# Xen on x86, 15 years later

Recent development, future direction

QEMU Deprivileging

PVShim

Panopticon

NVDIMM

Large guests (288 vcpus)

PVH Guests

PVCalls

VM Introspection / Memaccess

PV IOMMU

ACPI Memory Hotplug

PVH dom0

Posted Interrupts

KConfig

Sub-page protection

Hypervisor Multiplexing

# Talk approach

- Highlight some key features
    - Recently finished
    - In progress
    - Cool Idea: Should be possible, nobody committed to working on it yet
- Highlight how these work together to create interesting theme

- PVH (with PVH dom0)
- KConfig
  - … to disable PV
- PVshim
- Windows in PVH

## PVH: Finally here

- Full PVH DomU support in Xen 4.10, Linux 4.15
    - First backwards-compatibility hack
- Experimental PVH Dom0 support in Xen 4.11

## PVH: What is it?

- Next-generation paravirtualization mode
  - Takes advantage of hardware virtualization support
  - No need for emulated BIOS or emulated devices
  - Lower performance overhead than PV
  - Lower memory overhead than HVM
  - More secure than either PV or HVM mode

- PVH (with PVH dom0)
- KConfig
  - … to disable PV
- PVshim
- Windows in PVH

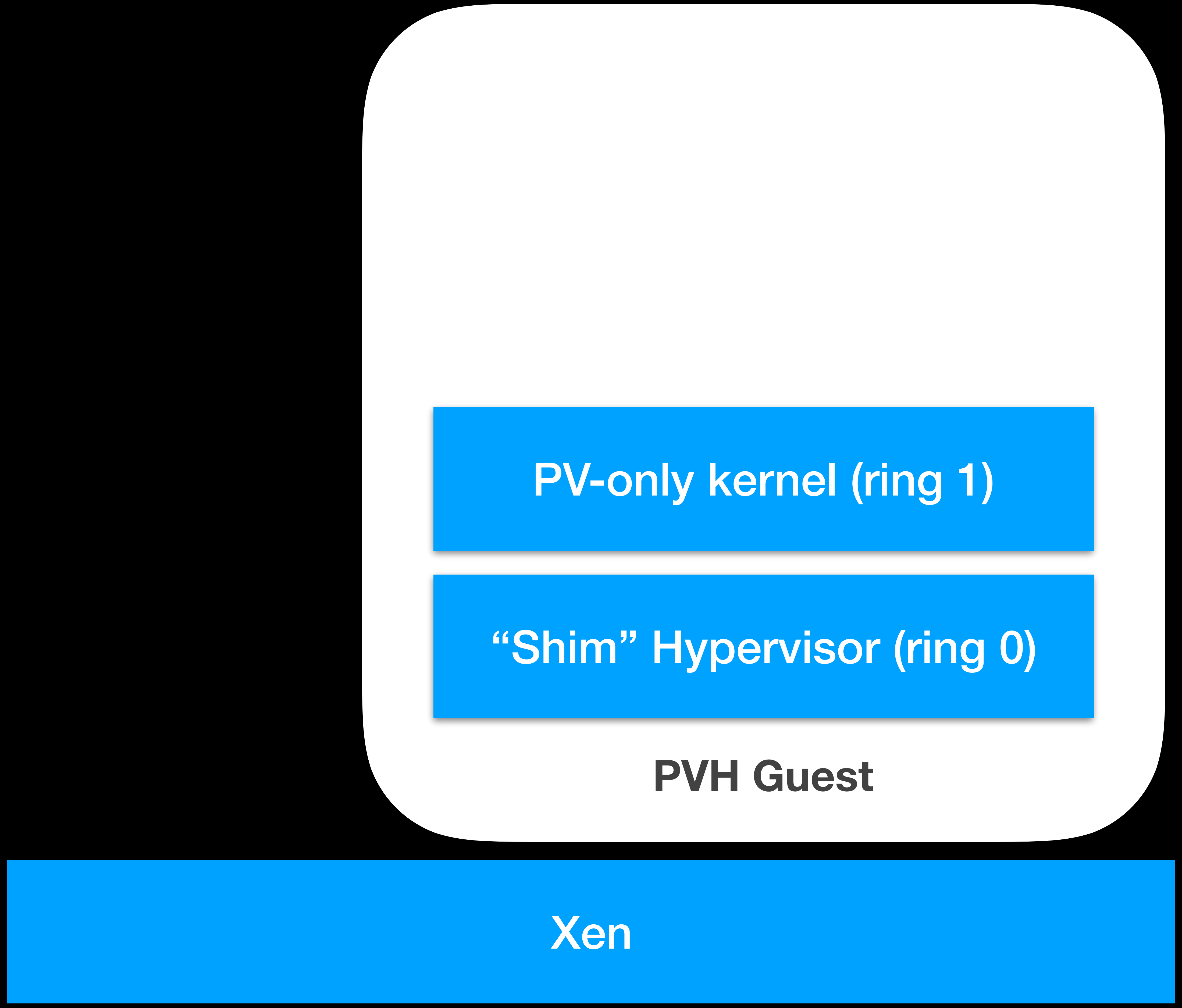
# KConfig

- KConfig for Xen allows…
  - Users to produce smaller / more secure binaries
  - Makes it easier to merge experimental functionality
- KConfig option to disable PV entirely

- PVH
- KConfig
  - … to disable PV
- PVshim
- Windows in PVH

# PVShim

- Some older kernels can only run in PV mode
    - Expect to run in ring 1, ask a hypervisor to perform privileged actions
- “Shim”: A build of Xen designed to allow an unmodified PV guest to run in PVH mode
- `type='pvh'` / `pvshim=1`



- PVH
- KConfig
  - … to disable PV
- PVshim
- Windows in PVH

## No-PV Hypervisors

- PVH
- KConfig
  - … to disable PV
- PVshim
- Windows in PVH

## Windows in PVH

- Windows EFI should be able to do
- OVMF (Virtual EFI implementation) already has
  - PVH support
  - Xen PV disk, network support
- Only need PV Framebuffer support…?

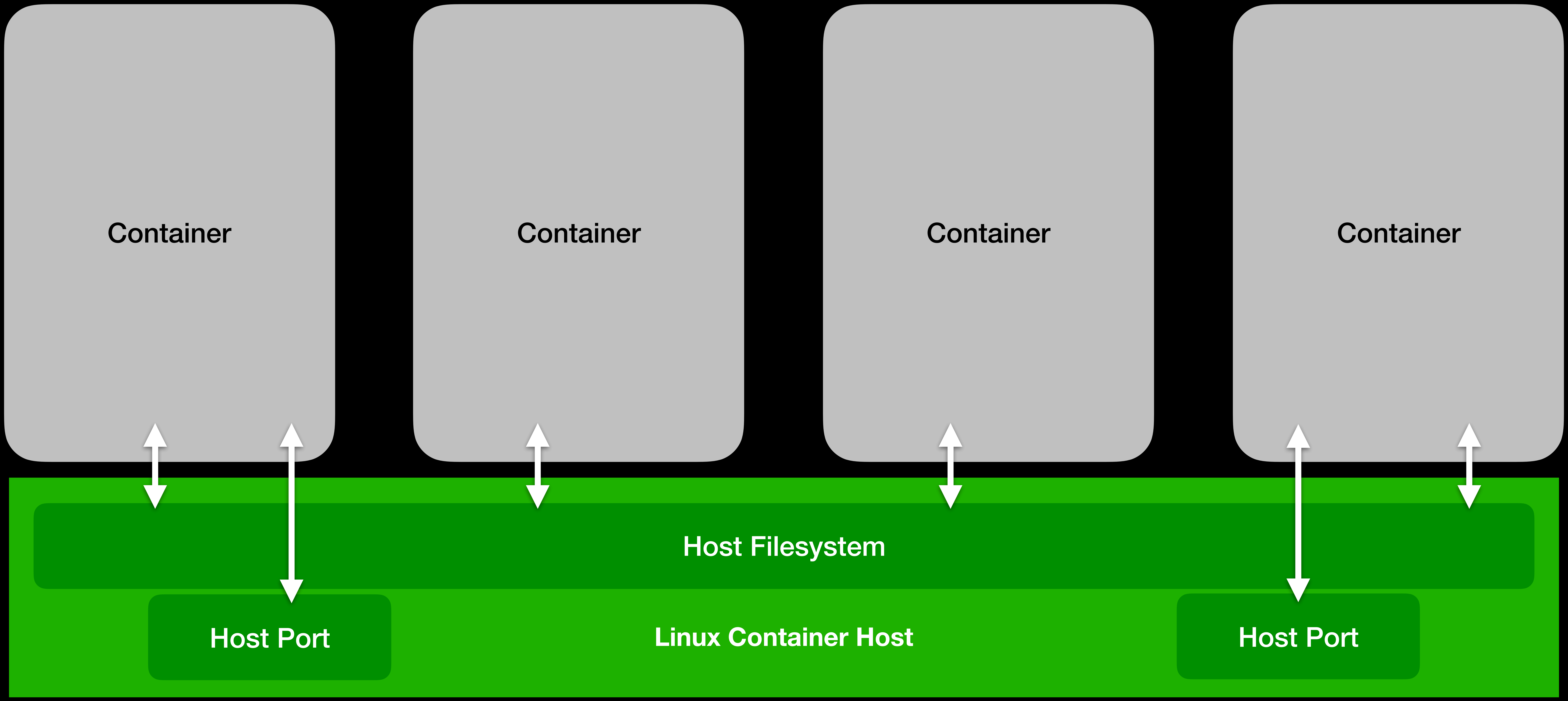
## One guest type to rule them all

- PVH
- KConfig
  - … to disable PV
- PVshim
- Windows in PVH

## Is PV mode obsolete then?

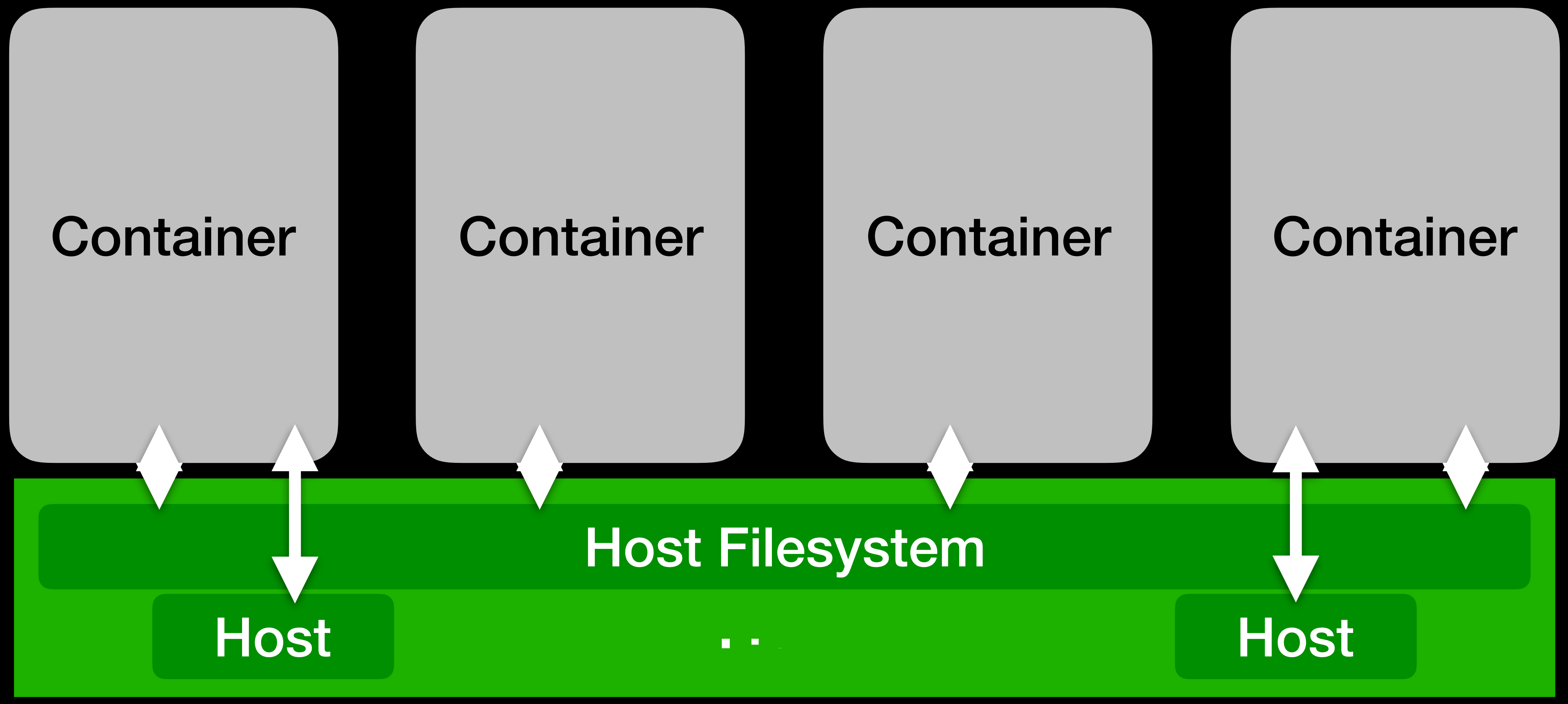
- KConfig: No HVM
- PV 9pfs
- PVCalls
- rkt Stage 1
- Hypervisor Multiplexing

## Containers: Passing through “host” OS resources



## Containers: Passing through “host” OS resources

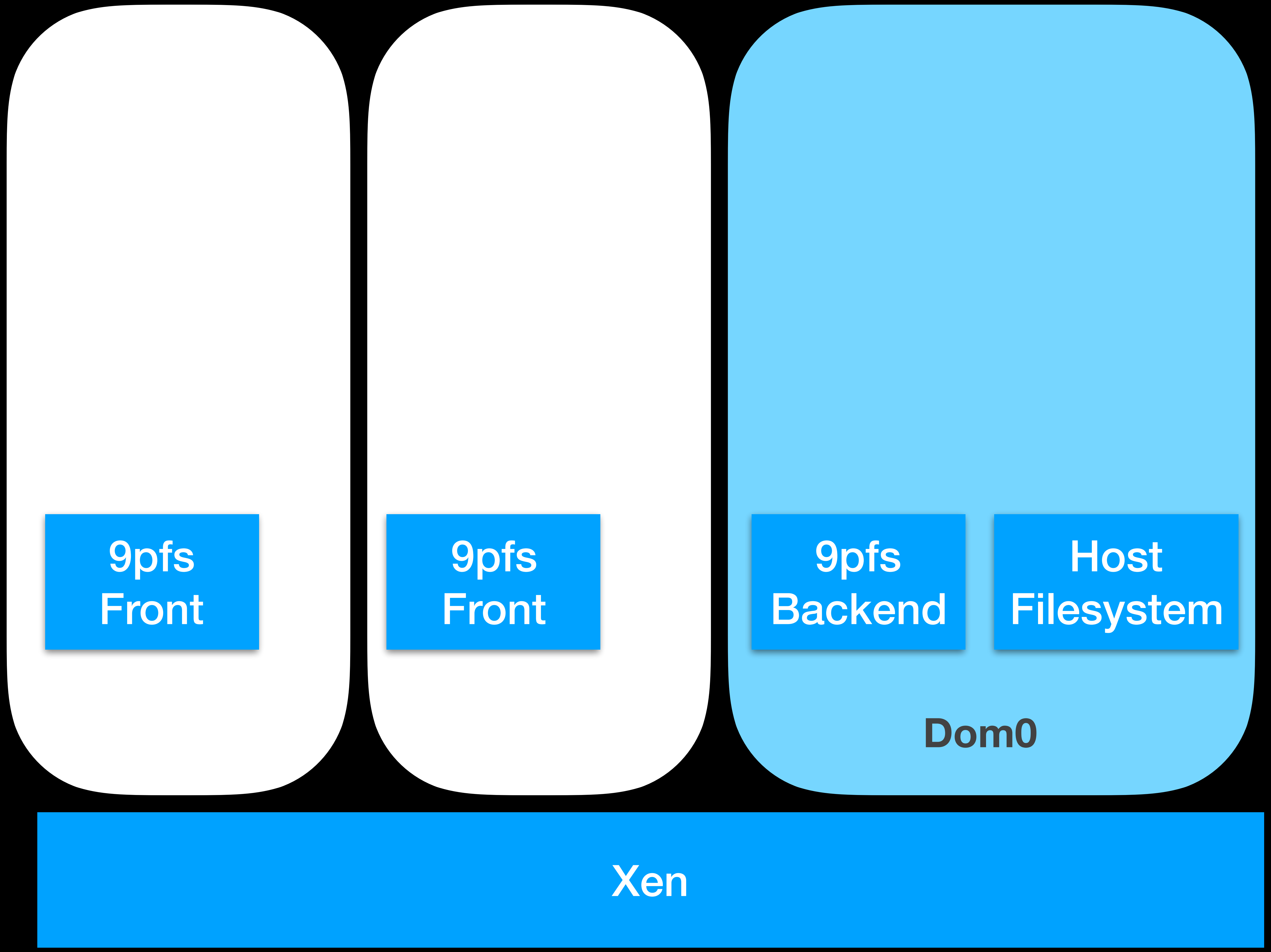
- Allows file-based difference tracking rather than block-based
- Allows easier inspection of container state from host OS
- Allows setting up multiple isolated services without needing to mess around with multiple IP addresses



- KConfig: No HVM
- PV 9pfs
- PVCalls
- rkt Stage 1
- Hypervisor Multiplexing

## PV 9pfs

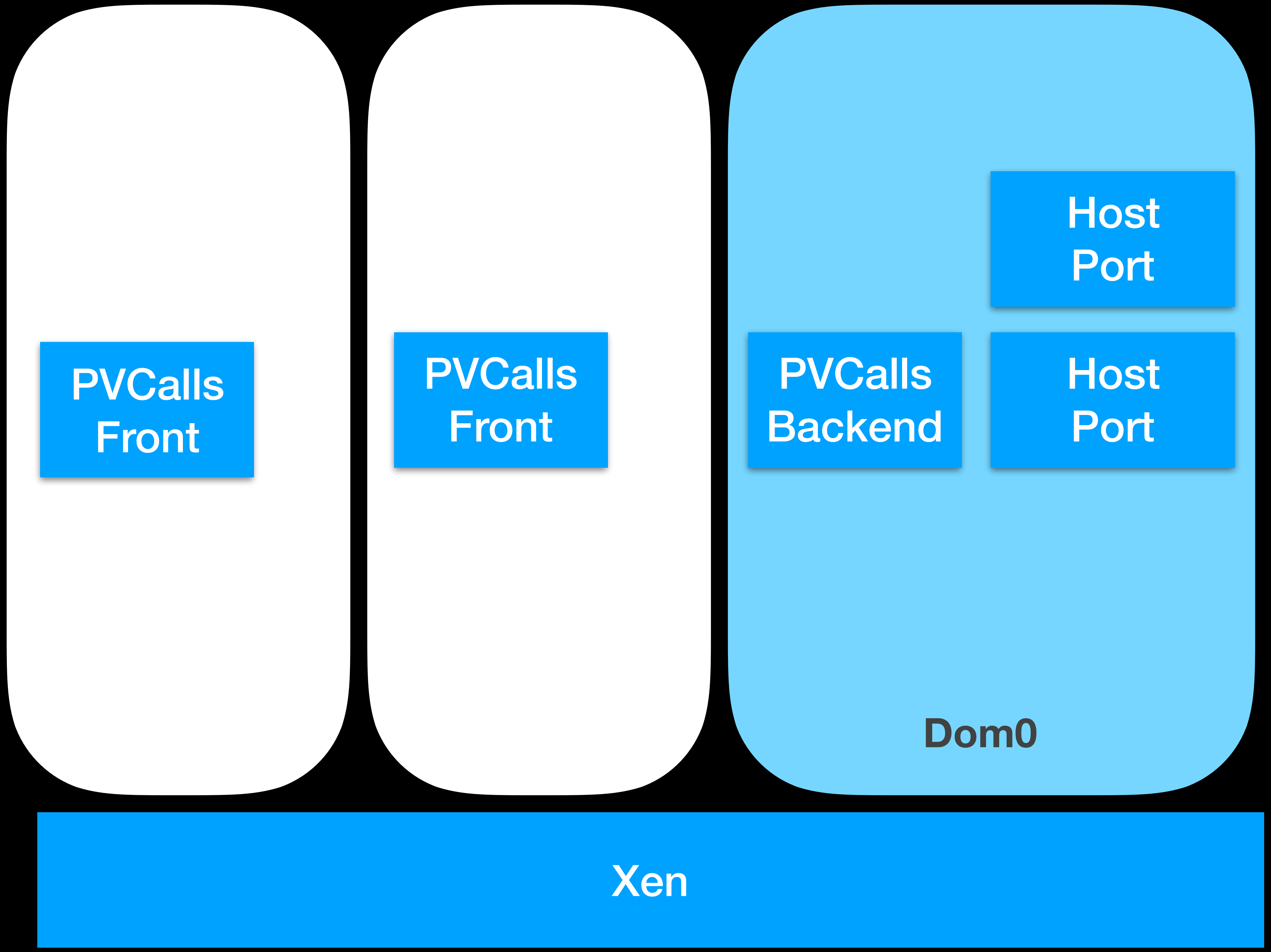
- Allows dom0 to expose files directly to guests



- KConfig: No HVM
- PV 9pfs
- PVCalls
- “Stage 1 Xen”
- Hypervisor Multiplexing

# PV Calls

- Pass through specific system calls
  - `socket()`
  - `listen()`
  - `accept()`
  - `read()`
  - `write()`



- KConfig: No HVM
- PV 9pfs
- PVCalls
- “Stage 1 Xen”
- Hypervisor Multiplexing

## rkt Stage 1

- rkt: “Container abstraction” part of CoreOS
- Running rkt containers (part of CoreOS) under Xen

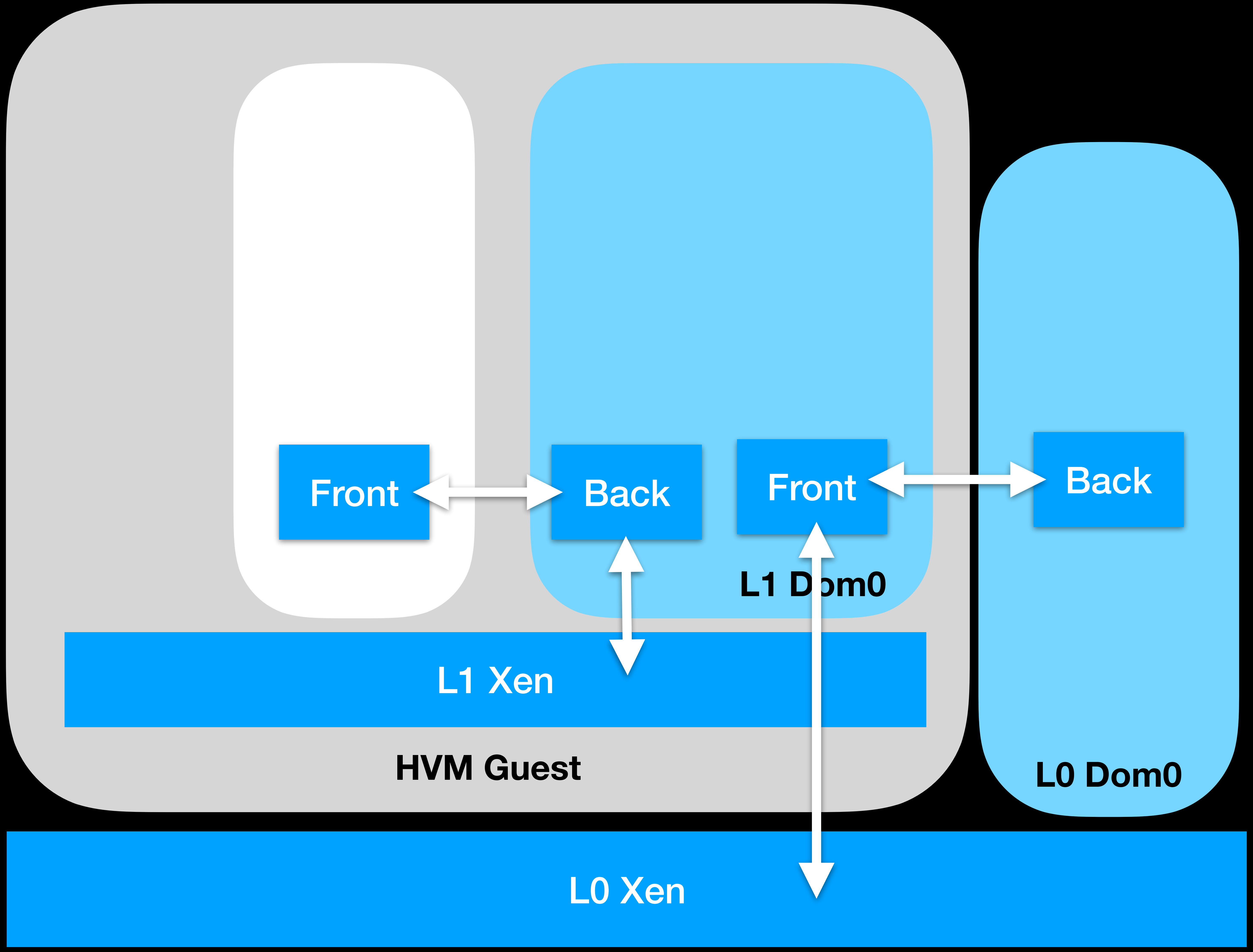
## Xen as full Container Host

- KConfig: No HVM
- PV 9pfs
- PVCalls
- rkt Stage 1
- Hypervisor Multiplexing

- KConfig: No HVM
- PV 9pfs
- PVCalls
- “Stage 1 Xen”
- Hypervisor Multiplexing

# Hypervisor multiplexing

- Xen can run in an HVM guest /without nested HVM support/
- PV protocols use xenbus + hypercalls
- At the moment, Linux code assumes only one xenbus / hypervisor
    - Host PV drivers
    - OR Guest PV drivers
- Multiplexing: Allow both



# Xen as Cloud-ready Container Host

- KConfig: No HVM
- PV 9pfs
- PVCalls
- “Stage 1 Xen”
- Hypervisor Multiplexing

QEMU Deprivileging

PVShim

Panopticon

NVDIMM

Large guests (288 vcpus)

PVH Guests

PVCalls

VM Introspection / Memaccess

PV IOMMU

ACPI Memory Hotplug

PVH dom0

Posted Interrupts

KConfig

Sub-page protection

Hypervisor Multiplexing

## QEMU Deprivileging

- Restricting hypercalls to a single guest
- Restricting what QEMU can do within dom0

## Panopticon / No Secrets

- Spectre-style information leaks
- You can only leak what you can see
- Xen has all of physical memory mapped
  - But this is not really necessary
- Assume that all guests can read hypervisor memory at all times

QEMU Deprivileging

PVShim

Panopticon

NVDIMM

Large guests (288 vcpus)

PVH Guests

PVCalls

VM Introspection / Memaccess

PV IOMMU

ACPI Memory Hotplug

PVH dom0

Posted Interrupts

KConfig

Sub-page protection

Hypervisor Multiplexing

## Questions