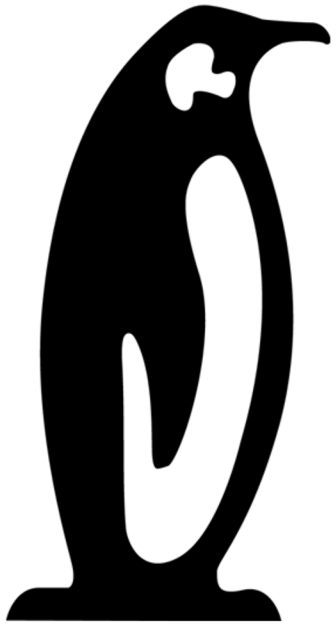


Linux Plumbers Conference

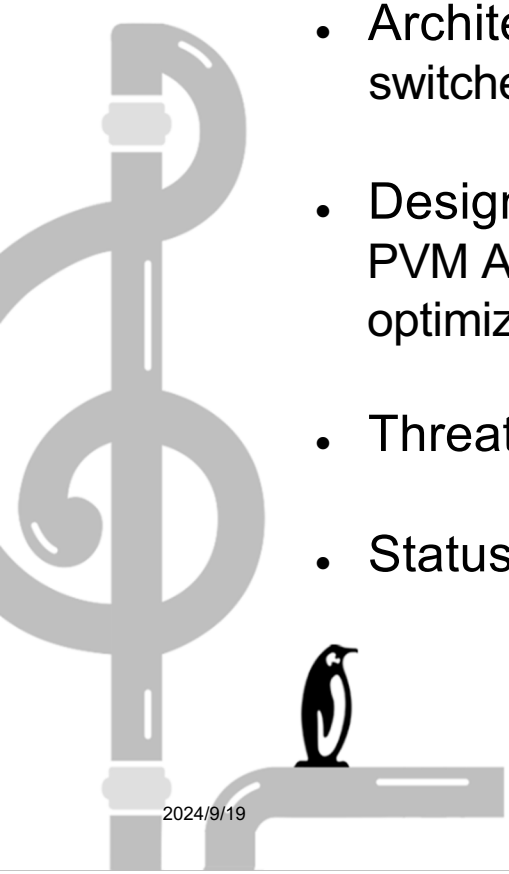
Vienna, Austria | September 18-20, 2024

Pagetable-based Virtual Machine (PVM) as a PV flavor for KVM

Jiangshan Lai (Ant Group)
Wenlong Hou (Ant Group)

Overview

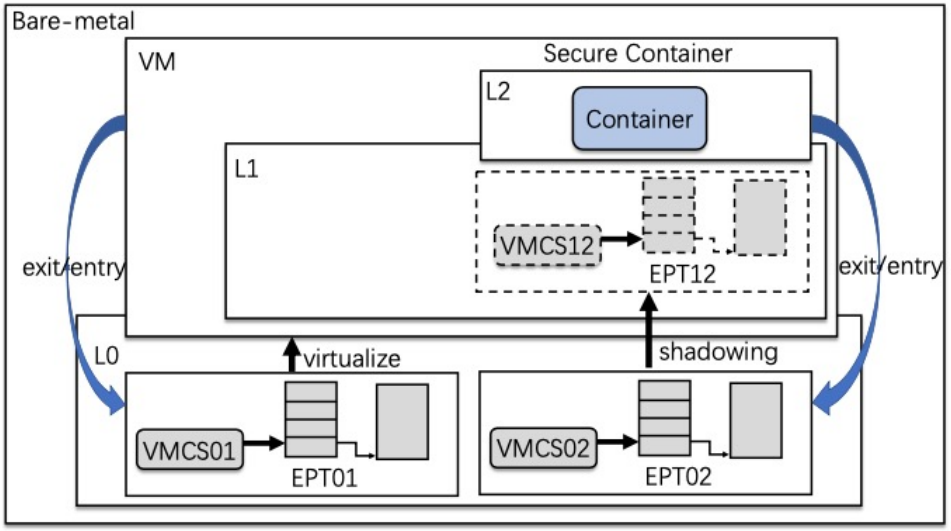
- Motivation
  A PV flavor for KVM
- Architecture
  switcher/guest/hypervisor
- Design details
  PVM ABI/Spec
  optimizations on KVM SPT
- Threat model
- Status and Future works



Motivation

- Run secure containers on spot instance
  - Nested virtualization is disabled, i.e., no hardware assistance.
  - L0 security
    TCB becomes larger and increases L0 attack surface.
  - High world switching costs
    Full emulation involves L0/L1/L2. e.g., shadow EPT emulation.

- A new implementation
  - No L0 involved, L1 <-> L2
  - Customized, not generic



Pagetable-based Virtual Machine (PVM)

- PV flavor for KVM (x86)
  - software-implemented

    The infrastructure is almost complete in kernel.
    Guest: PVOPS
    KVM: SPT/instruction emulator/APIC emulation

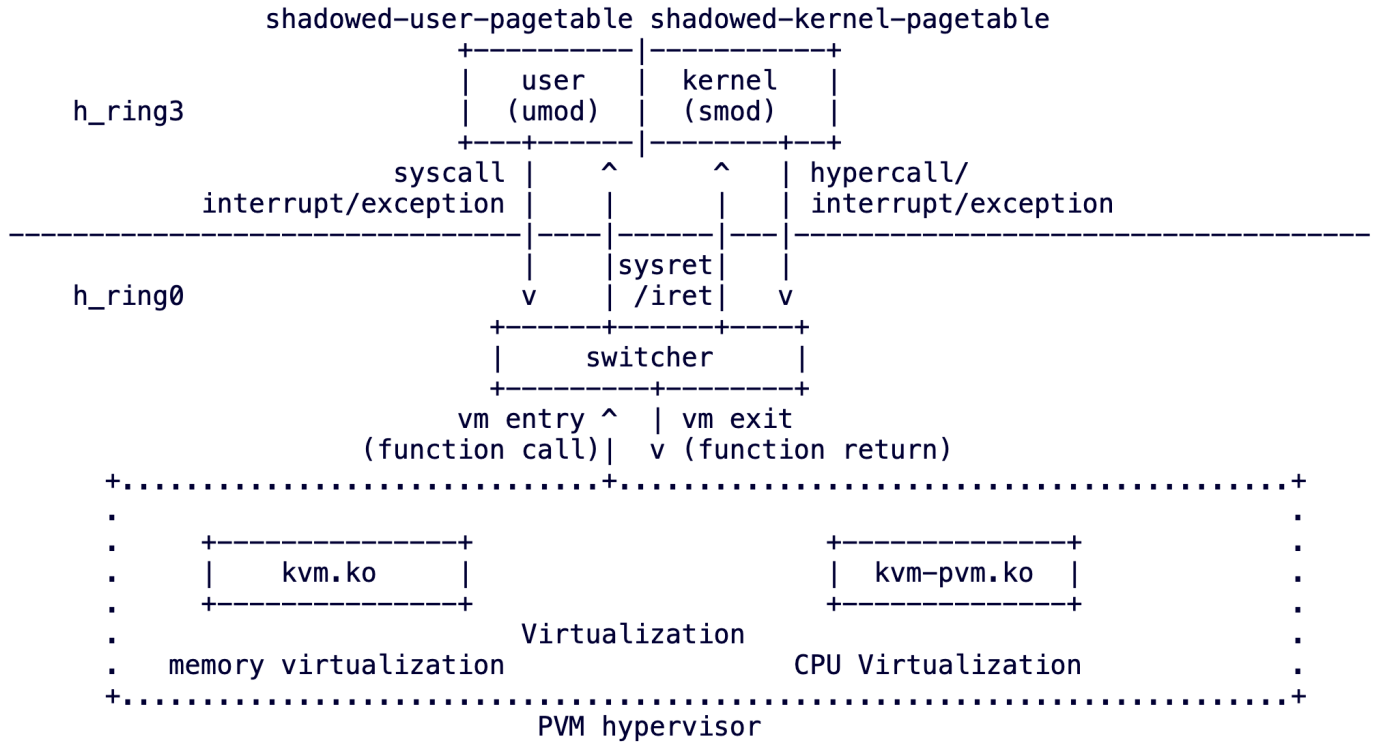
  - simple x86 spec (target future)

    No history burdens, e.g. no IDT (FRED) and no 32-bit supervisor mode (x86-S).

- Pagetable-based

  pagetable-based privilege separation and emulation
  pagetable-based virtual memory emulation

Architecture



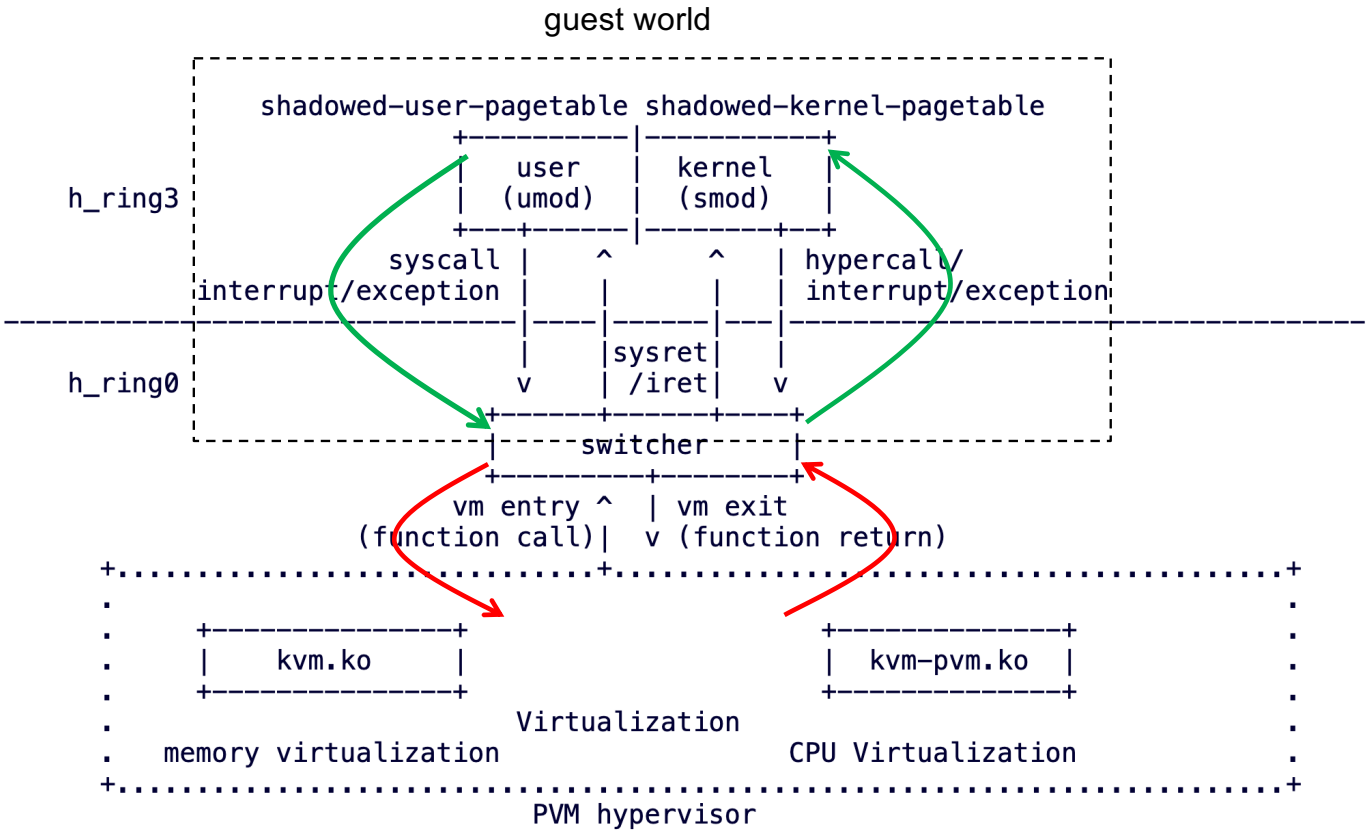
Switcher

- Integral entry
  PVM guest shares the host entries with userspace process.
  - Advantage
    Full control, Zero overhead incurred, Integral (e.g., mitigations)
  - Issues
    Integral entry is not fully implemented: atomic-ist-entry
    https://lore.kernel.org/lkml/20230403140605.540512-1-jiangshanlai@gmail.com/

- tss_extra
  host/guest states and control field -> VMCS/VMCB
  accessible in guest pagetable -> tss_struct

- Direct switching
  Handle VM-exit events directly in switcher. (e.g., mode switching)

Direct Switching



PVM Guest

- Exclusive address space separation
  - PIE kernel

    Exclusive with host kernel image address (in user space for LASS)

    RFC: https://lore.kernel.org/lkml/cover.1682673542.git.houwenlong.hwl@antgroup.com

    impact: Size (+1%~+%2), Performance (same)

    reply: use case, improving security is not enough

    status: 64-bit kernel is ready for PIE building

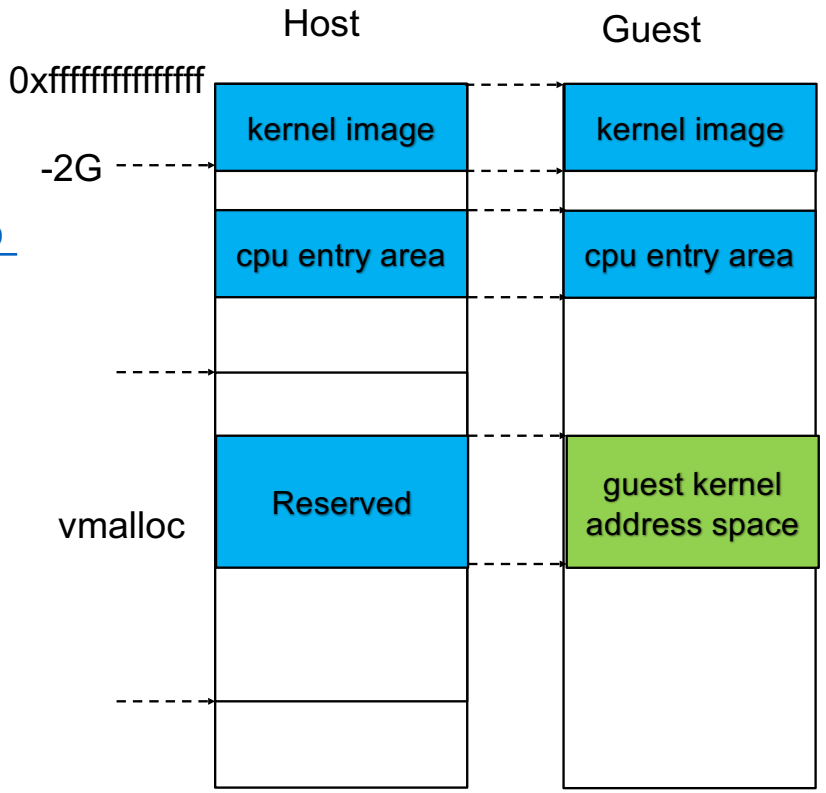
  - Customized kernel address space

    Arrange guest kernel address space within the allowed range.

    - Limitations

      smaller physical address space

      no KASAN



PVM Guest

- PV guest identifier
  - Detection
    hardware CS + synthetic CPUID instruction (INVLPG)
  - X86_FEATURE_PV_GUEST
    common features shared with XENPV (e.g., paravirt patching)

- Event delivery
  self contained, try to follow FRED

- Misc
  kvm_hypercall (syscall)
  vdso_getcpu (lsl, rdtscp)
  vsyscall (XONLY)

PVM Hypervisor

- Atomic event injection and delivery
  inject_irq/nmi/exception callbacks

- Host MMU
  - clone host mappings
    swapper_pg_dir -> host_mmu_root_pgd (template)
    pti_init() -> clone host mappings during module loading
    pgd_ctor() -> copy mappings into root SP
  - allowed range
    Prevent overwriting host mappings.

- Host PCID pool

- PVCS
  Save/restore context during event delivery
  shared memory, PFN cache (KVM_GUEST_AND_HOST_USE_PFN)

Issue: GUEST_USE_PFN (dropped), memory hotplug, KPTI
Plan: PerVM area, one PGD (4/5 PTs)

SPT optimizations

- Problem
  write protection, sync/unsync -> write lock
- Optimizations (Done)
  - Prelink kernel SP
  - Lockless get_root/put_root
  - Unsync_pages list
  - Lockless IVNLPG
  - PV MMU
  - Parallel #PF handling
- Optimizations (TODO)
  - Finer-grain TLB flushing
  - Move TLB flushing out of MMU lock

Threat Model

- Security problem to XENPV
  speculation side channel attacks

- Protect host from guest in PVM
  integral entry -> mitigations used between userspace and kernel
  vcpu_load/vm_entry -> mitigations used for host/KVM guest
- Protect guest kernel from guest userspace in PVM
  all native kernel/guest mitigations
  thread switching mitigations
- Host state leak on Meltdown
  - PVCS/tss_extra
  - GDT (pcpu)

Status

- Opensource progress
  - Github
    - Contributors: loopholes and Tencent Cloud
  - Kata Container PTG

- New use cases
  - Migrate applications between different public cloud providers
  - Fass

- Changes from V1
  - Bug fixes
  - ABI optimization
  - New features (e.g, LA57)
  - PVM in TDX guest

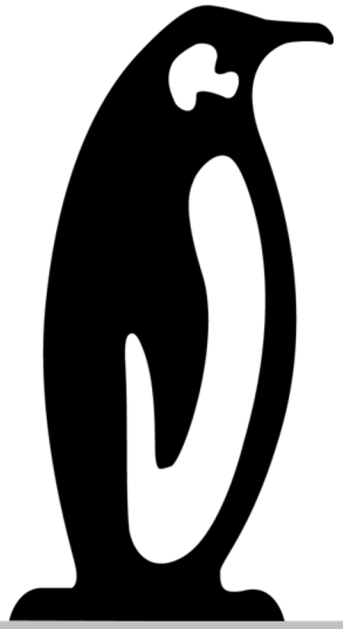
Future key works

- PMU virtualization
  pmu_intel.ko/pmu_amd.ko
- #VE/#VC handling
  support nested virtualization in TDX/SEV guest
- Time
  ABI/Spec (e.g., tsc in migration)
- Multi KVM
  PVM coexists with VMX/SVM
- ASI
  Handle PVM VM-Exit events without switching hardware CR3 to kernel CR3
- More architectures (ARM64/RISCV)

References

[pvm-rfc]: https://lore.kernel.org/kvm/20240226143630.33643-1-jiangshanlai@gmail.com/
[sosp-2023-acm]: https://dl.acm.org/doi/10.1145/3600006.3613158
[sosp-2023-pdf]: https://github.com/virt-pvm/misc/blob/main/sosp2023-pvm-paper.pdf
[sosp-2023-slides]: https://github.com/virt-pvm/misc/blob/main/sosp2023-pvm-slides.pptx
[asm-to-c]: https://lore.kernel.org/lkml/20211126101209.8613-1-jiangshanlai@gmail.com/
[atomic-ist-entry]: https://lore.kernel.org/lkml/20230403140605.540512-1-jiangshanlai@gmail.com/
[pie-patchset]: https://lore.kernel.org/lkml/cover.1682673542.git.houwenlong.hwl@antgroup.com
[linux-pie]: https://github.com/virt-pvm/linux/tree/pie
[linux-pvm]: https://github.com/virt-pvm/linux/tree/pvm
[pvm-get-started]: https://github.com/virt-pvm/misc/blob/main/pvm-get-started-with-kata.md
[pvm-get-started-nested-in-vm]: https://github.com/virt-pvm/misc/blob/main/pvm-get-started-with-kata.md#verify-kata-containers-with-pvm-using-vm-image
[pvm-TODO]: https://github.com/virt-pvm/linux/issues/12

Test and contribute!
Thanks!



Linux Plumbers Conference

Vienna, Austria | September 18-20, 2024