

Linux Plumbers Conference

Vienna, Austria | September 18-20, 2024



OpenHCL: A Linux and Rust based paravisor

Chris Oo – Microsoft



LINUX
PLUMBERS
CONFERENCE Vienna, Austria / Sept. 18-20, 2024



What is a paravisor?

- Firmware component that runs inside the guest at a higher privilege level
- Provide emulation for unenlightened guests on CVM platforms
 - APIC emulation and interrupt virtualization
- Provide services for guests
 - vTPM
 - Legacy emulated devices like serial
 - Device translation



Why have a paravisor?

- Run guests that are not fully enlightened such as Windows and older Linux
- Provide security isolation for guests that are not fully hardened
- Provide emulated devices such as vTPM, serial
- Provide device translation
 - Translate NVMe to paravirtualized storage
- Host debuggers and diagnostic processes
 - Allow debugging guests where traditional debuggers are hard like CVMs



Why run vmm code in a paravisor?

- Move emulated devices inside into the guest
 - Require guest to host compromise outside of compromising an emulated device
- Support legacy OSes with accelerated devices assigned to a guest
 - Translate an assigned NVMe device to emulated IDE inside the paravisor
- Share confidential and non-confidential architecture
 - Run the same VMM in the same environment for both



OpenHCL overview

- Linux and usermode Rust based paravisor from the OpenVMM project
- Runs at a higher privilege level
 - VTL2 on Hyper-V
 - L1 VM on Intel TDX
 - VMPL0 on AMD SEV-SNP
- Provides various services to the guest
 - CVM enlightenments and support
 - Emulated device support
 - Device translation such as from NVMe to paravirt storage
 - Diagnostics



OpenHCL features

- Supports Hyper-V isolation (VTLs) on x86-64 and ARM64, AMD SEV-SNP, Intel TDX
- Supports device translation
- Supports vTPM
- Supports various legacy device emulators such as serial
- Supports Hyper-V legacy bios, Hyper-V UEFI and Linux direct boot guests



Usage in Azure

- Used in new Azure Boost SKUs
- Meets storage and networking performance requirements
- Used in over 10M cores and counting



Design philosophy

- Track upstream kernel
 - Aim to upstream all kernel patches or have a path to upstream
- Do as much in usermode as possible
 - Host the VMM itself in usermode
 - Device drivers in usermode
- Do as much in safe idiomatic rust as possible
- Rust async-focused usermode VMM
- Keep VMM code OS agnostic
 - Allows for running outside of OpenHCL



Why Rust?

- Prevent whole classes of memory safety issues
 - Borrow checker
 - Send & Sync traits
- Modern language with ecosystem of useful crates
 - Traits
 - Async
 - Modern tooling with rust-analyzer
- Still provides enough low level control
 - Able to use C APIs and talk directly with hardware



Why async Rust?

- Support a variety of different devices, especially high performance devices
- Control execution inside different environments
 - The VMM run in different environments, so being able to tailor executors without needing to rewrite device code
- Control OS scheduling overhead
 - Executor could be multithreaded, or singlethreaded or whatever works best for that environment

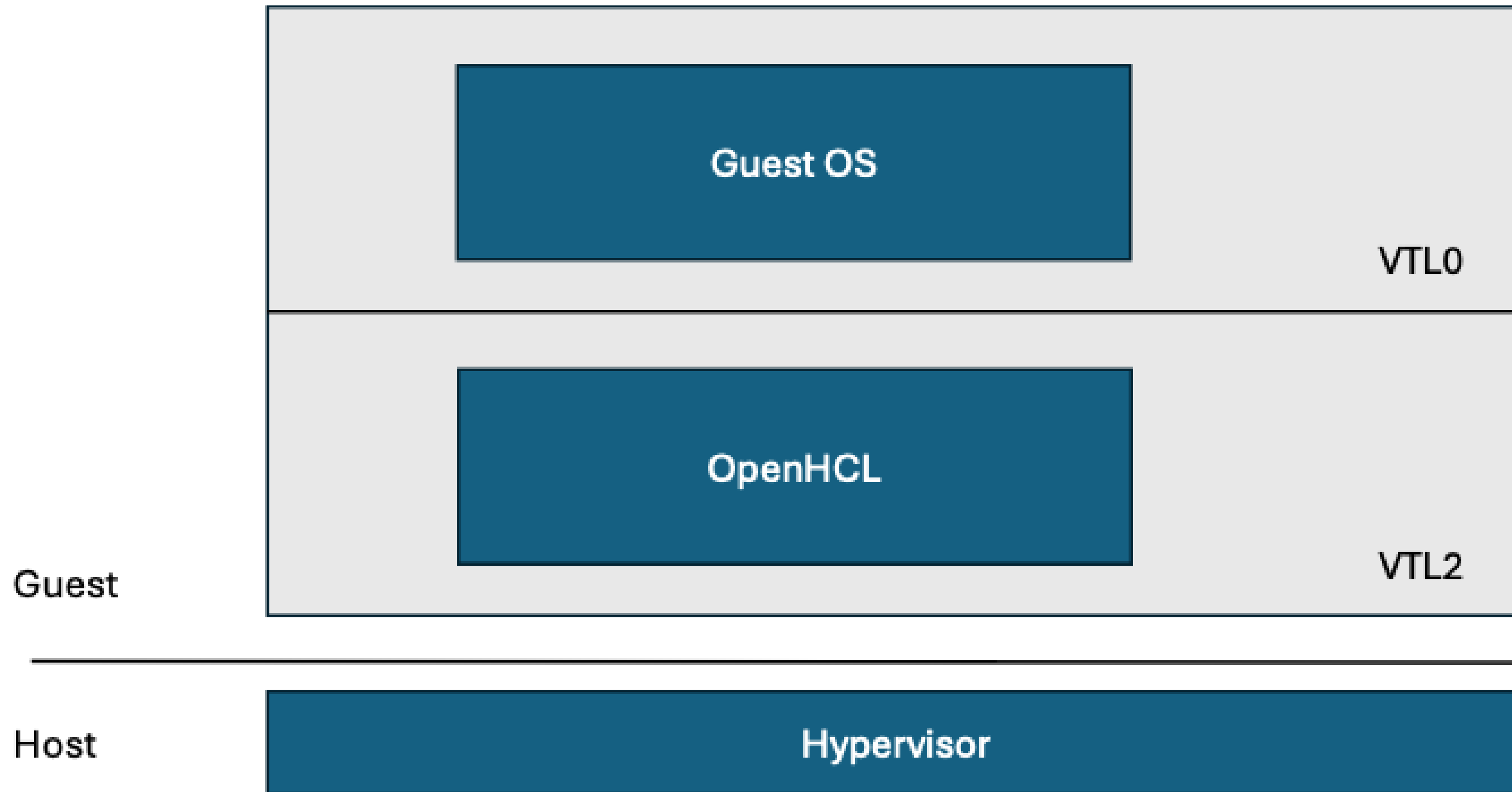


Async Rust in OpenHCL

- Minimize paravisor runtime by minimizing OS threads
- Utilize per vcpu executors for most devices
 - Each thread affinitized to a single vcpu
 - Minimize context switch overhead between VMM exit handler and devices
 - Very important for device translation, where OpenHCL is in the hotpath
- Some low performance devices and diagnostic services handled on separate executor thread



OpenHCL VM



OpenHCL architecture



Why use Linux?

- Write std Rust for VMM code
 - no_std Rust is especially difficult to code in
 - We want to use stable Rust toolchains, not nightly for some no_std features
- Supports standard tooling like gdbserver, perf, etc
- Write usermode drivers via VFIO
- Broadly supported Rust toolchain and crate ecosystem
- Familiar OS platform for contributors



Linux Kernel details

- Boot via device tree, no ACPI
- Minimal Kconfig
 - Minimize binary size and runtime RAM usage
 - Device drivers in usermode via VFIO
- mshv_vtl driver
 - Provides virtualization APIs for usermode
 - Provides access to physical addresses via mmap
- Otherwise standard APIs used by usermode VMM



Usermode processes

- Packaged as a single binary with multiple aliases and launch names
- openhcl_init
- openhcl_dump
- openhcl_crash
- VMM processes
 - control & diagnostics server
 - main vmm worker



openhcl_init

- Lightweight and simple init process to set various settings and launch other processes
 - Minimize binary size and complexity vs other options
- Setup kmsg logging
- Launches main VMM process next



openvmm (diag server)

- Main control process
- Launches child vmm worker that handles exits from the guest
- Handles diagnostics requests via ohcldiag-dev
 - ohcldiag-dev runs on the host, and communicates with diag_server via VSOCK



Example diagnostic commands

- kmsg
 - Dump the current kmsg log in OpenHCL
- inspect
 - Objects implement the Inspect trait throughout the codebase
 - Allows inspection of object & system state via human readable text
 - Some values modifiable at runtime, such as tracing log filter
- shell
 - Remote shell into OpenHCL, useful for interactive development
 - Provided via BusyBox



ohcldiag-dev demo

```
gdb fish x bash bash
xsave: _,
xsave_state_bv_broken: false,
}
hvlite [?] user/cho/whp-debug-exception [v1.81.0]
> cargo run --target x86_64-pc-windows-msvc -p ohcldiag-dev -- 'c:\users\cho\appdata\local\temp\uhdiag' inspect vm/partition/caps
Finished `dev` profile [unoptimized + debuginfo] target(s) in 0.28s
Running `target/x86_64-pc-windows-msvc/debug/ohcldiag-dev.exe 'c:\users\cho\appdata\local\temp\uhdiag' inspect vm/partition/caps`
{
  can_freeze_time: false,
  cet: true,
  cet_ss: true,
  dr6_tsx_broken: false,
  hv1: true,
  hv1_reference_tsc_page: true,
  nxe_forced_on: false,
  reset_rdx: 0xa00f11,
  sgx: false,
  tsc_aux: true,
  vendor: "AuthenticAMD",
  x2apic: true,
  x2apic_enabled: false,
  xsave: _,
  xsave_state_bv_broken: false,
}
hvlite [?] user/cho/whp-debug-exception [v1.81.0]
> cargo run --target x86_64-pc-windows-msvc -p ohcldiag-dev -- 'c:\users\cho\appdata\local\temp\uhdiag' inspect vm/partition
Finished `dev` profile [unoptimized + debuginfo] target(s) in 0.28s
Running `target/x86_64-pc-windows-msvc/debug/ohcldiag-dev.exe 'c:\users\cho\appdata\local\temp\uhdiag' inspect vm/partition`
{
  caps: _,
  clear_halt: false,
  cpuid: _,
  dependencies: "vmtime,chipset",
  enable_vtl_protection: false,
  enter_modes: _,
  halt_count: 0,
  irq_routes: _,
  lower_vtl_memory_layout: _,
  monitor_page: _,
  no_sidecar_hotplug: false,
  power_state: "running",
  software_devices: _,
  topology: _,
  unit_state: "running",
  use_mmio_hypercalls: false,
  vp: _,
}
hvlite [?] user/cho/whp-debug-exception [v1.81.0]
[ 14.350799] Run /init as init process
[ 14.382577] with arguments:
[ 14.406438] /init
[ 14.424675] nokasrl
[ 14.444129] with environment:
[ 14.469838] HOME=/
[ 14.488979] TERM=linux
[ 14.515951] mount (62) used greatest stack depth: 14232 bytes left
Root device '/dev/sda3' does not exist. Dropping to a shell.
sh: can't access tty; job control turned off
/ # 109.939774500s INFO vmbus_server::channels: sending offer to guest channel_id=0x6 connection_id=0x22006 key={00000001-facb-11e6-bd58-64006a7986d3}-{3029aa91-03dd-4319-9563-4eae65aef343}-0
109.940252500s INFO vmbus_server::channels: new channel offer_id=OfferId(5) key={00000001-facb-11e6-bd58-64006a7986d3}-{3029aa91-03dd-4319-9563-4eae65aef343}-0 confidential=true
109.997179600s INFO vmbus_server::channels: opened channel offer_id=0x5 channel_id=0x6 result=0
110.066796100s INFO vmbus_server: revoking channel offer_id=OfferId(5)
110.067141000s INFO vmbus_server::channels: rescinding channel from guest channel_id=0x6
116.624617800s INFO vmbus_server::channels: sending offer to guest channel_id=0x6 connection_id=0x22006 key={00000001-facb-11e6-bd58-64006a7986d3}-{002411d9-3001-4060-9917-b57e4543e4fb}-0
116.625157800s INFO vmbus_server::channels: new channel offer_id=OfferId(5) key={00000001-facb-11e6-bd58-64006a7986d3}-{002411d9-3001-4060-9917-b57e4543e4fb}-0 confidential=true
116.626585500s INFO vmbus_server::channels: opened channel offer_id=0x5 channel_id=0x6 result=0
116.688860300s INFO vmbus_server: revoking channel offer_id=OfferId(5)
116.689165100s INFO vmbus_server::channels: rescinding channel from guest channel_id=0x6
122.951511100s INFO vmbus_server::channels: sending offer to guest channel_id=0x6 connection_id=0x22006 key={00000001-facb-11e6-bd58-64006a7986d3}-{27116efb-988e-4e88-9185-67ee621d7ab1}-0
122.951975500s INFO vmbus_server::channels: new channel offer_id=OfferId(5) key={00000001-facb-11e6-bd58-64006a7986d3}-{27116efb-988e-4e88-9185-67ee621d7ab1}-0 confidential=true
122.953322200s INFO vmbus_server::channels: opened channel offer_id=0x5 channel_id=0x6 result=0
123.025475900s INFO vmbus_server: revoking channel offer_id=OfferId(5)
123.025826200s INFO vmbus_server::channels: rescinding channel from guest channel_id=0x6
147.189113000s INFO vmbus_server::channels: sending offer to guest channel_id=0x6 connection_id=0x22006 key={00000001-facb-11e6-bd58-64006a7986d3}-{97c462fd-c7d1-4055-a922-2d3893100c5a}-0
147.189557000s INFO vmbus_server::channels: new channel offer_id=OfferId(5) key={00000001-facb-11e6-bd58-64006a7986d3}-{97c462fd-c7d1-4055-a922-2d3893100c5a}-0 confidential=true
147.190803500s INFO vmbus_server::channels: opened channel offer_id=0x5 channel_id=0x6 result=0
147.255105500s INFO vmbus_server: revoking channel offer_id=OfferId(5)
147.255409700s INFO vmbus_server::channels: rescinding channel from guest channel_id=0x6
154.862583100s INFO vmbus_server::channels: sending offer to guest channel_id=0x6 connection_id=0x22006 key={00000001-facb-11e6-bd58-64006a7986d3}-{deddd7fb-bb64-474f-ae2e-08de7ace10ad}-0
154.864236900s INFO vmbus_server::channels: opened channel offer_id=0x5 channel_id=0x6 result=0
154.927287500s INFO vmbus_server: revoking channel offer_id=OfferId(5)
154.927621700s INFO vmbus_server::channels: rescinding channel from guest channel_id=0x6
```



gdbstub demo

```
gdb x bash x bash
0xffffffff814fffae    cmp    %rbx,%r13
> 0xffffffff814fffb1    jne    0xffffffff814fff89
0xffffffff814fffb3    pop    %rbx
0xffffffff814fffb4    pop    %rbp
0xffffffff814fffb5    pop    %r12
0xffffffff814fffb7    pop    %r13
0xffffffff814fffb9    jmp    0xffffffff81c02000
0xffffffff814fffbe    xchg  %ax,%ax
0xffffffff814fffc0    push  %r13
0xffffffff814fffc2    mov   %rdx,%r13
0xffffffff814fffc5    mov   $0x5,%edx
0xffffffff814ffcca    push  %r12
0xffffffff814ffccc    mov   %rsi,%r12
0xffffffff814ffccf    mov   $0xffffffff820c86ea,%rsi
0xffffffff814ffcd6    push  %rbp
0xffffffff814ffcd7    mov   %rdi,%rbp
0xffffffff814ffdda    push  %rbx
0xffffffff814ffddb    mov   %rcx,%rbx
0xffffffff814ffdde    call  0xffffffff8193bf50
0xffffffff814ffde3    test  %eax,%eax
0xffffffff814ffde5    jne   0xffffffff81500029
0xffffffff814ffde7    add   $0x5,%rbp
0xffffffff814ffdeb    mov   $0x2,%eax
0xffffffff814ffdf0    mov   %al,(%r12)
0xffffffff814ffdf4    xor   %edx,%edx
0xffffffff814ffdf6    xor   %esi,%esi
0xffffffff814ffdf8    mov   %rbp,%rdi
0xffffffff814ffdfb    call  0xffffffff81940560
0xffffffff81500000    mov   $0x2c,%esi
0xffffffff81500005    mov   %rbp,%rdi
0xffffffff81500008    mov   %rax,0x0(%r13)
0xffffffff8150000c    call  0xffffffff8193bf80
0xffffffff81500011    cmp   $0x1,%rax
0xffffffff81500015    sbb  $0xffffffffffffffff,%rax

remote Thread 1.1 In: L?? PC: 0xffffffff814fffb1
(gdb) si
0xffffffff81c02000 in ?? ()
(gdb) si
0xffffffff814fffae in ?? ()
(gdb) si
0xffffffff814fffb1 in ?? ()
(gdb)

[ 7.968014] iommu: Default domain type: Translated
[ 7.968578] iommu: DMA domain TLB invalidation policy: lazy mode
[ 7.970270] SCSI subsystem initialized
[ 7.971177] libata version 3.00 loaded.
[ 7.971805] ACPI: bus type USB registered
[ 7.972689] usbcore: registered new interface driver usbfs
[ 7.973653] usbcore: registered new interface driver hub
[ 7.974592] usbcore: registered new device driver usb
[ 7.975703] pps_core: LinuxPPS API ver. 1 registered
[ 7.976579] pps_core: Software ver. 5.3.6 - Copyright 2005-2007 Rodolfo Giometti <giometti@linux.it>
[ 7.977610] PTP clock support registered
20.317785600s INFO vmbus_server::channels: Guest negotiated version vtl=0x0 version=VersionInfo { version: Iron, feature_flags: FeatureFlags { guest_specified_signal_parameters: false, channel_interrupt_redirection: false, modify_connection: false, confidential_channels: false } } client_id=00000000-0000-0000-0000-000000000000 trusted=false
[ 7.980580] hv_vmbus: Vmbus version:5.3
20.359039900s INFO vmbus_server::channels: sending offer to guest channel_id=0x1 connection_id=0x2001 key={0e0b6031-5213-4934-818b-38d90ced39db}-{b6650ff7-33bc-4840-8048-e0676786f393}-0
20.359373500s INFO vmbus_server::channels: sending offer to guest channel_id=0x2 connection_id=0x2002 key={44c4f61d-4444-4400-9d52-802e27ede19f}-{678e2bee-f7d6-4d96-9643-d6f9e4518d4f}-0
20.359698500s INFO vmbus_server::channels: sending offer to guest channel_id=0x3 connection_id=0x2003 key={ba6163d9-04a1-4d29-b605-72e2ffb1dc7f}-{ba6163d9-04a1-4d29-b605-72e2ffb1dc7f}-0
[ 7.982066] Advanced Linux Sound Architecture Driver Initialized.
[ 7.983877] NetLabel: Initializing
[ 7.984577] NetLabel: domain hash size = 128
[ 7.985582] NetLabel: protocols = UNLABELED CIPSOv4 CALIPSO
[ 7.986600] NetLabel: unlabeled traffic allowed by default
[ 7.987778] PCI: Using ACPI for IRQ routing
[ 7.988578] PCI: System does not support PCI
[ 7.989674] vg 20.718174400s INFO vmbus_server::channels: sending offer to guest channel_id=0x5 connection_id=0x22005 key={0000170c-facb-11e6-bd58-64006a7986d3}-{4dd010a9-cf76-478a-b58f-26b2dabe1813}-0
20.718547400s INFO vmbus_server::channels: new channel offer_id=OfferId(4) key={0000170c-facb-11e6-bd58-64006a7986d3}-{4dd010a9-cf76-478a-b58f-26b2dabe1813}-0 confidential=true
a 20.720207400s INFO vmbus_server::channels: opened channel offer_id=0x4 channel_id=0x5 result=0
a 20.729941500s INFO underhill_log: inner_target="debug_worker" "GDB client connected" fields={"architecture": "X86_64", "address": "2:1305481385"} extra={"timestamp": "17.870818600s"}
20.731312300s INFO underhill_log: inner_target="vmm_core::partition_unit::debug" "debugger attached" fields={} extra={"timestamp": "17.880894000s"}
r 20.743463000s INFO underhill_log: inner_target="debug_worker" "got initial breakpoint" fields={"reason": "Break"} extra={"timestamp": "17.887689000s"}
20.744987800s INFO underhill_log: inner_target="log" "Unknown command: Ok(\\"vMustReplyEmpty\\")" fields={"log.file": "/home/coo/.cargo/registry/src/index.crates.io-6f17d22bba15001f/gdbstub-0.6.6/src/stub/core_impl.rs", "log.line": 242, "log.module_path": "gdbstub::stub::core_impl", "log.target": "gdbstub::stub::core_impl"} extra={"timestamp": "17.894483600s"}
20.807217500s INFO underhill_log: inner_target="log" "Unknown command: Ok(\\"qTStatus\\")" fields={"log.line": 242, "log.target": "gdbstub::stub::core_impl", "log.file": "/home/coo/.cargo/registry/src/index.crates.io-6f17d22bba15001f/gdbstub-0.6.6/src/stub/core_impl.rs", "log.module_path": "gdbstub::stub::core_impl"} extra={"timestamp": "17.956481600s"}
[]
```



openvmm (VMM worker)

- Main process that acts as a VMM for the guest
- Handle exits from the platform
- Per vCPU executor hosting async tasks
- Interacts with mshv_vtl driver to perform VMM functions
 - Modifying register state
 - Accessing ram via mmap

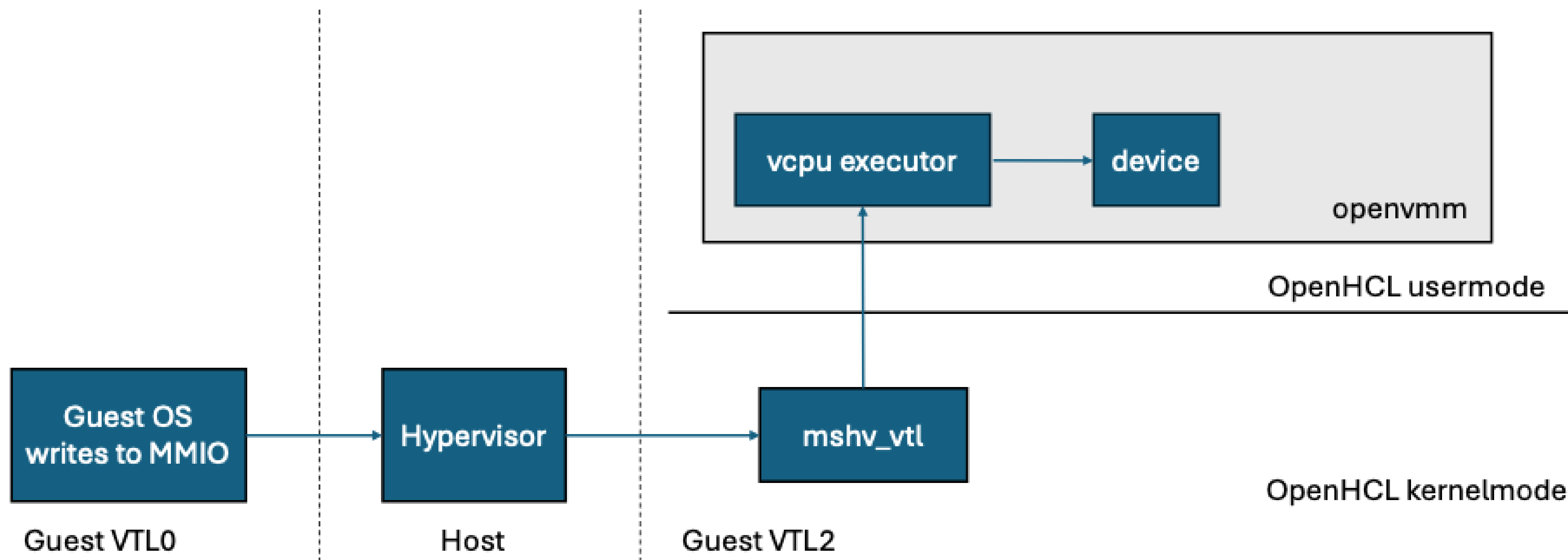


Other utility processes

- `openhcl_dump`
 - Collect core dumps of processes and write them to `openhcl_crash`
 - Separate process to allow dumping any other process in the system
- `openhcl_crash`
 - Write core dumps of usermode crashes to the host via `hvsock`



MMIO dispatch flow



Future roadmap

- Open source later this year
- Support ARM CCA
- Support KVM as host
- Support hosting devices in separate processes
 - IE sandbox vTPM from other devices



Q/A



**LINUX
PLUMBERS
CONFERENCE** Vienna, Austria / Sept. 18-20, 2024