API Remoting-based GPU Support for Unikernels

Martin Kröning, Niklas Eiling, Stefan Lankes, Antonello Monti

SPONSORED BY THE

Federal Ministry of Education and Research

Published as *GPU Acceleration in Unikernels Using Cricket GPU Virtualization* at the 13th International Workshop on Runtime and Operating Systems for Supercomputers (ROSS) at SC23:

- sc23.supercomputing.org/proceedings/workshops/workshop_pages/ws_ross103.html
- doi.org/10.1145/3624062.3624236

ACS | Automation of Complex Power Systems

E.ON Energy Research Center

RWTH AACHEN UNIVERSITY

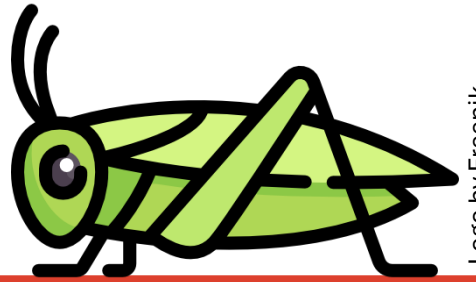
Motivation

Status Quo

- GPUs are crucial for HPC
- Unikernels are emerging (Cloud)
  - Convergent Computing
  - Lower overhead
  - Lower complexity

Goal

- Let's use GPUs from Unikernels!
- Challenge: GPU drivers are proprietary and complex
- Driver porting is infeasible



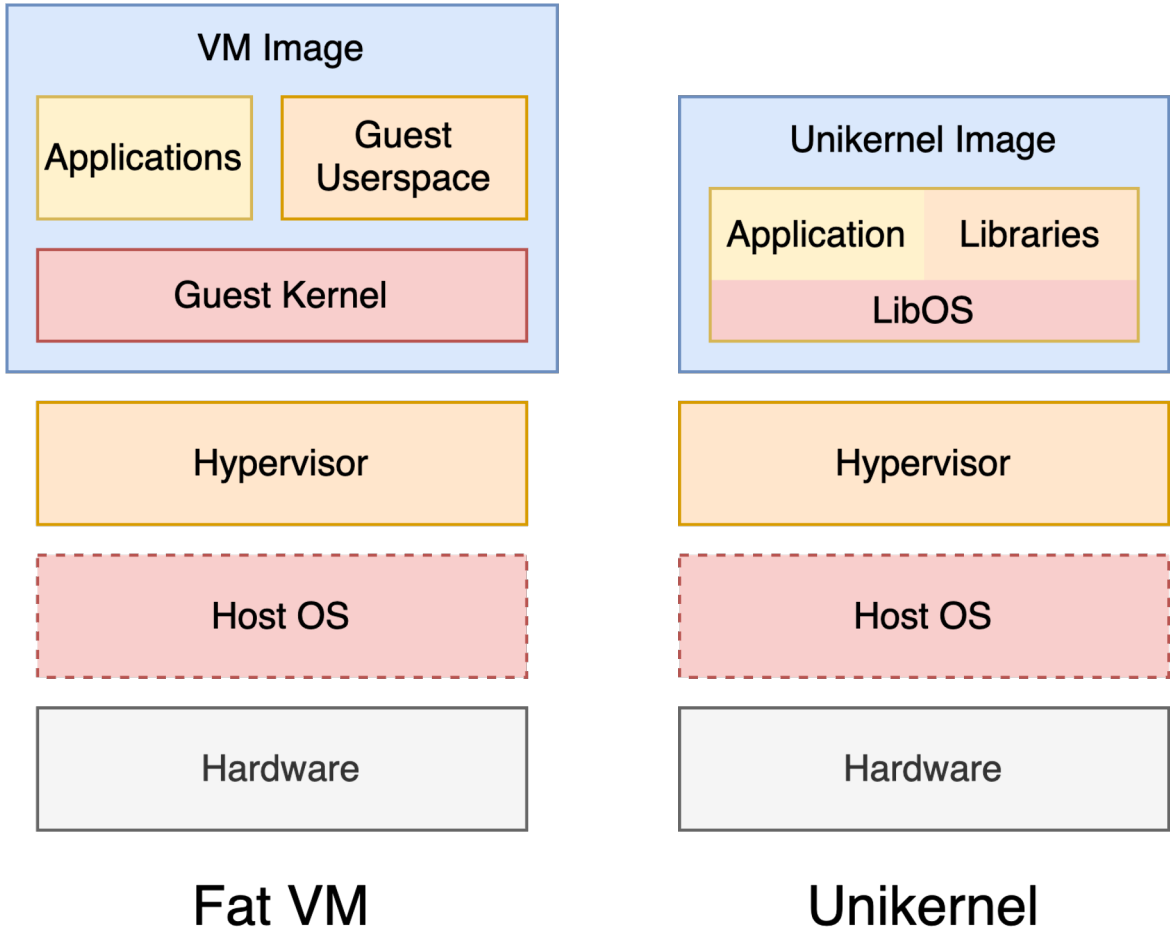
Logo by Freepik

Solution

- Port Cricket GPU Virtualization

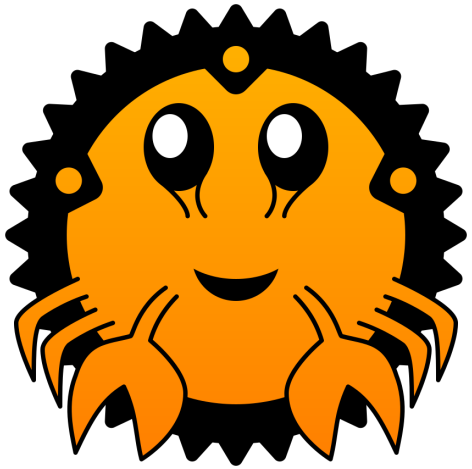
Unikernels

- Specialized for use cause
  - Tiny images
- One process per image
  - No isolation necessary
- Single address space operating system
  - No address space context switch
- Single privilege level
  - No privilege context switch
- System calls are just function calls



Hermit

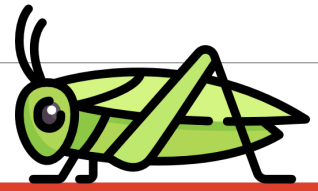
- In-House research OS
- Written in Rust
- Custom Syscall Interface



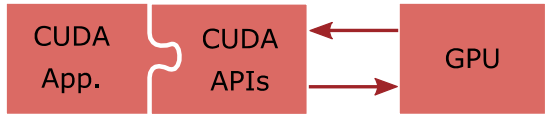
Unikraft

- Commercialized FOSS project
- Written in C
- Linux Syscall Interface

Cricket: A Virtualization Layer for GPUs



API Remoting



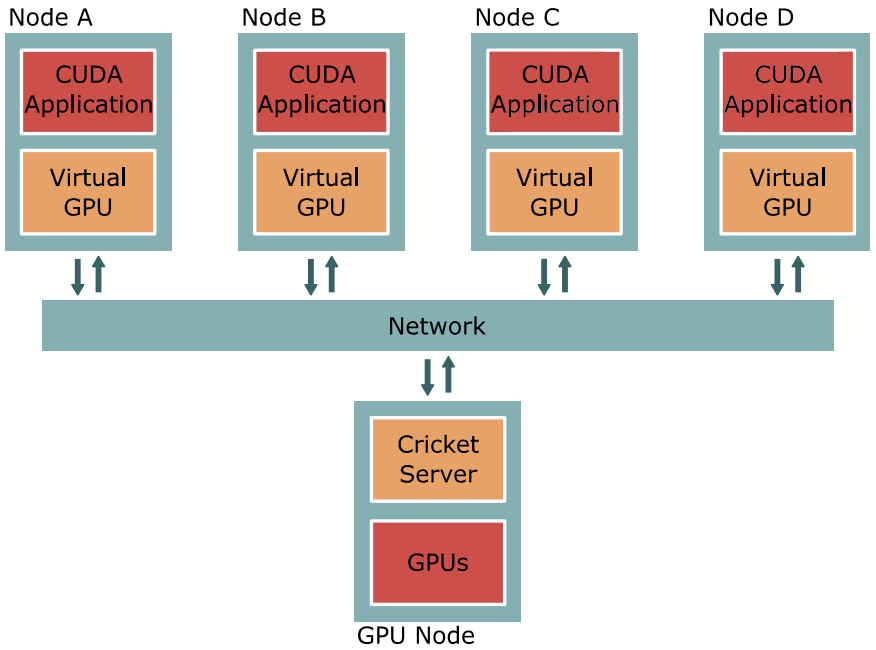
(a) GPU application without virtualization

(b) GPU application with virtualization layer

- Separates proprietary device dependent code into separate process
- Allows full control of device interactions
  - Control, manipulation and limiting the use of GPU resources
- Low virtualization overhead

Use Cases

Remote execution, Scheduling, Monitoring



Niklas Eiling, Stefan Lankes, and Antonello Monti
An Open-Source Virtualization Layer for CUDA Applications (2020)

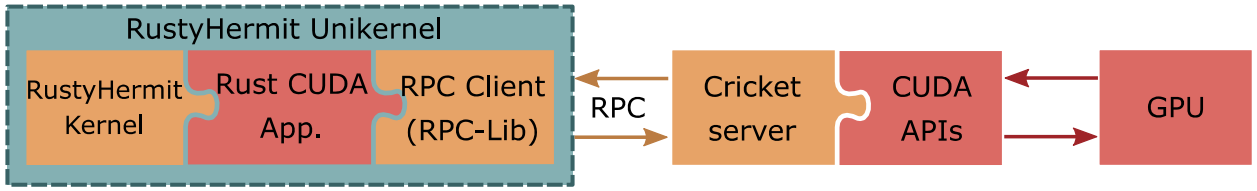
Adapting Cricket for Unikernels

API Remoting

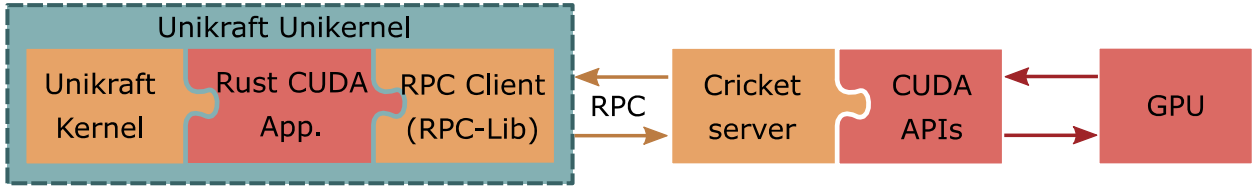
- Cricket is based on ONC RPCs
- Open-Source C implementation is old and complex
- For Unikernels: new Rust implementation of RPC client
  - Simpler OS Interface than C Impl.
  - Hermit: Client side is rust-only
- All user code is run inside Unikernel
- Only CUDA APIs are run outside

Cricket for Unikernels

(a) Cricket with C client



(b) Cricket with Hermit client

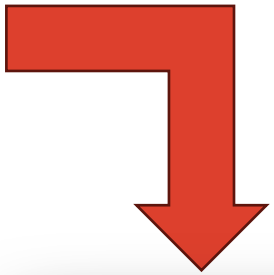


(c) Cricket with Unikraft client

Rust Procedural Macros Generate Remote Procedure Calls

github.com/RWTH-ACS/RPC-Lib

```
program MATH {
    version VERS_1 {
        int ADD(int, int) = 1;
    } = 1;
} = 67908;
```



```
#[include_rpcl("math.x")]
struct RpcClient;

let mut client = RpcClient::new("127.0.0.1")?;

let res = client.ADD(&1, &2)?;

assert!(result == 3);
```

CUDA in Rust

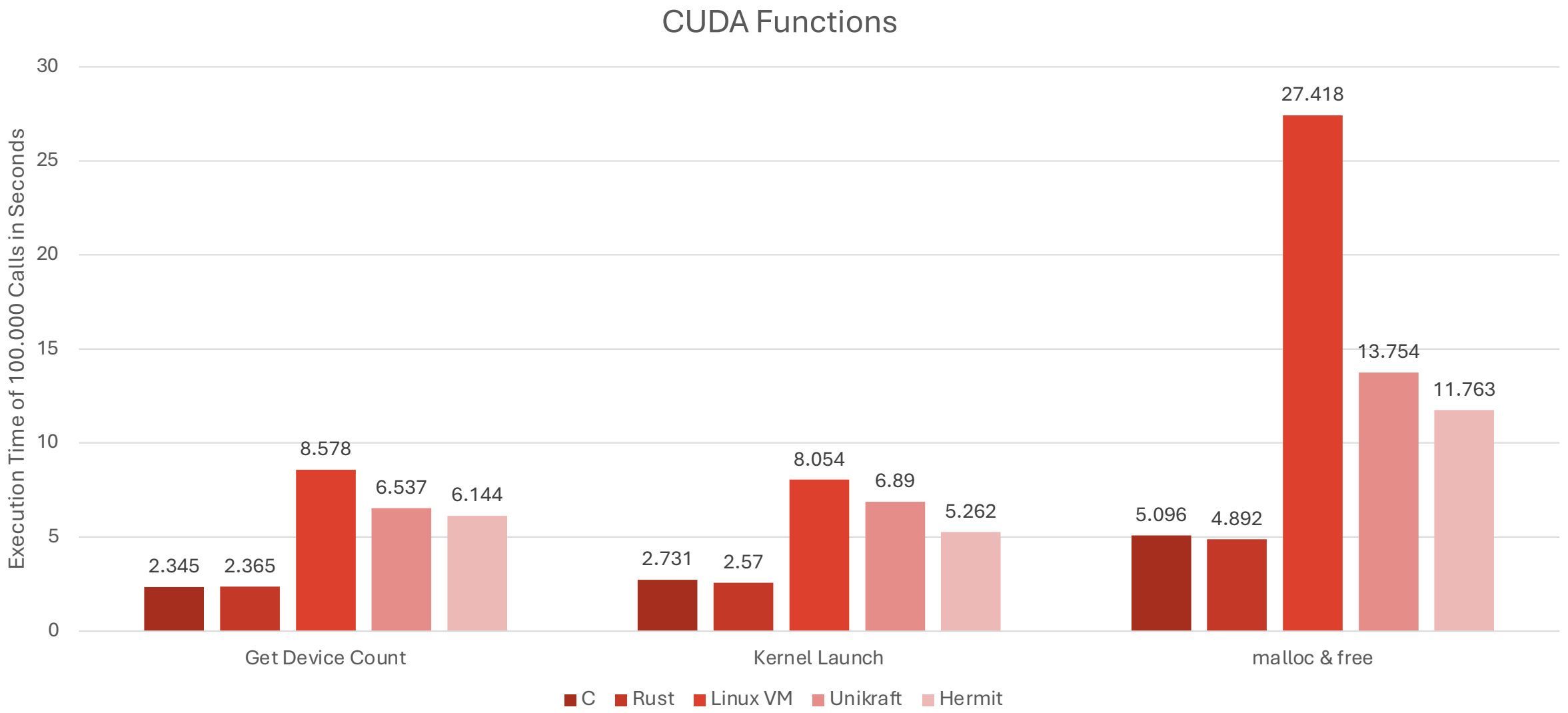
git.rwth-aachen.de/acs/public/virtualization/rpc-lib/cricket-rs

```rust
let mut cricket = Cricket::connect("10.0.0.2")?;
cricket.cuda_set_device(0)?;

let host_mem = vec![0; 536_870_912];
let mut device_mem = cricket.cuda_malloc(host_mem.len());
cricket.cuda_memcpy_htod(&host_mem, &mut device_mem)?;

// Do some work

cricket.cuda_free(device_mem)?;
```

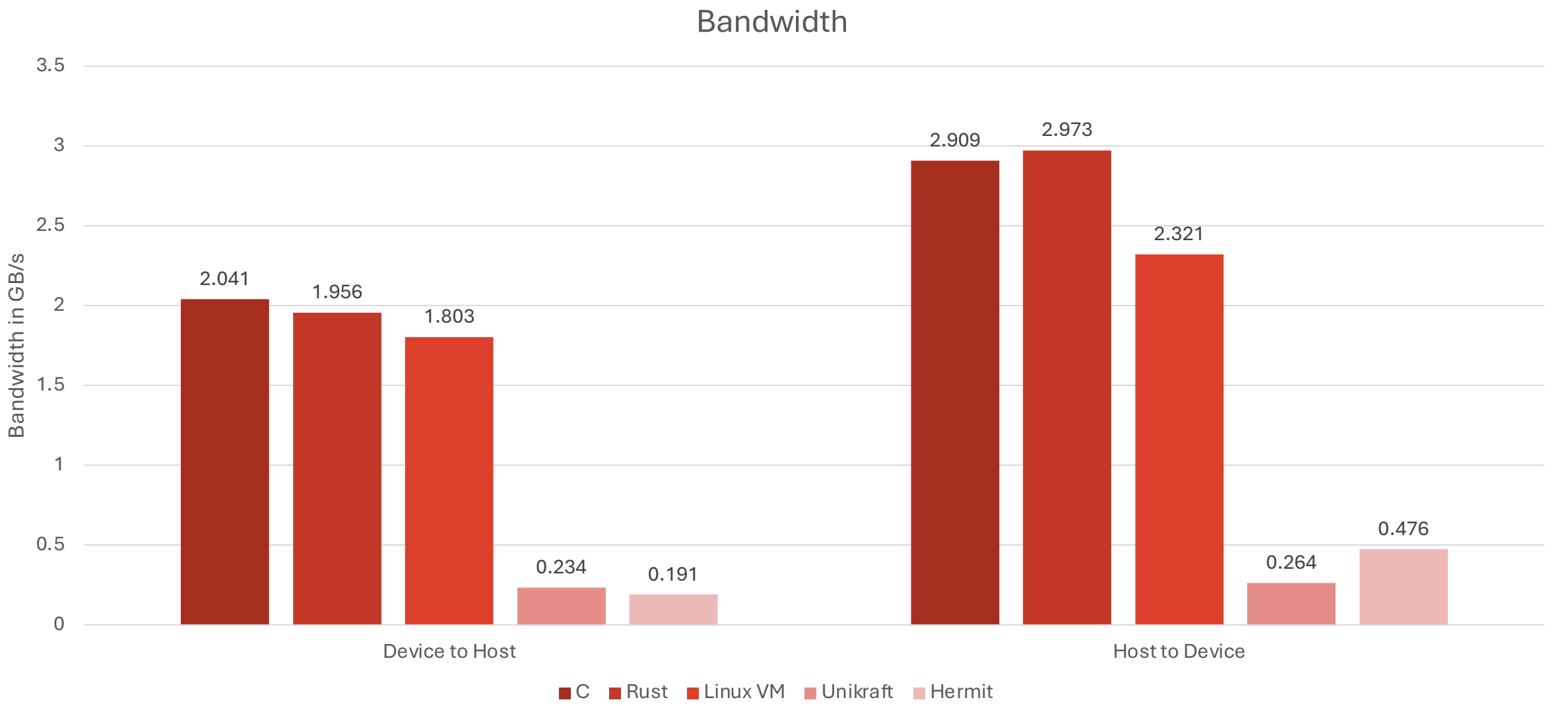
Benchmark Setup

Name	App Lang.	OS	Hypervisor	Network
C	C	Rocky Linux	-	Native (mlx5_core)
Rust	Rust	Rocky Linux	-	Native (mlx5_core)
Linux VM	Rust	Fedora	QEMU/KVM	virtio-net-pci with vhost
Unikraft	Rust	Unikraft	QEMU/KVM	virtio-net-pci with vhost
Hermit	Rust	Hermit	QEMU/KVM	virtio-net-pci with vhost

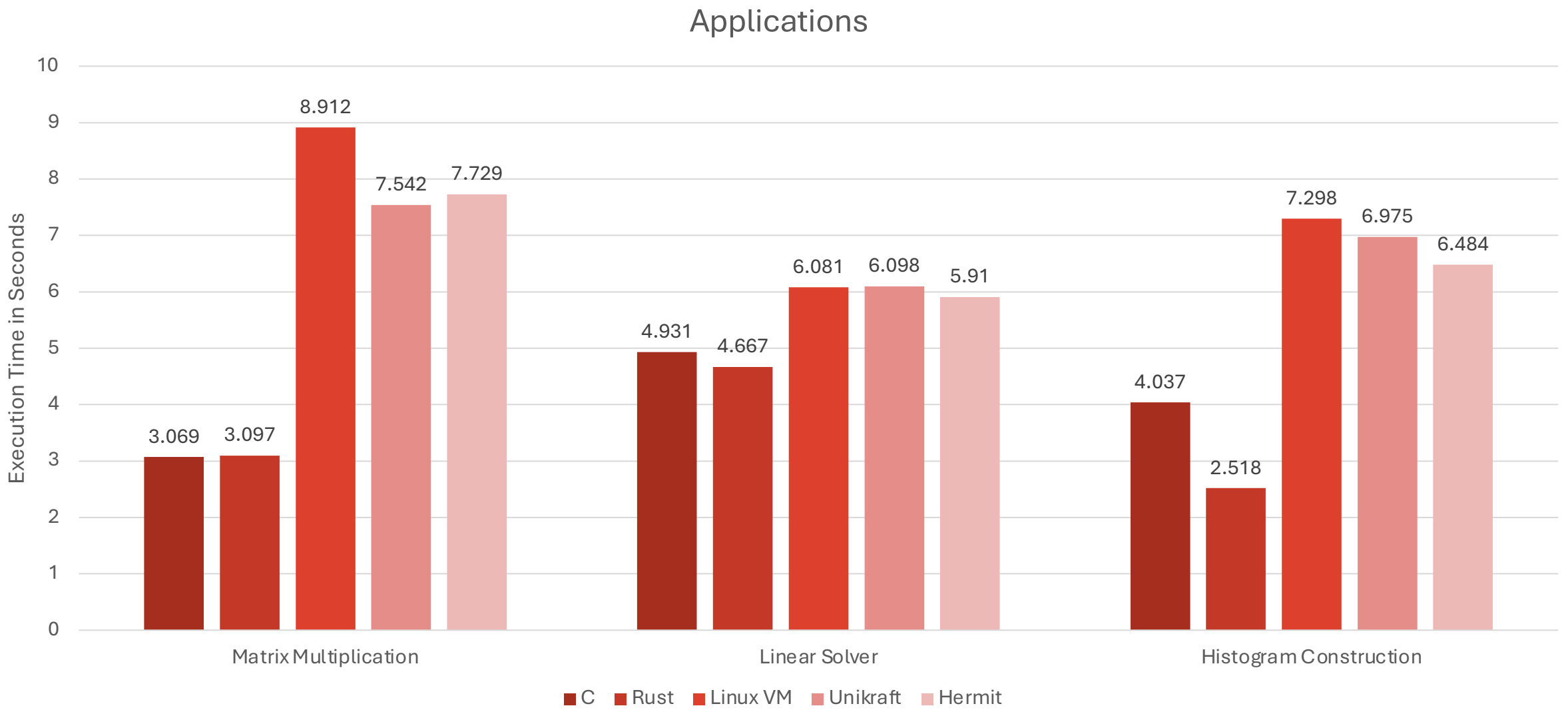
Microbenchmarks



Microbenchmarks



CUDA Samples Benchmarks

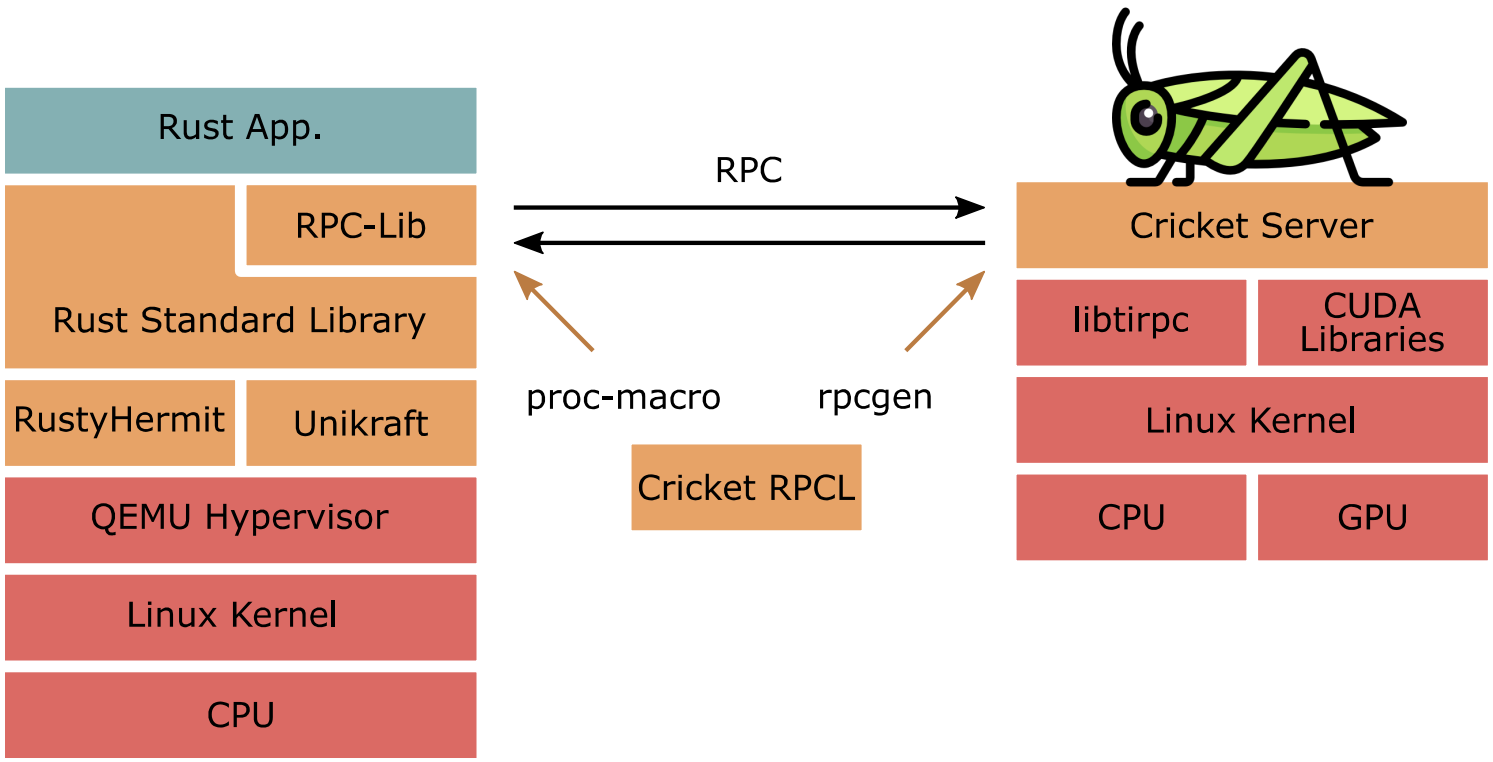


Conclusion

- 👍 RPC based GPU access is feasible
- 👎 Low bandwidth with Unikernels

Current Work

- General virtio driver improvements
- Implement `VIRTIO_NET_F_MQ`
- Parallelize Network Stack
- Use DPDK for Host Kernel Bypass
- Use vDPA for Data Path Acceleration



Questions?

Thank you for your kind attention!

Check us out GitHub: hermit-os.org
Come say hi on Zulip: hermit.zulipchat.com

Martin Kröning – martin.kroening@eonerc.rwth-aachen.de

Institute for Automation of Complex Power Systems
RWTH Aachen University
Mathieustraße 10
52074 Aachen

acs.eonerc.rwth-aachen.de