News from the Hermit Crab

From Soundness Foundations to GPU Virtualization

Martin Kröning
Agenda

1. Introduction to Hermit
2. Interesting Internals
3. GPU Virtualization with Cricket
4. Application & Kernel Profiling
Introduction to Hermit
Unikernels

Standard VM

**VM Image**
- Applications
- Guest Userspace
- Guest Kernel

**Hypervisor**

**Host OS**

**Hardware**

**Unikernel Image**

**Application**

**Libraries**

**Host OS**

**Hypervisor**

**Host OS**

**Hardware**

**Container Image**

- Applications
- Guest Userspace

**Container Runtime**

**Host OS**

**Hardware**

**Container**

**Unikernels**

News from the Hermit Crab
2024-02-03 | Martin Kröning | ACS
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
The Hermit Operating System

Overview
- Written in Rust
- Official tier 3 Rust target for Rust applications
- GCC + Newlib fork for C applications

Features
- Multi-core support
- Easily configurable
- **New**: Compiles on Windows
- **New**: Supports stable Rust through [rust-std-hermit](https://example.com)
Interesting Internals
Hermit Image

Unikernel Image

User Target
*
unknown-hermit

Application
Crates
std
alloc
core

Kernel Target (softfloat)
x86_64-unknown-none
aarch64-unknown-none-softfloat
riscv64gc-unknown-none-elf

hermit-builtins
libm
mem-intrinsics
core

Hermit Kernel
libhermit.a

hermit-kernel
Crates
alloc
core
On the Challenge of Sound Code for Operating Systems
DOI: 10.1145/3623759.3624554

Goal: Soundness—Safety must not require context!

**hermit-sync**
- SpinMutex
- OnceCell
- Lazy
- TakeStatic
- InterruptMutex
- InterruptRefCell

**count-unsafe**
- Counts unsafe functions, expressions, implementations, etc.
An Evolving Network Stack

- **2022**: Moved from user space into kernel space
- **2023**: Implemented support for BSD-style sockets
- **2024**: Add poll support for async I/O
GPU Virtualization with Cricket
GPU Virtualization with Cricket

github.com/RWTH-ACS/cricket

API Remoting

![Diagram](a) GPU application without virtualization

- Separates proprietary device dependent code into separate process
- Allows full control of device interactions
- Low virtualization overhead

![Diagram](b) GPU application with virtualization layer

Use Cases

Remote execution, scheduling, monitoring
Adapting Cricket for Unikernels

GPU Acceleration in Unikernels
Using Cricket GPU Virtualization
DOI: 10.1145/3624062.3624236

API Remoting
- Cricket is based on ONC RPCs
- Reference C impl is old and complex
  - Uses Linux-specific network features
- For unikernels: New Rust impl
- All user code is run inside unikernel
- Only CUDA APIs are run outside
Application & Kernel Profiling
Profiling through Instrumentation with rftrace

github.com/hermit-os/rftrace/tree/next

Rust Source

```rust
fn square(x: i32) -> i32 {
    x * x
}
```

Assembly

```assembly
square:
    mov   eax, edi
    imul  eax, edi
    ret
```

Instrumented Assembly

```assembly
square:
    push  rbp
    mov   rbp, rsp
    call  mcount
    imul  edi, edi
    mov   eax, edi
    pop   rbp
    ret
```
Hermit Image with rftrace

Unikernel Image

User Target
*--unknown-hermit

Application
Crates
std
alloc
core

Kernel Target (softfloat)
x86_64-unknown-none
aarch64-unknown-none-softfloat
riscv64gc-unknown-none-elf

Hermit Kernel
libhermit.a

rftrace-backend
core

rftrace
rftrace.a
Trace Replay

```
[ 1] | hello_world::test1() {
0.150 us [ 1] |   core::fmt::Arguments::new_const();
[ 1] |   sys_write() {
 [ 1] |     hermit::hermit::arch::x86_64::kernel::switch::kernel_function3() {
0.136 us [ 1] |       hermit::core::ptr::write();
0.130 us [ 1] |       hermit::core::ptr::write();
0.134 us [ 1] |       hermit::core::ptr::write();
 [ 1] |       hermit::hermit::arch::x86_64::kernel::CoreLocal::get() {
0.134 us [ 1] |         hermit::x86_64::addr::VirtAddr::zero();
 [ 1] |         hermit::x86_64::registers::GsBase::read() {
0.258 us [ 1] |           x86_64::registers::model_specific::Msr::read();
 [ 1] |         hermit::x86_64::addr::VirtAddr::new() {
```
Trace Visualization
Acknowledgments

SPONSORED BY THE

Federal Ministry of Education and Research

Funded by the European Union
Thank you for your kind attention!

Check us out GitHub: [github.com/hermit-os](https://github.com/hermit-os)
Come say hi on Zulip: [hermit.zulipchat.com](https://hermit.zulipchat.com)

**Martin Kröning** – [martin.kroening@eonerc.rwth-aachen.de](mailto:martin.kroening@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](http://www.acs.eonerc.rwth-aachen.de)