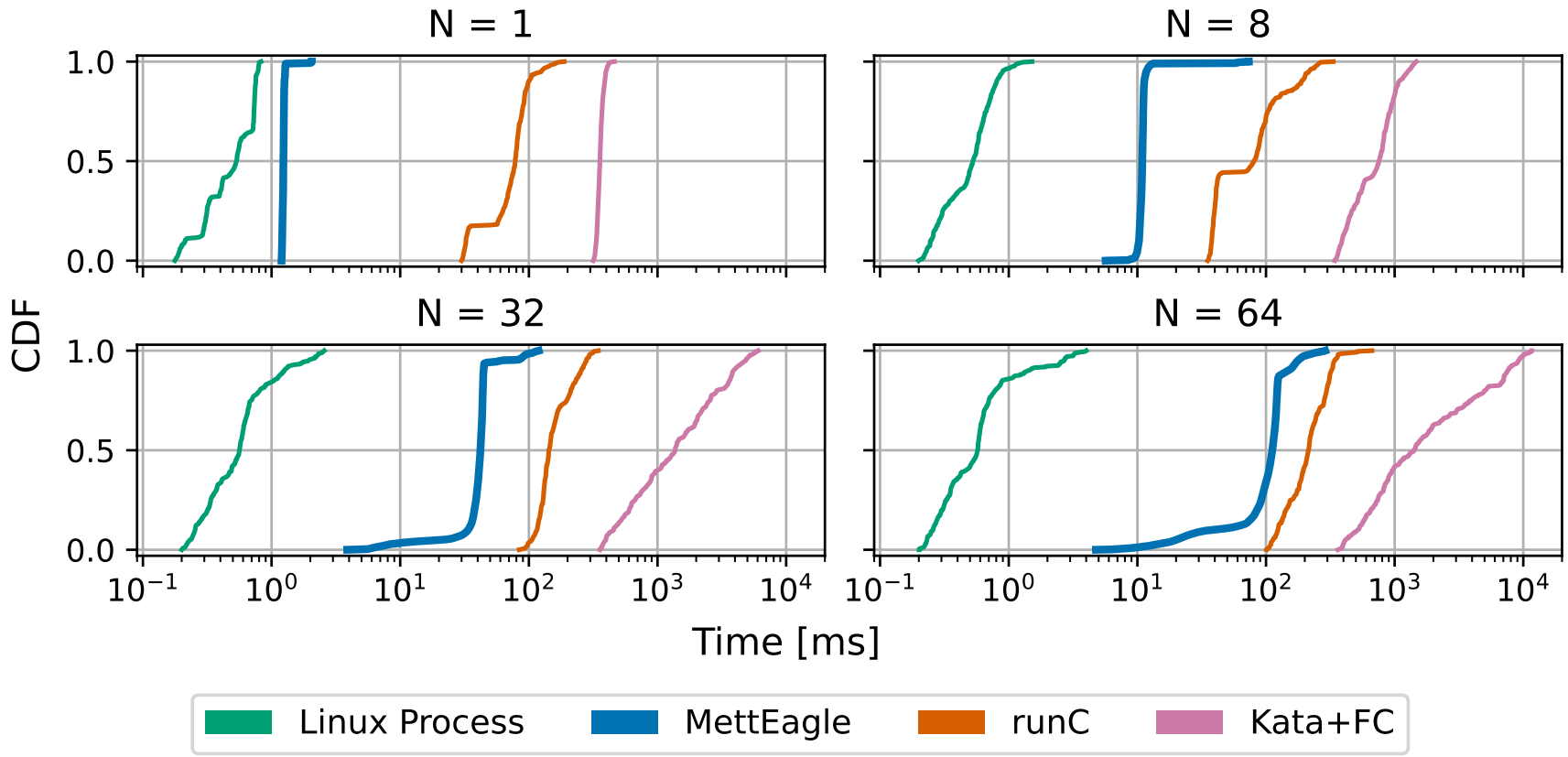
Performance Evaluation – Parallel Container Cold Start

- Benchmarks on dual-socket server machine (2 x 32 Cores, 512 GB RAM)
  - Fine-grained locking in microkernel key to performance



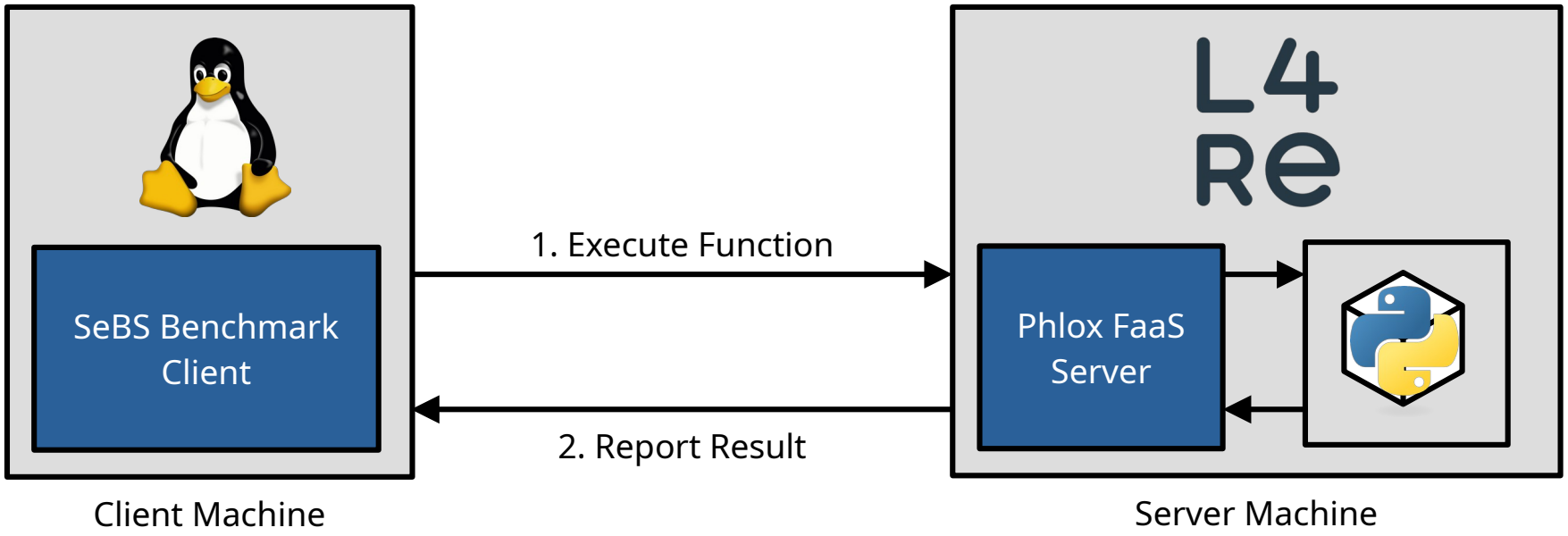
Performance Evaluation – Parallel Container Cold Start

- Benchmarks on dual-socket server machine (2 x 32 Cores, 512 GB RAM)
  - Fine-grained locking in microkernel key to performance



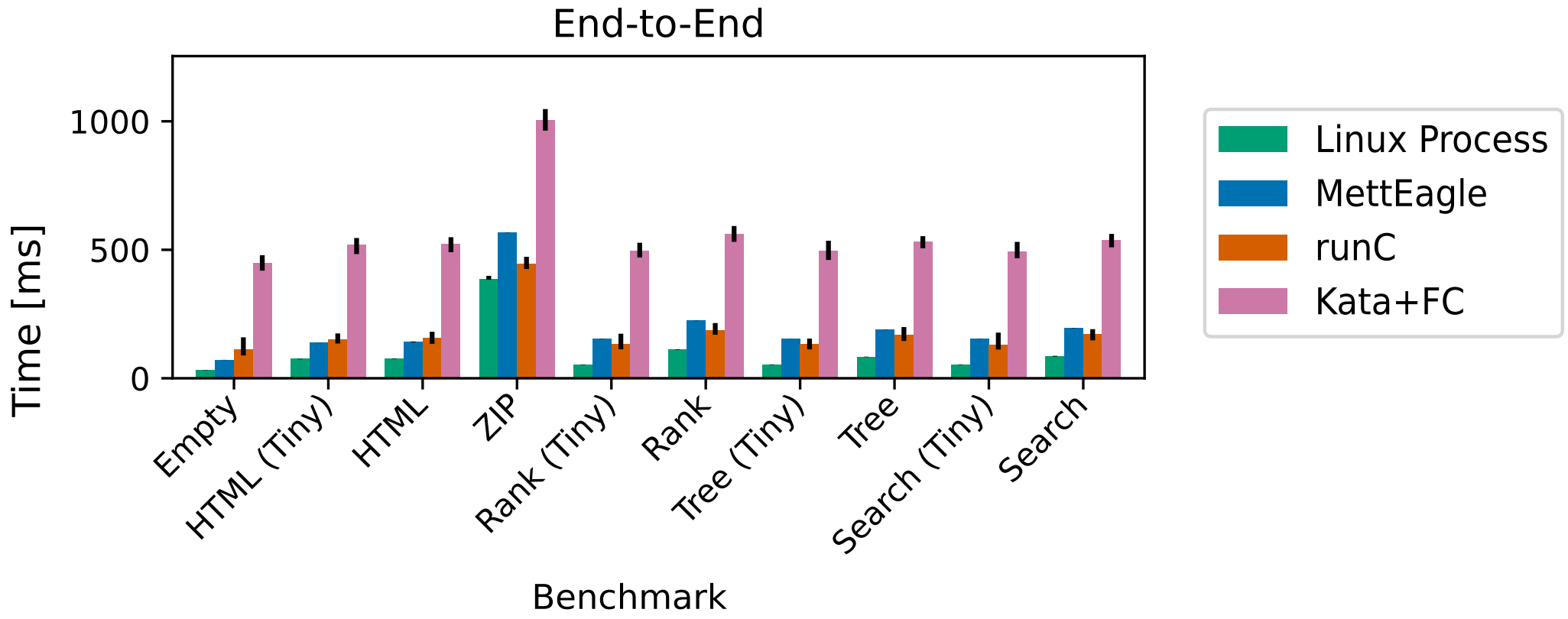
Performance Evaluation – Sequential FaaS Benchmark

- Linux client running SeBS benchmark exercises FaaS server prototype of L4Re
  - Execution of short Python scripts demanding for microkernel architecture

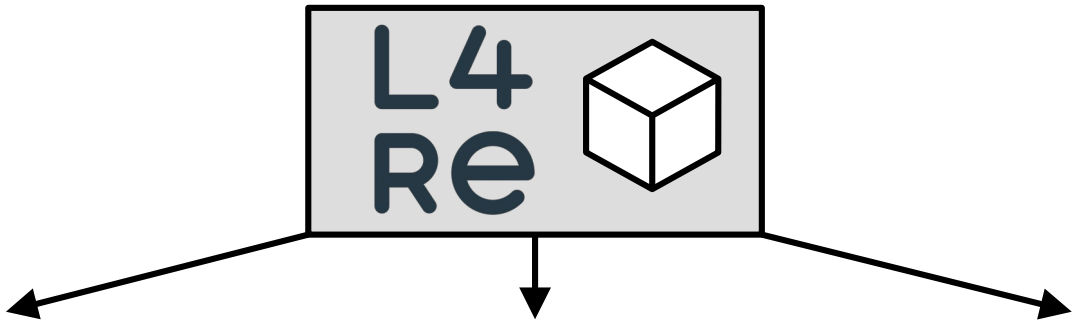


Performance Evaluation – Sequential FaaS Benchmark

- Linux client running SeBS benchmark exercises FaaS server prototype of L4Re
  - More numbers (network performance, parallel FaaS) in the paper

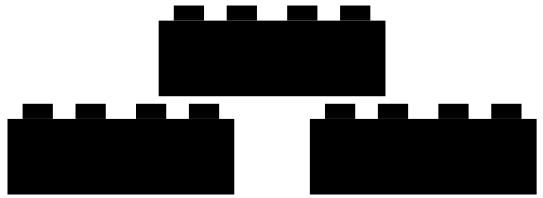


Summary: Containers on Microkernels

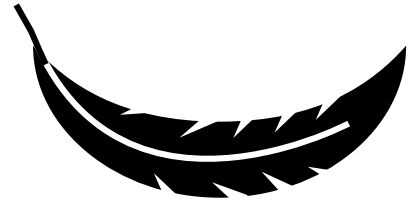




More secure than containers on monoliths

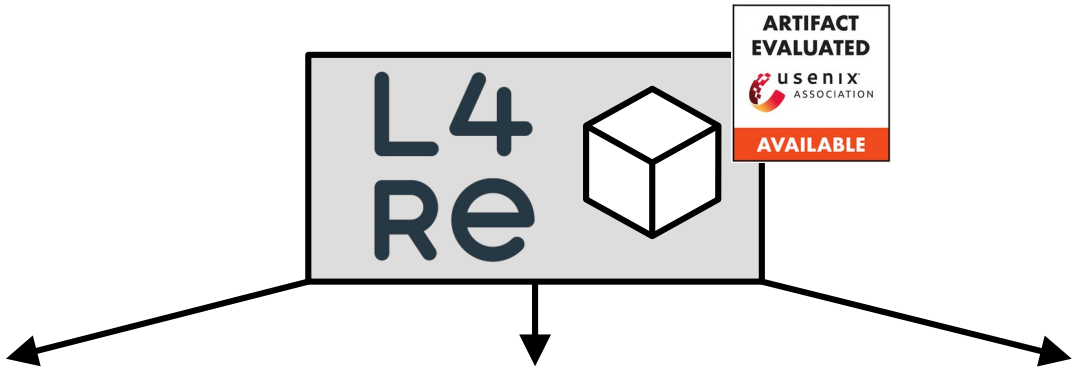


Conceptually simple and modular design



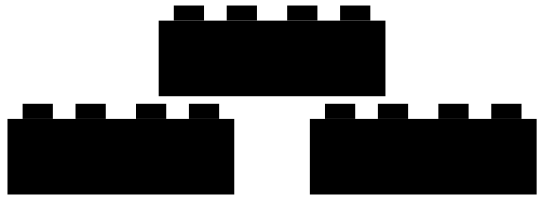
Low overhead, also for cloud workloads

Summary: Containers on Microkernels

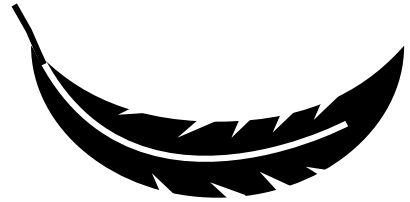




More secure than containers on monoliths



Conceptually simple and modular design



Low overhead, also for cloud workloads

L4Re on GitHub:

https://github.com/kernkonzept

Check out the paper artifact:

https://doi.org/10.5281/zenodo.15211297

Backup Slides

Security Evaluation – Code Complexity

Component of MettEagle	SLOC
L4Re kernel	41,406
Sigma0 (root pager)	1,249
Moe (root task)	4,810
Ned (system loader)	2,870
IO (PCI driver)	22,012
LSMM (parallelized memory manager)	4,289
PROMFS (parallelized boot FS)	780
SPAFS (file system)	501
LUNA (NIC driver & UDP stack)	8,735
Compartment Service	1,793
Phlox (FaaS launcher)	826
Σ	89,271

SLOCCOUNT of MettEagle Container Stack

Component of Linux	SLOC
Linux kernel v6.7.4 (including ramfs)	1,454,290
NIC driver module	31,675
containerd	922,901
runc	290,946
Σ	2,699,812

SLOCCOUNT of Linux Container Stack

Security Evaluation – CVE Analysis

CVE	Cat.	Eval.	CVE	Cat.	Eval.	CVE	Cat.	Eval.
CVE-2023-39191	sc	FM	CVE-2021-4197	cg	PM	CVE-2017-18509	ns	FM
CVE-2023-35001	ns	PM	CVE-2021-3600	sc	FM	CVE-2017-18367	sc	FM
CVE-2023-31248	ns	PM	CVE-2021-3493	ns	PM	CVE-2017-17450	ns	N
CVE-2022-34918	ns	PM	CVE-2020-25220	cg	PM	CVE-2017-17448	ns	N
CVE-2022-32250	ns	PM	CVE-2019-11815	ns	N	CVE-2016-4805	ns	PM
CVE-2022-30594	sc	FM	CVE-2019-10639	ns	PM	CVE-2016-2853	ns	PM
CVE-2022-24122	ns	N	CVE-2019-9893	sc	FM	CVE-2016-1576	ns	PM
CVE-2022-1055	ns	FM	CVE-2019-2054	sc	FM	CVE-2015-1328	ns	PM
CVE-2022-0492	cg	PM	CVE-2018-16884	ns	PM	CVE-2015-8709	ns	FM
CVE-2022-0185	ns	PM	CVE-2018-18955	ns	FM	CVE-2014-1733	sc	FM
CVE-2021-4204	sc	FM	CVE-2017-1000111	ns	PM	CVE-2011-2189	ns	N

Table 3: List of CVEs classified by category (Cat.) and evaluation (Eval.). The categories consist of `seccomp` and / or `eBPF` (sc), `namespaces` (ns) and `cgroups` (cg) on Linux. The evaluation is marked as FM (fully mitigated), PM (partially mitigated), and N (not mitigated).

Performance Evaluation – Network Throughput

- Measured on a 10G Ethernet NIC using UDP

