

Blink and you miss it — VMM–OS Co-Design for Millisecond Boot of Isolated Serverless Applications

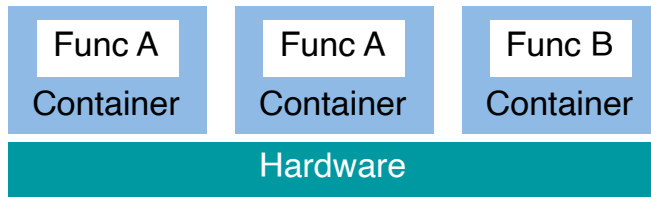
J. Klimt, S. Simon, M. Kröning, E. Zscheile, P. Vasilopoulos, S. Lankes, A. Monti

Isolating Serverless Functions

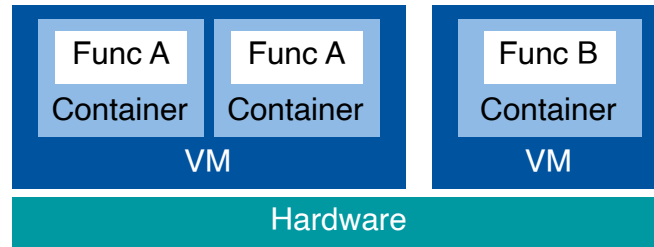
- Function as a Service (FaaS) runs in **multi-tenant cloud**
- Mutually distrusting customers

→ **Workload isolation**

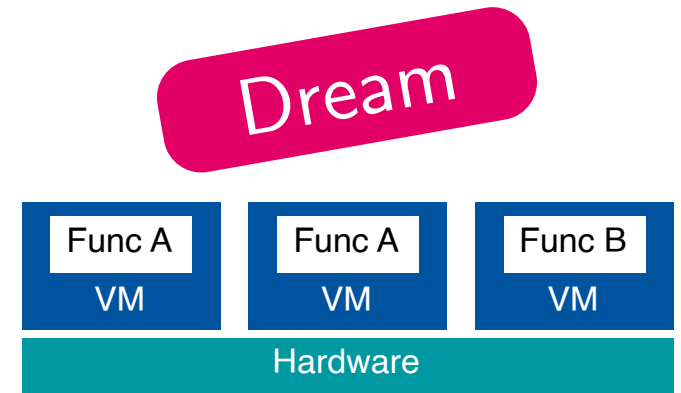
System Architectures:



(a) Containers: fast, but shared kernel
→ weak isolation



(b) Containers in VMs: Standard;
doubled overhead [5, 11]



(c) Purely VM based: one VM per
function

Problem:

- FaaS workloads run for ≈ 100 ms (median) [7]
- general-purpose VMMs boot-time: >170 ms.

Goal + Contributions

Goal: VM start-up fast enough for purely VM-based serverless

Idea: Design VMM and kernel *for each other*

Contributions: VMM-OS Co-Design & Application Templates

15 ms

"Hello World" VM runtime

2 ms

VM start to first packet

4–8×

less boot time than other
VMMs

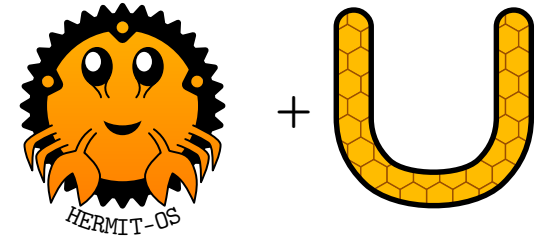
43 %

less memory consumption
on 128 instances

Design

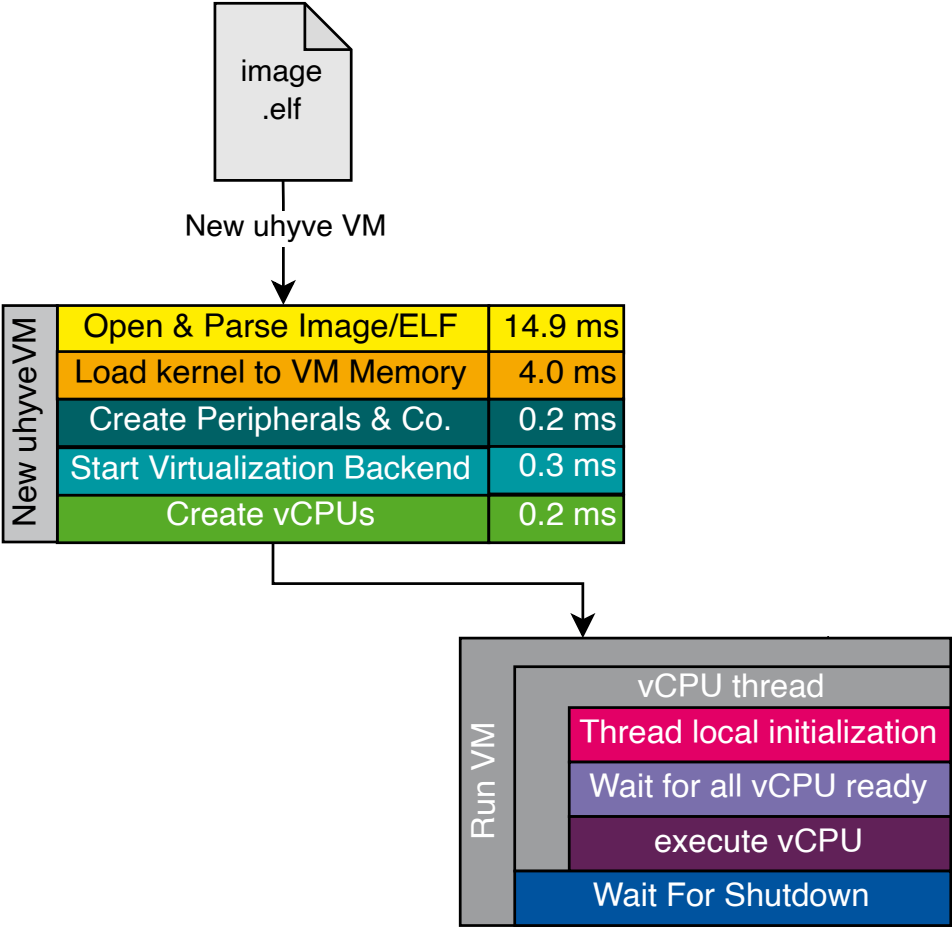
VMM-OS Co-Design

- We created the Uhyve VMM for the Hermit Unikernel.
- Uhyve knows its guest – no need for compatibility except Hermit:

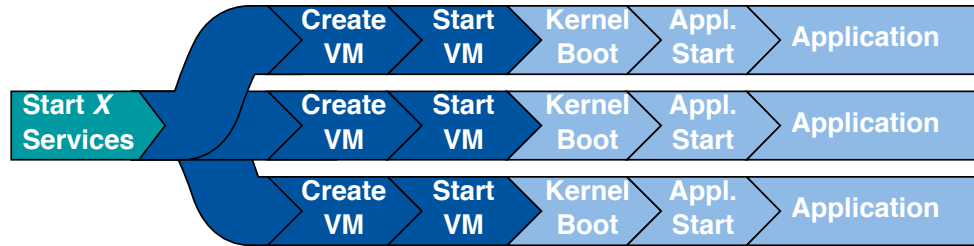


- Direct Boot:
 - ≡ Direct 64-bit VM
 - ≡ Hardware info, arguments and environment passed via FDT
 - ≡ ELF loading and relocation in the VMM
 - ≡ Kernel loaded to a random physical address → ASLR
 - ≡ Page tables set up by the VMM
 - ≡ No CPU/APIC/ACPI detection in Hermit
 - ≡ All vCPUs start simultaneously
- No device emulation, only virtio-net
- Hypercalls for:
 - ≡ Serial output
 - ≡ Shutdown with exit code
 - ≡ Sandboxed host file-system access
- Hypercall interface based on MMIO/PortIO:
 - ≡ Code shared with Hermit
- Library-first

Uhyve Execution Flow



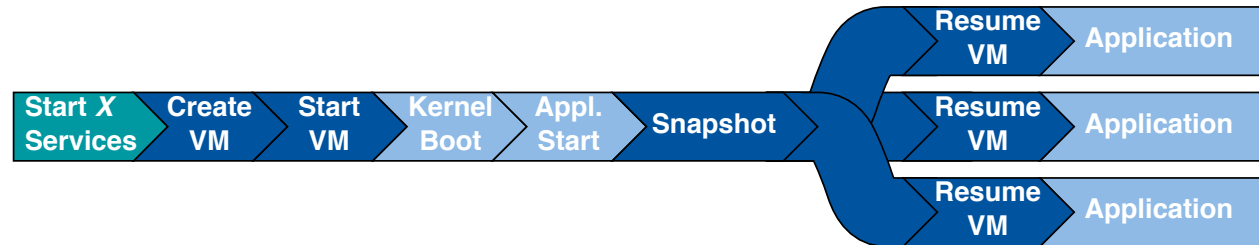
Application Templates



“Classic” flow: every VM is created and booted from scratch

Same runtimes for serverless functions are invoked **many times**:

- The **kernel** doesn't change
- The **runtime** doesn't change
- Only the **parameters** change



Application templates: kernel boot *and* application start-up are de-duplicated

Application Templates: The Snapshot Hypercall

Problem:

- Parts of the OS have to change.
- When is the application execution starting?

Solution: Snapshot Hypercall:

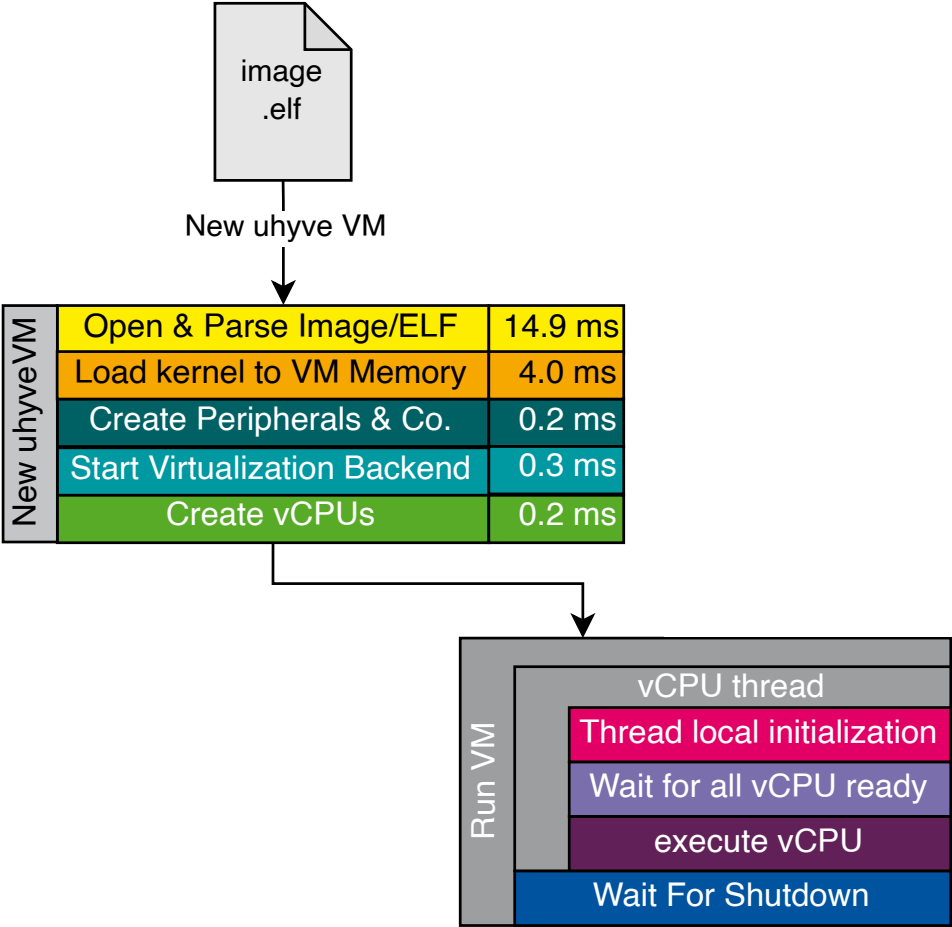
- Application decides when to snapshot:
instruction-accurate
- New appl. parameters in return value
- Kernel re-initializes the network stack

Kernel:

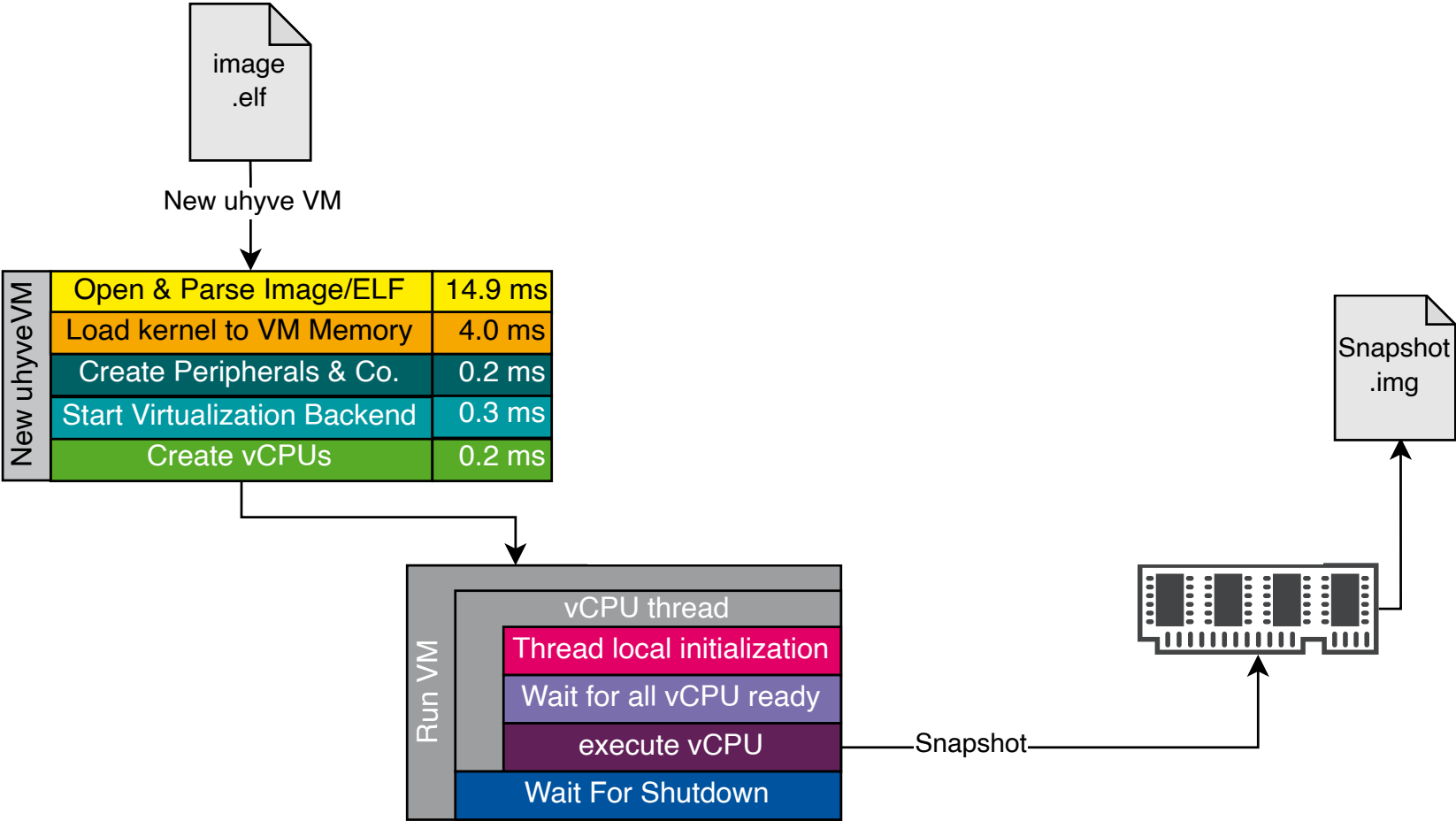
```
let mut s = SnapshotParams { ... };
uhyve_hypercall(Hypercall::Snapshot(&mut s));

if s.restored {
    if let Some(ip) = s.new_hermit_ip {
        // ...
    }
    network::reinit();
}
```

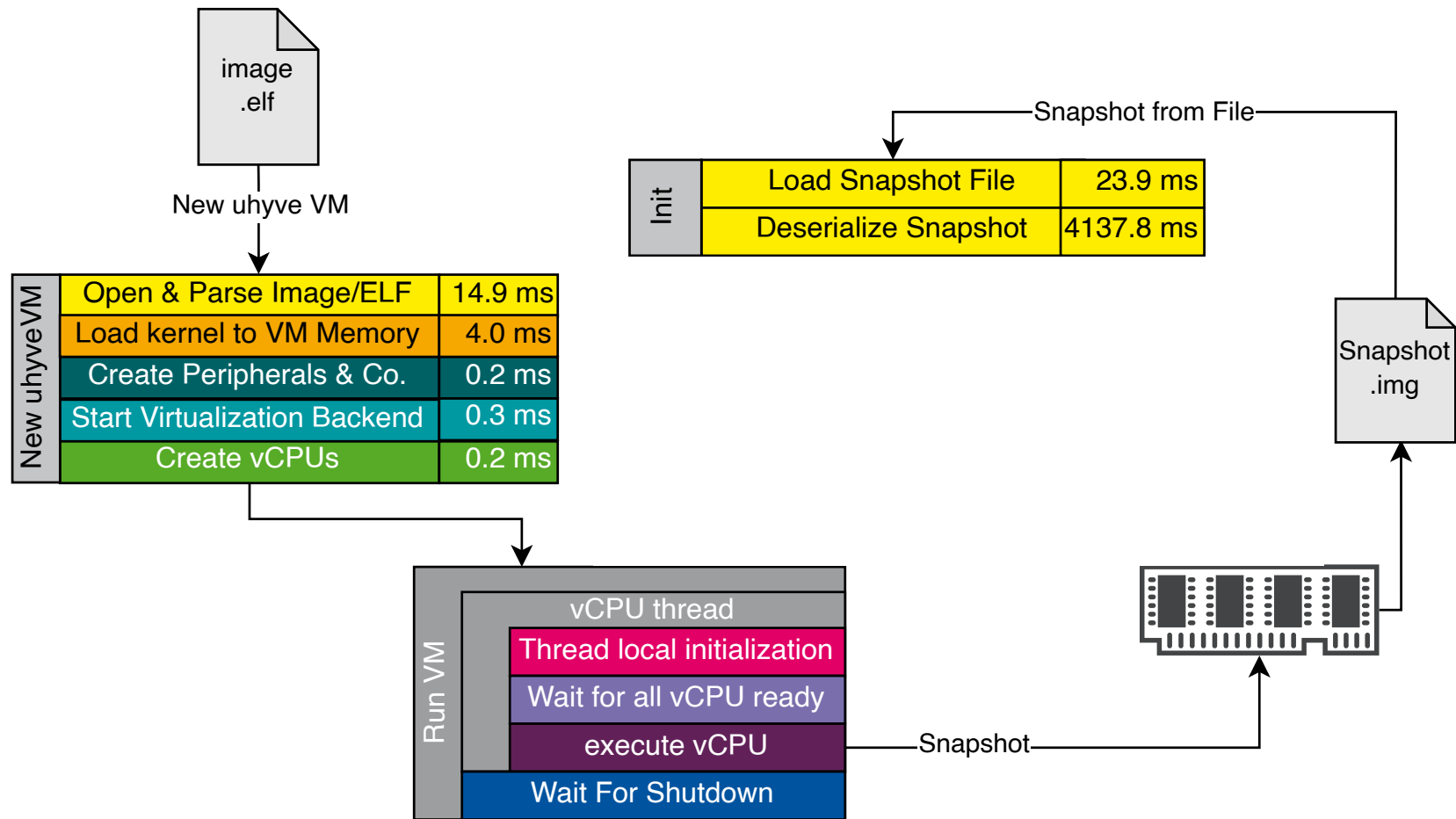
Uhyve Execution Flow



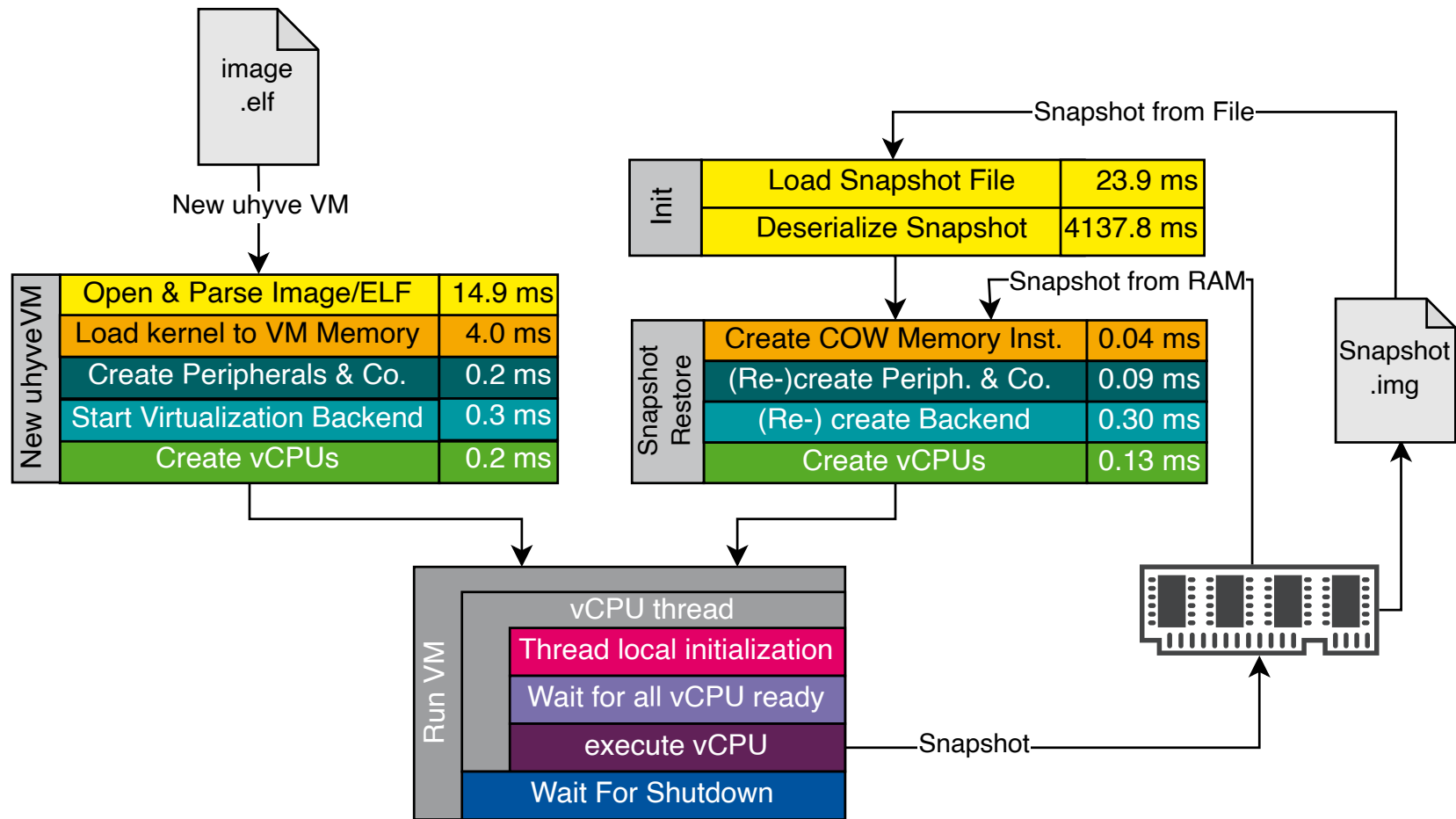
Uhyve Execution Flow



Uhyve Execution Flow



Uhyve Execution Flow

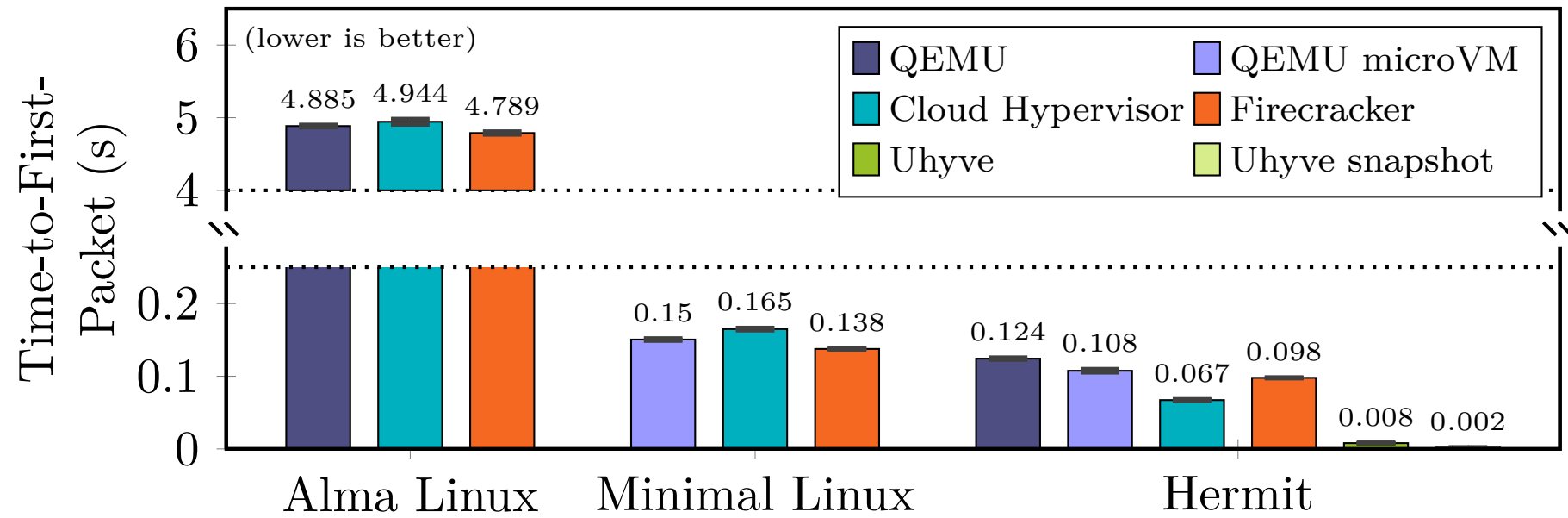


Evaluation

Evaluation Setup

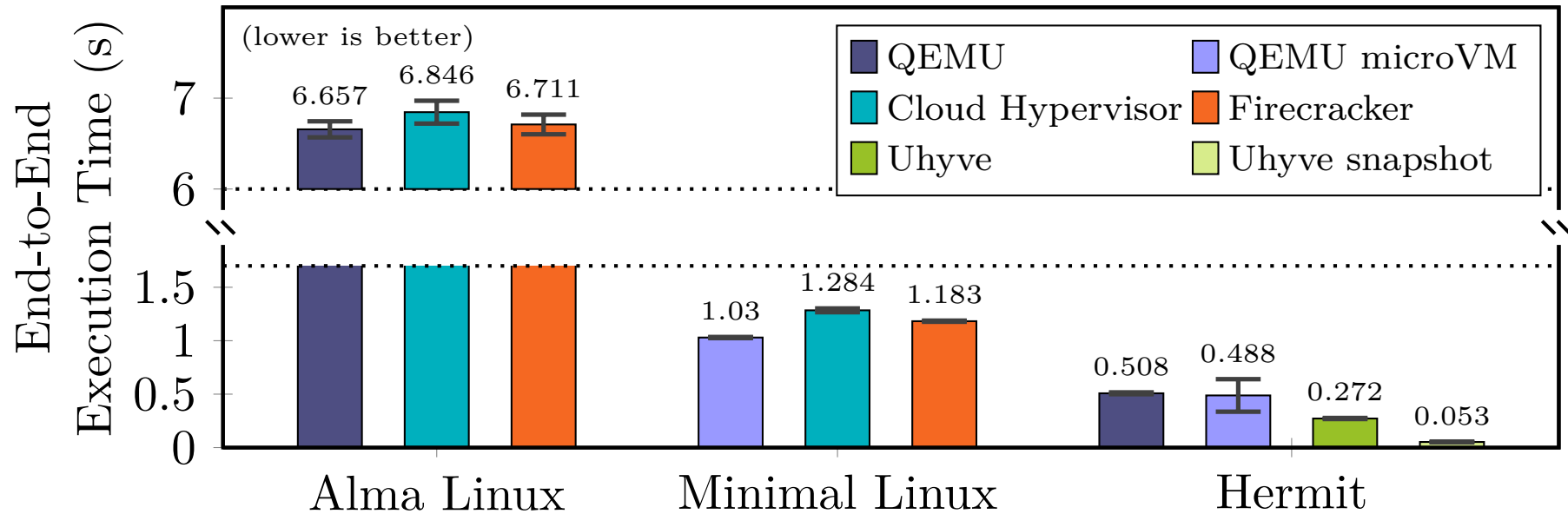
- **VMMs:** Uhyve, QEMU, QEMU microVM, Firecracker, Cloud Hypervisor
- **Guests:**
 - ≡ Alma Linux 10 (Generic Cloud) – a typical customer image
 - ≡ Minimal Linux (kernel 7.0.9 + initrd containing only the application)
 - ≡ Hermit
- “Application template” is already in RAM
- 1 vCPU, 2 GiB memory per VM
- AMD EPYC 9354P (32 cores), 128 GB DDR5, Ubuntu 25.10 (Linux 6.17)
- ≥ 25 runs and ≥ 30 s per benchmark

Time-to-first-Packet



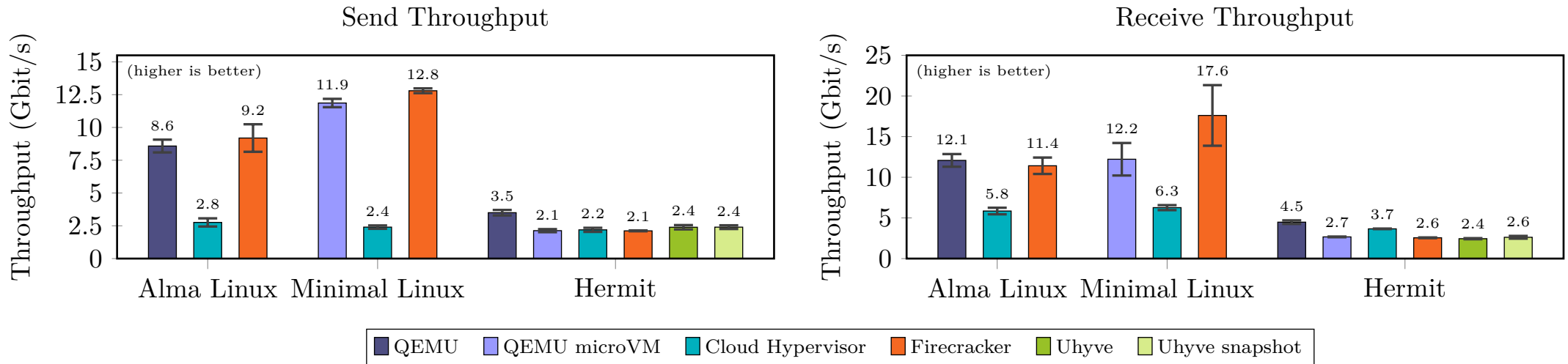
■ Uhyve is **8–16×** (regular) and **32–62×** (template) faster than other VMMs

“Real” Workload: Python + NumPy



- Python runtime dominated by imports (≈ 250 ms ramdisk, ≈ 500 ms virtiofsd, ≈ 220 ms uhyve-fs)
- Uhyve is $\approx 2\times$ (regular) and $\approx 9\times$ (template) faster than other VMMs

Network Throughput



- Linux is faster: path MTU discovery (≈ 65 KiB) vs. Hermit's static (here 1500 byte) MTU
- Uhyve + Hermit is moreless on par with other VMMs + Hermit

Summary

Summary

Contributions

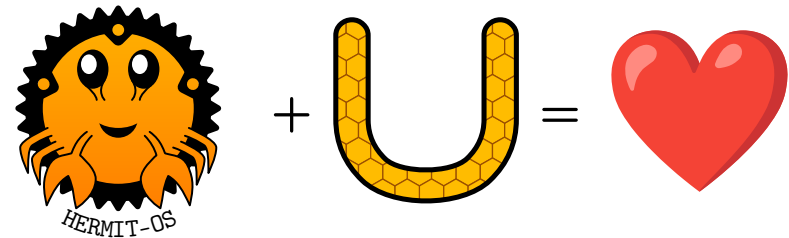
- **Uhyve**: VMM co-designed with and for the Hermit Kernel
- **Application templates**: instruction-accurate snapshots that skip kernel boot and runtime start-up

Future Work

- Integration into serverless frameworks
- More realistic FaaS benchmarks

Results

- 15 ms “Hello World” VM, 4–8× faster than other VMMs
- ≈ 2 ms to first network packet with templates
- 10× faster end-to-end Python workload



Bonus Experiment:

Can you publish a paper on ACM written in Typst instead of LaTeX?

Outcome:

Yes!



Typst (typst.app)



**Funded by
the European Union**

This work has received funding from the Horizon Europe Programmes InterSCADA (Grant Nr. 101173007) and HedgeloT (Grant Nr. 101136216). Views and opinions expressed are, however, those of the authors only and do not necessarily reflect those of the European Union or the European Commission. Neither the European Union nor the granting authority can be held responsible for them.

Thank you for your kind attention!

J. Klimt, S. Simon, M. Kröning, E. Zscheile, P. Vasilopoulos, S. Lankes, A. Monti – jonathan.klimt@eonerc.rwth-aachen.de

Institute for Automation of Complex Power Systems
E.ON Energy Research Center, RWTH Aachen University
Mathieustraße 10
52074 Aachen

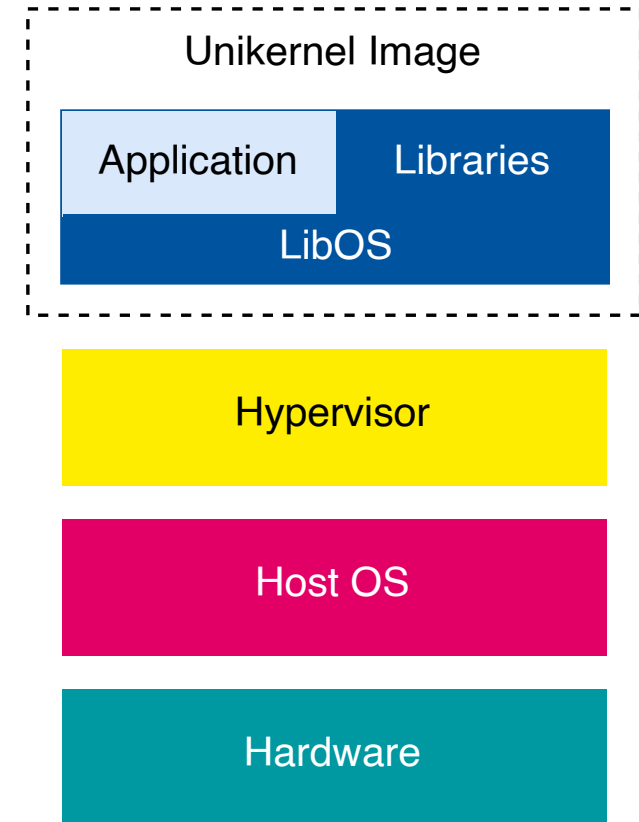
www.acs.eonerc.rwth-aachen.de

Recap: Unikernel

Basic principle: Compile Application + Kernel into single static binary.

Advantages:

- Fast boot
- Small image
- No “OS-Noise”



Related Work

QEMU [6]:

- Realistic emulation of real machines, focus on feature completeness

QEMU microVM [2]:

- Minimal platform, but still a generic OS interface.

Firecracker [4]:

- Lightweight Linux microVMs – generic Linux boot

Cloud Hypervisor [3]:

- General purpose cloud VMM with snapshots – generic Linux boot

NOVA [9]:

- Microhypervisor focused on minimal TCB, not on start-up time

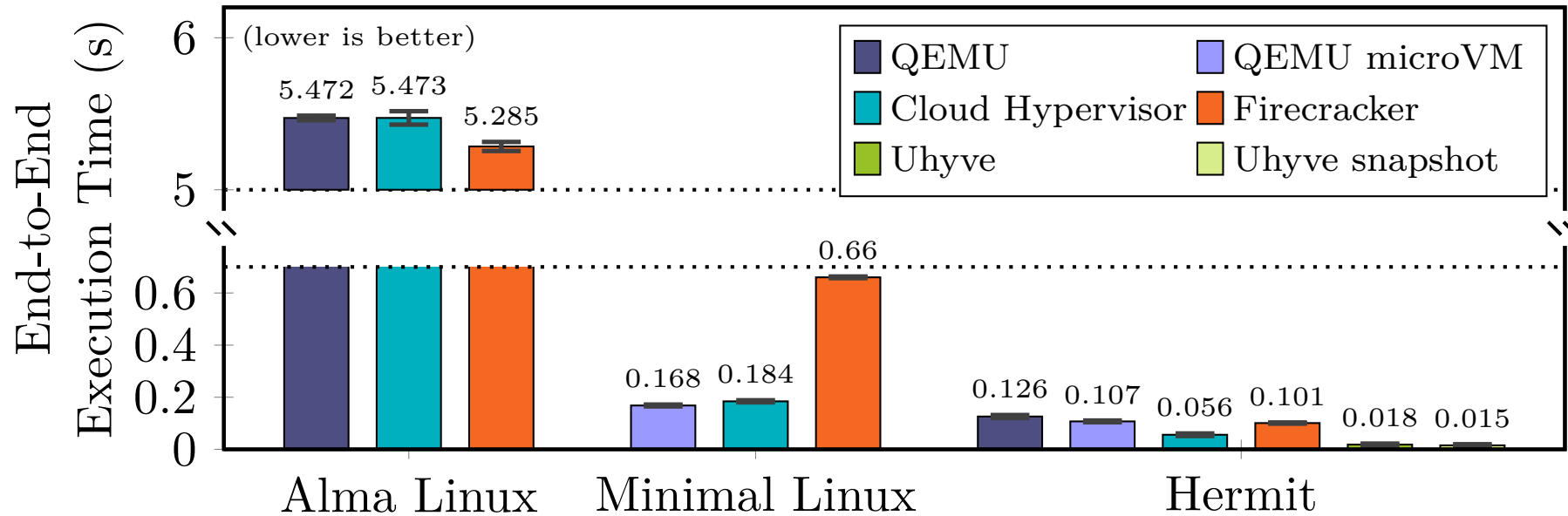
libkrun [1]:

- Library for building VMMs (abstraction over KVM)

UaaF [10], SURE [8]:

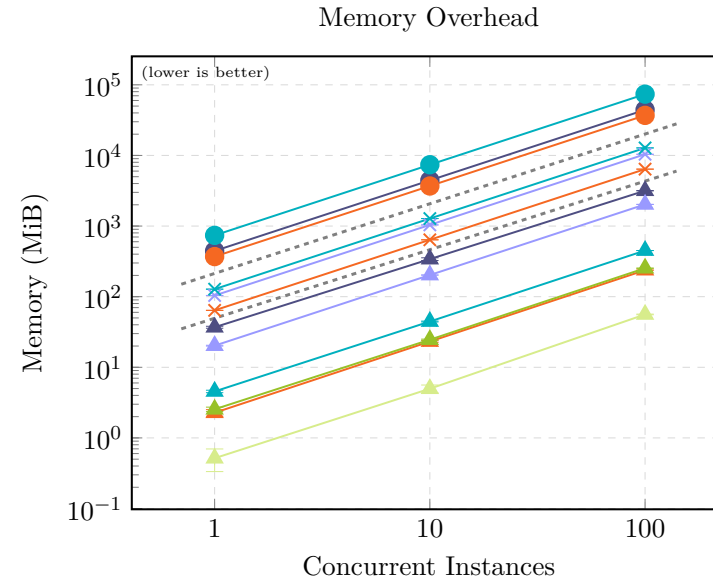
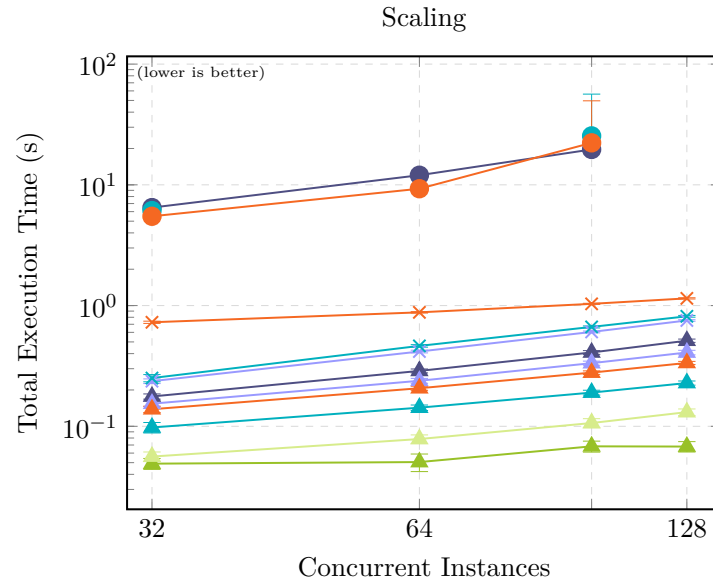
- Unikernels for serverless, but run on general-purpose VMMs

Boot Overhead: "Hello World"



■ 3–7× (regular) and 4–8× (template) faster

Scaling "Hello World"



Alma Linux:	QEMU	Cloud Hypervisor	Firecracker
Minimal Linux:	QEMU microVM	Cloud Hypervisor	Firecracker
Hermit:	QEMU	QEMU microVM	Cloud Hypervisor
		Firecracker	Uhyve
			Uhyve snapshot*

- Unikernel: $\approx 10\times$ less memory, $2\text{--}5\times$ less runtime than Linux
- Templates share memory via CoW: $4\text{--}34\times$ less memory than other VMMs running Hermit

References

- [1] 2026. containers/libkrun. Retrieved May 9, 2026 from <https://github.com/containers/libkrun>
- [2] 'microvm' virtual platform (microvm) — QEMU documentation. Retrieved May 22, 2026 from <https://www.qemu.org/docs/master/system/i386/microvm.html>
- [3] Run Cloud Virtual Machines Securely and Efficiently. Retrieved May 8, 2026 from <https://www.cloudhypervisor.org/>
- [4] Alexandru Agache, Marc Brooker, Andreea Florescu, Alexandra Iordache, Anthony Liguori, Rolf Neugebauer, Phil Piwonka, and Diana-Maria Popa. 2020. Firecracker: Lightweight Virtualization for Serverless Applications. In *17th USENIX Symposium on Networked Systems Design and Implementation (NSDI 20)*, February 2020. USENIX Association, Santa Clara, CA.
- [5] Amazon Web Services Inc. Serverless and Containers - Logical Separation on AWS. Retrieved June 3, 2026 from <https://web.archive.org/web/20260413123157/https://docs.aws.amazon.com/whitepapers/latest/logical-separation/serverless-and-containers.html>
- [6] Fabrice Bellard. 2005. QEMU, a Fast and Portable Dynamic Translator. *Usenix ATC, Freenix Track* (2005), 41–46.
- [7] Nima Nasiri, Nalin Munshi, Simon Daniel Moser, Marius Pirvu, Vijay Sundaresan, Daryl Maier, Thatta Premnath, Norman Böwing, Sathish Gopalakrishnan, and Mohammad Shahrad. 2026. In-Production Characterization of an Open Source Serverless Platform and New Scaling Strategies. In *Proceedings of the 21st European Conference on Computer Systems (EUROSYS '26)*, April 26, 2026. Association for Computing Machinery, New York, NY, USA, 2054–2072. <https://doi.org/10.1145/3767295.3769377>
- [8] Federico Parola, Shixiong Qi, Anvaya B. Narappa, K. K. Ramakrishnan, and Fulvio Risso. 2024. SURE: Secure Unikernels Make Serverless Computing Rapid and Efficient. In *Proceedings of the ACM Symposium on Cloud Computing*, November 20, 2024. ACM, Redmond WA USA, 668–688. <https://doi.org/10.1145/3698038.3698558>
- [9] Udo Steinberg and Bernhard Kauer. 2010. NOVA: a microhypervisor-based secure virtualization architecture. In *Proceedings of the 5th European conference on Computer systems (EuroSys '10)*, April 13, 2010. Association for Computing Machinery, New York, NY, USA, 209–222. <https://doi.org/10.1145/1755913.1755935>
- [10] Bo Tan, Haikun Liu, Jia Rao, Xiaofei Liao, Hai Jin, and Yu Zhang. 2020. Towards Lightweight Serverless Computing via Unikernel as a Function. In *2020 IEEE/ACM 28th International Symposium on Quality of Service (IWQoS)*, June 2020. 1–10. <https://doi.org/10.1109/IWQoS49365.2020.9213020>
- [11] Liang Wang, Mengyuan Li, Yinqian Zhang, Thomas Ristenpart, and Michael Swift. 2018. Peeking Behind the Curtains of Serverless Platforms. 2018. 133–146. Retrieved June 3, 2026 from <https://www.usenix.org/conference/atc18/presentation/wang-liang>